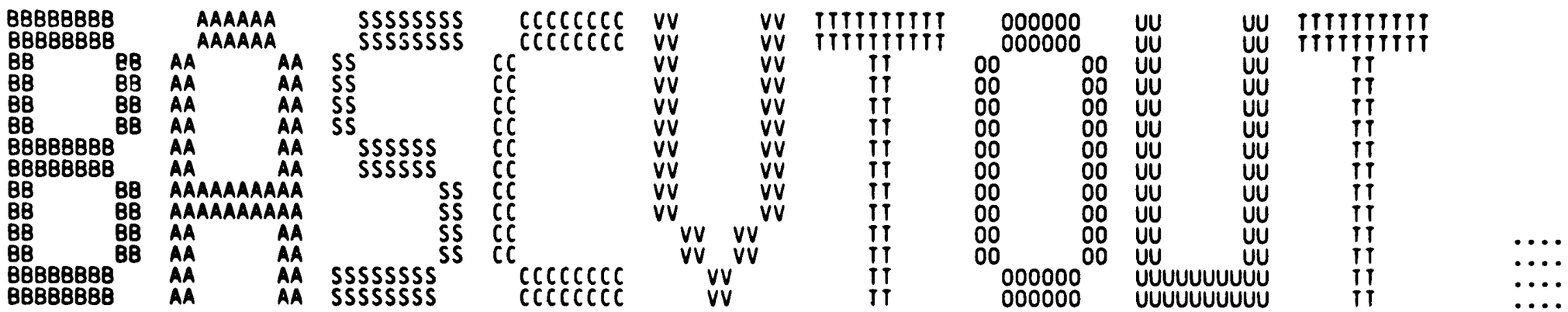
```
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBB         BBB   AAA         AAA   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 CCCCCCCC VV VV TTTTTTTTTT OOOOOO UU UU TTTTTTTTTT
BBBBBBBB AAAAAA SSSSSSSS CCCCCCCC VV VV TTTTTTTTTT OOOOOO UU UU TTTTTTTTTT
BB BB AA AA SS CC VV VV TT OO OO UU UU TT
BB BB AA AA SS CC VV VV TT OO OO UU UU TT
BB BB AA AA SS CC VV VV TT OO OO UU UU TT
BB BB AA AA SS CC VV VV TT OO OO UU UU TT
BBBBBBBB AA AA SSSSSS CC VV VV TT OO OO UU UU TT
BBBBBBBB AA AA SSSSSS CC VV VV TT OO OO UU UU TT
BB BB AAAAAAAAAA SS CC VV VV TT OO OO UU UU TT
BB BB AAAAAAAAAA SS CC VV VV TT OO OO UU UU TT
BB BB AA AA SS CC VV VV TT OO OO UU UU TT
BB BB AA AA SS CC VV VV TT OO OO UU UU TT
BBBBBBBB AA AA SSSSSSSS CCCCCCCC VV TT OOOOOO UUUUUUUUUU TT
BBBBBBBB AA AA SSSSSSSS CCCCCCCC VV TT OOOOOO UUUUUUUUUU TT
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
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS

```
;     1          0001  0 MODULE BAS$CVT_OUT (                      ! Convert single or double float to Basic E, F, or G
;     2          0002  0                 IDENT = '1-054'           ! File:BASCVTOUT.B32 Edit: STAN1054
;     3          0003  0                 ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !**************************************************************************
;     7          0007  1 !*                                                                        *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                              *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                  *
;    11          0011  1 !*                                                                        *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;    17          0017  1 !*  TRANSFERRED.                                                           *
;    18          0018  1 !*                                                                        *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;    21          0021  1 !*  CORPORATION.                                                           *
;    22          0022  1 !*                                                                        *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                *
;    25          0025  1 !*                                                                        *
;    26          0026  1 !*                                                                        *
;    27          0027  1 !**************************************************************************
;    28          0028  1 !
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 ! FACILITY: VAX-11 BASIC support
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       Convert a double or single precision floating point number to E, F, or
;    36          0036  1 !       G format for VAX-11 BASIC.  These routines will support PRINT, PRINT USING,
;    37          0037  1 !       NUM$, NUM1$, STR$, and FORMAT$.  For E and F format, if the input value
;    38          0038  1 !       is single precision, a maximum of 6 significant digits are returned; for
;    39          0039  1 !       a double precision value, a maximum of 16 significant digits are returned.
;    40          0040  1 !
;    41          0041  1 ! ENVIRONMENT: User mode - AST reentrant.
;    42          0042  1 !
;    43          0043  1 ! AUTHOR: Donald Petersen, CREATION DATE: 26-Jun-79
;    44          0044  1 !
;    45          0045  1 ! MODIFIED BY:
;    46          0046  1 !
;    47          0047  1 !       DGP, 26-Jun-79 : VERSION 1
;    48          0048  1 ! 1-001 - original
;    49          0049  1 ! 1-003 - Change algorithm for determinig the number of significant digits
;    50          0050  1 !         we ask the kernel routine for.  DGP 16-Jul-79
;    51          0051  1 ! 1-004 - More work on Print Using.  DGP  23-Jul-79
;    52          0052  1 ! 1-005 - Add E formatting for Print Using.  DGP 27-Jul-79
;    53          0053  1 ! 1-006 - Change the conversion macros to JSB routines to save space. DGP 30-Jul-79
;    54          0054  1 ! 1-007 - Continuing development of E format.  DGP 31-Jul-79
;    55          0055  1 ! 1-008 - Add optional arg. to BAS$CVT_OUT_D_T for number of digits in F format
;    56          0056  1 !         before swiching to E.  DGP 07-Sep-79
;    57          0057  1 ! 1-009 - BAS$CVT_OUT_F_F does not set RT_RND when calling kernel conversion
```

```
;    58          0058  1 !                routine if V_FLT_DEC_PT is set in input flags.  DGP 10-Sep-79
;    59          0059  1 ! 1-010 - Load KERNEL_BLOCK [STRING_PTR] before checking for zero.  DGP
;    60          0060  1 !                21-Sep-79
;    61          0061  1 ! 1-011 - Accept currency, radix point, and digit separator symbols as
;    62          0062  1 !                parameters in order to allow the BASIC user to change them.  Can't
;    63          0063  1 !                do the translation here because the format interpreter needs the
;    64          0064  1 !                lengths of some of the symbols.  DGP 30-Oct-79
;    65          0065  1 ! 1-012 - Scale factor is only a byte.  DGP 25-Nov-79
;    66          0066  1 ! 1-013 - Fix bug in scaling in BAS$CVT_OUT_D_F.  DGP 06-Dec-79
;    67          0067  1 ! 1-014 - When calling the kernal conversion routine, bias 'right round' by
;    68          0068  1 !                the scale factor in BAS$CVT_OUT_D_F.  DGP 03-Mar-80
;    69          0069  1 ! 1-015 - Put the BUILTIN ACTUALCOUNT into the routines that need it, rather
;    70          0070  1 !                than at the module level, in anticipation of the next BLISS compiler,
;    71          0071  1 !                which will require it there.  While we are here, make some
;    72          0072  1 !                improvements in the source text.  Note that this edit changes
;    73          0073  1 !                no code.  JBS 27-Aug-1980
;    74          0074  1 ! 1-016 - FIX BAS$CVT_OUT_D_G, PLAIN_E_FORM_11 and PLAIN_F_FORM_11 to return status.  FM 1-OCT-80
;    75          0075  1 ! 1-017 - Fix the PRINT USING, '$$##.##', .01   REJ & FM 1-OCT-80
;    76          0076  1 ! 1-018 - Fix bug of some fraction numbers printing out as zero and correct
;    77          0077  1 !                error checking for E format with negative numbers. Corresponds
;    78          0078  1 !                to bug version 14C, 14D, and 14E.  DGP 23-Feb-81
;    79          0079  1 ! 1-019 - Add support for G floating and H floating.  PLL 21-Aug-81
;    80          0080  1 ! 1-020 - Add PLAIN_E_HFLOAT and FANCY_E_HFLOAT.  PLL 28-Aug-81
;    81          0081  1 ! 1-021 - Add F format for H floating.  PLL 8-Sep-81
;    82          0082  1 ! 1-022 - Fix a bug in fancy_e_hfloat (doesn't handle zero correctly). PLL 20-Oct-81
;    83          0083  1 ! 1-023 - Add PLAIN_E_GFLOAT.  PLL 22-Oct-81
;    84          0084  1 ! 1-024 - E format for h floating should round on the fifth digit. Also,
;    85          0085  1 !                fix PLAIN_F_FORM_11 and FANCY_F_FORM_11 so that NUM1$ of g or h
;    86          0086  1 !                floating can print out all 308 or 4932 digits.  PLL 23-Oct-81
;    87          0087  1 ! 1-025 - The last edit broke NUM1$ for double, so fix PLAIN_F and FANCY_F. PLL
;    88          0088  1 !                26-Oct-81
;    89          0089  1 ! 1-026 - Add support for formatting characters <CD>, <%> and <0>.  PLL 25-Nov-81
;    90          0090  1 ! 1-027 - Collapse some routines into others, to cut down on the size of the
;    91          0091  1 !                module.  PLL 16-Dec-81
;    92          0092  1 ! 1-028 - BAS$CVT_OUT_x_F routines should not require all parameters.  PLL 8-Jan-82
;    93          0093  1 ! 1-029 - Fix typo in last edit.  PLL 8-Jan-1982
;    94          0094  1 ! 1-030 - Add support for packed decimal.  PLL 12-Jan-1982
;    95          0095  1 ! 1-031 - BAS$CVT_OUT_x_G routines should not assume an optional parameter
;    96          0096  1 !                exists.  PLL 3-Feb-82
;    97          0097  1 ! 1-032 - Attempt to fix the case of packed decimal zero.  PLL 12-Feb-82
;    98          0098  1 ! 1-033 - BAS$CVT_P_T is no longer used so remove EXTERNAL declaration.  PLL 17-Feb-82
;    99          0099  1 ! 1-034 - Add nooptimize to list of switches.  This is temporary - the Bliss
;   100          0100  1 !                compiler used for VMS 3.0 optimizes away a line of code.  PLL 22-Feb-82
;   101          0101  1 ! 1-035 - Fix bug in PLAIN_E_FORM_11 - max digits is 5 for hfloat.  PLL 9-Mar-1982
;   102          0102  1 ! 1-036 - Fix bug in Print Using of numbers with trailing zeroes (ex. 100).
;   103          0103  1 !                Leading_str_desc length (used to calculate trailing zeroes) must be
;   104          0104  1 !                initialized to zero.  PLL 10-Mar-1982
;   105          0105  1 ! 1-037 - Negative numbers should not print a minus sign when <CD> was specified
;   106          0106  1 !                by the format string (Print Using).  PLL 10-Mar-1982
;   107          0107  1 ! 1-038 - OTSLNK.REQ should be required from RTLIN:.  PLL 16-Mar-82
;   108          0108  1 ! 1-039 - Fix length of zero_str_desc in fancy_e_form_11.  It should never be
;   109          0109  1 !                negative.  Remove nooptimize.  PLL 22-Mar-1982
;   110          0110  1 ! 1-040 - The case ".##", .001 should round, not produce % .001 (PRINT
;   111          0111  1 !                USING).  PLL 17-Jun-1982
;   112          0112  1 ! 1-041 - Fix the case "#.##", .001.  (Last edit broke this.)  PLL 21-Jun-1982
;   113          0113  1 ! 1-042 - The case ".##", 0 should produce .00, not %0.  PLL 27-Jul-1982
;   114          0114  1 ! 1-043 - Fix the case "###", 0, which the last edit broke.  Two macros for
```

```
;  115          0115  1 !          zero are needed - one for the case of a fractional format, and one
;  116          0116  1 !          for formats with an integer part.
;  117          0117  1 !          Also fix COMMON_F for NUM1$(0), which was producing ".0" instead of
;  118          0118  1 !          "0".  PLL 2-Sep-1982
;  119          0119  1 !  1-044 - the macros introduced in edit 043 should set the entire value to
;  120          0120  1 !          zeros, not just the first byte.  MDL 1-Dec-1982
;  121          0121  1 !  1-045 - fix the case "#.#### ", very small number.  We would end up passing
;  122          0122  1 !          a string descriptor with a negative length to STR$CONCAT when called
;  123          0123  1 !          from COMMON_F.  MDL 2-Dec-1982
;  124          0124  1 !  1-046 - Local storage KERNEL_BLOCK should be initialized upon entry into
;  125          0125  1 !          the COMMON_E, COMMON_F and COMMON_G routines; there are code paths
;  126          0126  1 !          that can reference fields in it before it is set up, and would
;  127          0127  1 !          therefore get garbage unless it is initialized.  MDL 7-Dec-1982
;  128          0128  1 !  1-047 - change logic that checks to see if a number is effectively zero
;  129          0129  1 !          very slightly in order to fix bug of printing negative zero.
;  130          0130  1 !          MDL 8-Dec-1982
;  131          0131  1 !  1-048 - change calculation for number of needed leading zeros slightly
;  132          0132  1 !          in FANCY_F_FORM_11 routine.  MDL 10-Dec-1982
;  133          0133  1 !  1-049 - fix minor bug in edit 1-044; macro PUT_OUT_ZERO should zero out
;  134          0134  1 !          NO_INT_DIGITS + NO_FRAC_DIGITS, not just NO_FRAC_DIGITS.  Also,
;  135          0135  1 !          back out edit 1-048, it broke too many things.  MDL 10-Jan-1983
;  136          0136  1 !  1-050 - add ANSI print support code. The only difference is in E-format
;  137          0137  1 !          numbers, where the significand is between 1 & 10 rather than 0 & 1.
;  138          0138  1 !          See PLAIN_E_FORM11 for details.  MDL 29-Mar-1983
;  139          0139  1 !  1-051 - repair minor bug introduced in edit 1-050.  cannot assume that
;  140          0140  1 !          OTS$$AA_CUR_LUB is defined.  MDL 20-Apr-1983
;  141          0141  1 !  1-052 - for PRINT USING, a currency field can end with <CD> as well as
;  142          0142  1 !          with a minus sign in order to print a negative number.
;  143          0143  1 !          MDL 27-Sep-1983
;  144          0144  1 !  1-053 - correct formatting for DECIMAL numbers that occupy less places
;  145          0145  1 !          than reserved.  MDL 24-Feb-1984
;  146          0146  1 !  1-054 - Remove informational errors. STAN 24-Jul-1984.
;  147          0147  1 !--
;  148          0148  1
;  149          0149  1 !<BLF/PAGE>
```

```
;  151          0150  1 !
;  152          0151  1 ! SWITCHES
;  153          0152  1 !
;  154          0153  1
;  155          0154  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;  156          0155  1
;  157          0156  1 !
;  158          0157  1 ! LINKAGES
;  159          0158  1 !
;  160          0159  1
;  161          0160  1 REQUIRE 'RTLIN:OTSLNK';
;  162          0589  1
;  163          0590  1 REQUIRE 'RTLML:OTSLUB',
;  164          0730  1
;  165          0731  1 LINKAGE
;  166          0732  1     JSB_E_FORMAT = JSB (REGISTER = 0, REGISTER = 1, REGISTER = 2, REGISTER = 3,
;  167          0733  1         REGISTER = 4, REGISTER = 5, REGISTER = 6, REGISTER = 7, REGISTER = 8,
;  168          0734  1         REGISTER = 9, REGISTER = 10) : PRESERVE (11),
;  169          0735  1
;  170          0736  1     JSB_F_FORMAT = JSB (REGISTER = 0, REGISTER = 1, REGISTER = 2, REGISTER = 3,
;  171          0737  1         REGISTER = 4, REGISTER = 5, REGISTER = 6, REGISTER = 7, REGISTER = 8,
;  172          0738  1         REGISTER = 9, REGISTER = 10, REGISTER = 11),
;  173          0739  1
;  174          0740  1     JSB_G_FORMAT = JSB (REGISTER = 0, REGISTER = 1, REGISTER = 2, REGISTER = 3,
;  175          0741  1         REGISTER = 4, REGISTER = 5, REGISTER = 6) : PRESERVE (7, 8, 9, 10, 11);
;  176          0742  1
;  177          0743  1 !
;  178          0744  1 ! TABLE OF CONTENTS:
;  179          0745  1 !
;  180          0746  1
;  181          0747  1 FORWARD ROUTINE
;  182          0748  1     BAS$CVT_OUT_F_E,                        ! Convert float to E format
;  183          0749  1     BAS$CVT_OUT_D_E,                        ! Convert double to E format
;  184          0750  1     BAS$CVT_OUT_F_F,                        ! Convert float to F format
;  185          0751  1     BAS$CVT_OUT_D_F,                        ! Convert double to F format
;  186          0752  1     BAS$CVT_OUT_D_G,                        ! Convert double to G format
;  187          0753  1     BAS$CVT_OUT_G_E,                        ! Convert gfloat to E format
;  188          0754  1     BAS$CVT_OUT_G_F,                        ! Convert gfloat to F format
;  189          0755  1     BAS$CVT_OUT_H_E,                        ! Convert hfloat to E format
;  190          0756  1     BAS$CVT_OUT_G_G,                        ! Convert gfloat to G format
;  191          0757  1     BAS$CVT_OUT_H_G,                        ! Convert hfloat to G format
;  192          0758  1     BAS$CVT_OUT_H_F,                        ! Convert hfloat to F format
;  193          0759  1     BAS$CVT_OUT_P_F,                        ! Convert packed to F format
;  194          0760  1     BAS$CVT_OUT_P_E,                        ! Convert packed to E format
;  195          0761  1     BAS$CVT_OUT_P_G,                        ! Convert packed to G format
;  196          0762  1     COMMON_E : JSB_E_FORMAT,                ! E processing common to all
;  197          0763  1                                             !  data types
;  198          0764  1     COMMON_F : JSB_F_FORMAT,                ! F processing common to all
;  199          0765  1                                             !  data types
;  200          0766  1     COMMON_G : JSB_G_FORMAT,                ! G processing common to all
;  201          0767  1                                             !  data types
;  202          0768  1     FANCY_F_FORM_11 : JSB_FORMAT_A11 NOVALUE,   ! F format for PRINT USING
;  203          0769  1     FANCY_E_FORM_11 : JSB_FORMAT_A11 NOVALUE,   ! E format for PRINT USING
;  204          0770  1     PLAIN_E_FORM_11 : JSB_FORMAT_A6,            ! E format for STR$, NUM$, and PRINT
;  205          0771  1     PLAIN_F_FORM_11 : JSB_FORMAT_A6;            ! F format for STR$, NUM$, NUM1$, and PRINT
;  206          0772  1
;  207          0773  1 !
```

```
;  208          0774  1 ! INCLUDE FILES:
;  209          0775  1 !
;  210          0776  1
;  211          0777  1 REQUIRE 'RTLIN:RTLPSECT';
;  212          0872  1
;  213          0873  1 LIBRARY 'RTLSTARLE';
;  214          0874  1
;  215          0875  1 !
;  216          0876  1 ! MACROS:
;  217          0877  1 !
;  218          0878  1
;  219          0879  1 MACRO
;  220        M 0880  1     RT_RND =
;  221          0881  1  0, 0, 32, 0%,                  ! Value for right round
;  222        M 0882  1     OFFSET =
;  223          0883  1  4, 0, 32, 0%,                  ! Offset to first non-zero digit
;  224          0884  1                                 ! in string returned
;  225        M 0885  1     DEC_EXP =
;  226          0886  1  8, 0, 32, 1%,                  ! Decimal exponent
;  227        M 0887  1     SIGN =
;  228          0888  1  12, 0, 32, 1%,                 ! Sign of returned value
;  229        M 0889  1     STRING_ADDR =
;  230          0890  1  16, 0, 32, 0%,                 ! String address
;  231        M 0891  1     SIG_DIGITS =
;  232          0892  1  20, 0, 32, 0%,                 ! No. of significant digits
;  233        M 0893  1     CONV_FLAGS =
;  234          0894  1  24, 0, 32, 0%,                 ! Flags
;  235        M 0895  1     KERNEL_PTR =
;  236          0896  1  36, 0, 0, 0%;                  ! Addr of block for kernel routine
;  237          0897  1
;  238          0898  1 !<BLF/MACRO>
;  239          0899  1
;  240          0900  1 MACRO
;  241        M 0901  1     PUT_OUT_ZERO =
;  242        M 0902  1         BEGIN
;  243        M 0903  1         KERNEL_BLOCK [OFFSET] = 0;
;  244        M 0904  1         KERNEL_BLOCK [DEC_EXP] = 1;
;  245        M 0905  1         KERNEL_BLOCK [SIGN] = 0;
;  246        M 0906  1         KERNEL_BLOCK [SIG_DIGITS] = 1;
;  247        M 0907  1         NO_DIGITS = 1;
;  248        M 0908  1         (.KERNEL_BLOCK [STRING_ADDR])<0, 8> = %C'0';
;  249        M 0909  1         CH$FILL (%C'0', .NO_INT_DIGITS + .NO_FRAC_DIGITS, CH$PTR(.KERNEL_BLOCK [STRING_ADDR]) );
;  250        M 0910  1         END
;  251          0911  1     %;
;  252          0912  1 MACRO
;  253        M 0913  1     PUT_OUT_ZERO_FRACT =
;  254        M 0914  1         BEGIN
;  255        M 0915  1         KERNEL_BLOCK [OFFSET] = 0;
;  256        M 0916  1         KERNEL_BLOCK [DEC_EXP] = 0;
;  257        M 0917  1         KERNEL_BLOCK [SIGN] = 0;
;  258        M 0918  1         KERNEL_BLOCK [SIG_DIGITS] = 1;
;  259        M 0919  1         NO_DIGITS = 1;
;  260        M 0920  1         (.KERNEL_BLOCK [STRING_ADDR])<0, 8> = %C'0';
;  261        M 0921  1         CH$FILL (%C'0', .NO_FRAC_DIGITS, CH$PTR(.KERNEL_BLOCK [STRING_ADDR]) );
;  262        M 0922  1         END
;  263          0923  1     %;
;  264          0924  1
```

```
;  265          0925  1 !<BLF/PAGE>
```

```
;  267          0926  1
;  268          0927  1 MACRO
;  269        M 0928  1     INITIALIZE_DESC =
;  270        M 0929  1
;  271        M 0930  1         BIND
;  272        M 0931  1             DOT = UPLIT (BYTE ('.')),
;  273        M 0932  1             STAR = UPLIT ( REP K_RANGE OF BYTE ('*')),
;  274        M 0933  1             MINUS = UPLIT (BYTE ('-')),
;  275        M 0934  1             BLANK = UPLIT (BYTE (' ')),
;  276        M 0935  1             E = UPLIT (BYTE ('E')),
;  277        M 0936  1             BLANKS = UPLIT ( REP K_RANGE OF BYTE (' ')),
;  278        M 0937  1             ZEROES = UPLIT ( REP K_RANGE OF BYTE ('0'));
;  279        M 0938  1
;  280        M 0939  1         IF .E_D EQL 0
;  281        M 0940  1         THEN
;  282        M 0941  1             BEGIN
;  283        M 0942  1             M_D = MINUS;
;  284        M 0943  1             D_D = DOT;
;  285        M 0944  1             B_D = BLANK;
;  286        M 0945  1             E_D = E;
;  287        M 0946  1             END
;  288        M 0947  1
;  289          0948  1     %;
;  290          0949  1
;  291          0950  1 !<BLF/NOMACRO>
;  292          0951  1
;  293          0952  1 BUILTIN
;  294          0953  1     CVTLP,                                  ! Convert long to packed
;  295          0954  1     CVTPS,                                  ! Convert packed to leading sign
;  296          0955  1     CMPP;                                   ! Compare packed
;  297          0956  1
;  298          0957  1 !
;  299          0958  1 ! PSECT DECLARATIONS:
;  300          0959  1 !
;  301          0960  1 DECLARE_PSECTS (BAS);
;  302          0961  1 !
;  303          0962  1 ! EQUATED SYMBOLS:
;  304          0963  1 !
;  305          0964  1
;  306          0965  1 LITERAL
;  307          0966  1     V_RT_RND = 1^25,                        ! right round bit for kernel routine
;  308          0967  1     K_TMP_STR_LEN_G = 25,                   ! len. of gfloat string for kernel routine
;  309          0968  1     K_TMP_STR_LEN_H = 50,                   ! len. of hfloat string for kernel routine
;  310          0969  1     K_TMP_STR_LEN_D = 25,                   ! len. of double string for kernel output
;  311          0970  1     K_TMP_STR_LEN_F = 17,                   ! len. of float string for kernel output
;  312          0971  1     K_RANGE = 38,                           ! the exponent of the largest floating pt.
;  313          0972  1                                             ! number
;  314          0973  1     K_G_RANGE = 308,                        ! largest exp. for g floating
;  315          0974  1     K_H_RANGE = 4932,                       ! largest exp. for h floating
;  316          0975  1     K_P_RANGE = 4932,                       ! largest exp. for packed
;  317          0976  1     V_STRIP_SPACES = 1,                     ! bit flag - strip leading and trailing spaces
;  318          0977  1     V_TRAILING_SIGN = 2,                    ! bit flag - trailing sign
;  319          0978  1     V_COMMA = 4,                            ! bit flag - insert commas after every 3 digits
;  320          0979  1     V_CURRENCY = 8,                         ! bit flag - float the currency symbol
;  321          0980  1     V_STAR = 16,                            ! bit flag - asterisk fill to the left
;  322          0981  1     V_FLT_DEC_PT = 32,                      ! bit flag - float the decimal pt. in F format
;  323          0982  1     V_PERIOD = 64,                          ! bit flag - insert a period
```

```
;  324          0983  1     V_SUPPRESS_ZERO = 4096,                 ! bit flag - suppress zeroes
;  325          0984  1     V_LEADING_ZERO = 8192,                  ! bit flag - print leading zeroes
;  326          0985  1     V_CR_DR = 16384,                        ! bit flag - credit/debit
;  327          0986  1     V_ANSI_PRINT = 32768,                   ! bit flag - program is /ANSI
;  328          0987  1     K_NUM_F_SIG_DIG = 6,                    ! number of significant digits returned
;  329          0988  1                                             ! from a single precision conversion
;  330          0989  1     K_NUM_D_SIG_DIG = 16,                   ! number of significant digits returned
;  331          0990  1                                             ! from a double precision conversion
;  332          0991  1     K_NUM_G_SIG_DIG = 15,                   ! number of significant digits returned
;  333          0992  1                                             ! from a g float conversion
;  334          0993  1     K_NUM_H_SIG_DIG = 33,                   ! number of significant digits returned
;  335          0994  1                                             ! from an h float conversion
;  336          0995  1     K_NUM_P_SIG_DIG = 31,                   ! max number of significant digits
;  337          0996  1                                             ! in a packed decimal number
;  338          0997  1     K_KERNEL_BLK_SZ = 36;                   ! size in bytes of kernel routine parameter block
;  339          0998  1
;  340          0999  1 BIND
;  341          1000  1     PACKED_ZERO = UPLIT BYTE (REP 15 OF (%X'00'),%X'0C');
;  342          1001  1
;  343          1002  1 !
;  344          1003  1 ! OWN STORAGE:
;  345          1004  1 !
;  346          1005  1
;  347          1006  1 OWN
;  348          1007  1     DOT_DESC : INITIAL (%X'010E0001'),
;  349          1008  1     D_D : INITIAL (0),
;  350          1009  1     MINUS_DESC : INITIAL (%X'010E0001'),
;  351          1010  1     M_D : INITIAL (0),
;  352          1011  1     NULL_DESC : INITIAL (%X'010E0000'),
;  353          1012  1     N_D,
;  354          1013  1     BLANK_DESC : INITIAL (%X'010E0001'),
;  355          1014  1     B_D : INITIAL (0),
;  356          1015  1     E_DESC : INITIAL (%X'010E0001'),
;  357          1016  1     E_D : INITIAL (0);
;  358          1017  1
;  359          1018  1 !
;  360          1019  1 ! EXTERNAL REFERENCES:
;  361          1020  1 !
;  362          1021  1
;  363          1022  1 EXTERNAL ROUTINE
;  364          1023  1     LIB$STOP : NOVALUE,                     ! signal general errors
;  365          1024  1     BAS$$STOP : NOVALUE,                    ! signal errors and stop
;  366          1025  1     STR$CONCAT,                             ! String concatenate
;  367          1026  1     STR$FREE1_DX,                           ! dealloc dynamic string
;  368          1027  1     STR$DUPL_CHAR,                          ! create a string of chars
;  369          1028  1     OTS$$CVT_H_T_R8 : JSB_CVT_KERNEL,       ! OTS kernel conversion routines
;  370          1029  1     OTS$$CVT_G_T_R8 : JSB_CVT_KERNEL,       ! OTS kernel conversion routine
;  371          1030  1     OTS$$CVT_D_T_R8 : JSB_CVT_KERNEL;       ! OTS kernel conversion routine
;  372          1031  1
;  373          1032  1 EXTERNAL LITERAL
;  374          1033  1     BAS$K_PRIUSIFOR : UNSIGNED (8),         ! Print Using Format error
;  375          1034  1     OTS$_FATINTERR;                         ! Fatal internal RTL error
;  376          1035  1
```

```
;  378          1036  1 GLOBAL ROUTINE BAS$CVT_OUT_F_E (
;  379          1037  1         VALUE,
;  380          1038  1         NO_INT_DIGITS,
;  381          1039  1         NO_FRAC_DIGITS,
;  382          1040  1         FLAGS,
;  383          1041  1         BAS_OUT_STR_LEN,
;  384          1042  1         BAS_OUT_STR,
;  385          1043  1         CURRENCY,
;  386          1044  1         DIGIT_SEP,
;  387          1045  1         RADIX_PT
;  388          1046  1     ) =
;  389          1047  1
;  390          1048  1 !++
;  391          1049  1 ! FUNCTIONAL DESCRIPTION:
;  392          1050  1 !
;  393          1051  1 !       Write out a single precision number in explicit point scaled notation
;  394          1052  1 !       (E format).  This routine is called by the PRINT USING support of the
;  395          1053  1 !       Basic language.  It will return up to 6 significant digits.
;  396          1054  1 !
;  397          1055  1 ! FORMAL PARAMETERS:
;  398          1056  1 !
;  399          1057  1 !       VALUE.rf.r              the value to be converted
;  400          1058  1 !       NO_INT_DIGITS.rlu.r     number of integer digits in output string
;  401          1059  1 !       NO_FRAC_DIGITS.rlu.r    number of fraction digits in output string
;  402          1060  1 !       FLAGS.rlu.v             no flags are applicable
;  403          1061  1 !       BAS_OUT_STR_LEN.wlu.v   length of output string (handy for fixed length
;  404          1062  1 !                               output strings
;  405          1063  1 !       BAS_OUT_STR.wt.dx       the output string
;  406          1064  1 !       [CURRENCY.rt.dx]        currency symbol
;  407          1065  1 !       [DIGIT_SEP.rt.dx]       digit group separator
;  408          1066  1 !       [RADIX_PT.rt.dx]        radix point
;  409          1067  1 !
;  410          1068  1 ! IMPLICIT INPUTS:
;  411          1069  1 !
;  412          1070  1 !       NONE
;  413          1071  1 !
;  414          1072  1 ! IMPLICIT OUTPUTS:
;  415          1073  1 !
;  416          1074  1 !       NONE
;  417          1075  1 !
;  418          1076  1 ! COMPLETION CODES:
;  419          1077  1 !
;  420          1078  1 !       success         the value could be converted as required
;  421          1079  1 !       failure         the value did not fit in the field described
;  422          1080  1 !                       the return string was truncated.
;  423          1081  1 !
;  424          1082  1 ! SIDE EFFECTS:
;  425          1083  1 !
;  426          1084  1 !       NONE
;  427          1085  1 !
;  428          1086  1 !--
;  429          1087  1
;  430          1088  2     BEGIN
;  431          1089  2
;  432          1090  2     LITERAL
;  433          1091  2         K_ARG_COUNT = 9;
;  434          1092  2
```

```
;  435          1093  2     BUILTIN
;  436          1094  2         ACTUALCOUNT;
;  437          1095  2
;  438          1096  2     MAP
;  439          1097  2         RADIX_PT : REF BLOCK [, BYTE],
;  440          1098  2         CURRENCY : REF BLOCK [, BYTE],
;  441          1099  2         DIGIT_SEP : REF BLOCK [, BYTE],
;  442          1100  2         BAS_OUT_STR_LEN : REF VECTOR,
;  443          1101  2         VALUE : REF VECTOR,
;  444          1102  2         BAS_OUT_STR : REF VECTOR [2, LONG];
;  445          1103  2
;  446          1104  2     LOCAL
;  447          1105  2         STATUS;
;  448          1106  2
;  449          1107  2     IF ACTUALCOUNT () LSS K_ARG_COUNT THEN LIB$STOP (OTS$_FATINTERR);
;  450          1108  2
;  451          1109  2     STATUS = COMMON_E (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS, BAS_OUT_STR_LEN [0],
;  452          1110  2         BAS_OUT_STR [0], 0, .CURRENCY, .DIGIT_SEP, .RADIX_PT,
;  453          1111  2         DSC$K_DTYPE_F);
;  454          1112  2
;  455          1113  2     RETURN .STATUS;
;  456          1114  1     END;                                    !End of BAS$CVT_OUT_F_E


                                                    .TITLE  BAS$CVT_OUT
                                                    .IDENT  \1-054\

                                                    .PSECT  _BAS$DATA,NOEXE,  PIC,2

                                    010E0001  00000 DOT_DESC:
                                                    .LONG   17694721                        ;
                                    00000000  00004 D_D:    .LONG   0                       ;
                                    010E0001  00008 MINUS_DESC:
                                                    .LONG   17694721                        ;
                                    00000000  0000C M_D:    .LONG   0                       ;
                                    010E0000  00010 NULL_DESC:
                                                    .LONG   17694720                        ;
                                              00014 N_D:    .BLKB   4
                                    010E0001  00018 BLANK_DESC:
                                                    .LONG   17694721                        ;
                                    00000000  0001C B_D:    .LONG   0                       ;
                                    010E0001  00020 E_DESC: .LONG   17694721                ;
                                    00000000  00024 E_D:    .LONG   0                       ;

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

                                00#  00000 P.AAA:   .BYTE   0[15]                          ;
                                0C   0000F         .BYTE   12                            ;

                                           PACKED_ZERO=         P.AAA
                                                    .EXTRN  LIB$STOP, BAS$$STOP
                                                    .EXTRN  STR$CONCAT, STR$FREE1_DX
                                                    .EXTRN  STR$DUPL_CHAR, OTS$$CVT_H_T_R8
                                                    .EXTRN  OTS$$CVT_G_T_R8
                                                    .EXTRN  OTS$$CVT_D_T_R8
                                                    .EXTRN  BAS$K_PRIUSIFOR
                                                    .EXTRN  OTS$_FATINTERR
```

```
                                    0FFC 00000          .ENTRY  BAS$CVT_OUT_F_E, Save R2,R3,R4,R5,R6,R7,R8,-; 1036
                                                                R9,R10,R11
                          09            6C  91 00002          CMPB    (AP), #9                                ; 1107
                                        0D  1E 00005          BGEQU   1$
                             00000000G  8F  DD 00007          PUSHL   #OTS$_FATINTERR
                   00000000G 00         01  FB 0000D          CALLS   #1, LIB$STOP
                             5A         0A  D0 00014 1$:       MOVL    #10, R10                                ; 1110
                             58    20   AC  7D 00017          MOVQ    DIGIT_SEP, R8
                             57    1C   AC  D0 0001B          MOVL    CURRENCY, R7
                                        56  D4 0001F          CLRL    R6
                             54    14   AC  7D 00021          MOVQ    BAS_OUT_STR_LEN, R4
                             52    0C   AC  7D 00025          MOVQ    NO_FRAC_DIGITS, R2
                             50    04   AC  7D 00029          MOVQ    VALUE, R0
                                  0000V 30  0002D          BSBW    COMMON_E
                                        04  00030          RET                                             ; 1114

; Routine Size:  49 bytes,    Routine Base:  _BAS$CODE + 0010

;  457          1115  1
```

```
;  459          1116  1 GLOBAL ROUTINE BAS$CVT_OUT_D_E (
;  460          1117  1         VALUE,
;  461          1118  1         NO_INT_DIGITS,
;  462          1119  1         NO_FRAC_DIGITS,
;  463          1120  1         FLAGS,
;  464          1121  1         BAS_OUT_STR_LEN,
;  465          1122  1         BAS_OUT_STR,
;  466          1123  1         BAS_SCALE_FAC,
;  467          1124  1         CURRENCY,
;  468          1125  1         DIGIT_SEP,
;  469          1126  1         RADIX_PT
;  470          1127  1     ) =
;  471          1128  1
;  472          1129  1 !++
;  473          1130  1 ! FUNCTIONAL DESCRIPTION:
;  474          1131  1 !
;  475          1132  1 !       Write out a double precision number in explicit point scaled notation
;  476          1133  1 !       (E format).  This routine is called by the PRINT USING support of the
;  477          1134  1 !       Basic language.  It will return up to 16 significant digits.
;  478          1135  1 !
;  479          1136  1 ! FORMAL PARAMETERS:
;  480          1137  1 !
;  481          1138  1 !       VALUE.rd.r              the value to be converted
;  482          1139  1 !       NO_INT_DIGITS.rlu.r     number of integer digits in output string
;  483          1140  1 !       NO_FRAC_DIGITS.rlu.r    number of fraction digits in output string
;  484          1141  1 !       FLAGS.rlu.v             no flags are applicable
;  485          1142  1 !       BAS_OUT_STR_LEN.wlu.v   length of output string (handy for fixed length
;  486          1143  1 !                               output strings
;  487          1144  1 !       BAS_OUT_STR.wt.dx       the output string
;  488          1145  1 !       [BAS_SCALE_FAC.rb.v]    Basic scale factor in a range of 0 to -6
;  489          1146  1 !       [CURRENCY.rt.dx]        currency symbol
;  490          1147  1 !       [DIGIT_SEP.rt.dx]       digit group separator
;  491          1148  1 !       [RADIX_POINT.rt.dx]     radix point
;  492          1149  1 !
;  493          1150  1 ! IMPLICIT INPUTS:
;  494          1151  1 !
;  495          1152  1 !       NONE
;  496          1153  1 !
;  497          1154  1 ! IMPLICIT OUTPUTS:
;  498          1155  1 !
;  499          1156  1 !       NONE
;  500          1157  1 !
;  501          1158  1 ! COMPLETION CODES:
;  502          1159  1 !
;  503          1160  1 !       success         the value could be converted as required
;  504          1161  1 !       failure         the value did not fit in the field described
;  505          1162  1 !                       the return string was truncated
;  506          1163  1 !
;  507          1164  1 ! SIDE EFFECTS:
;  508          1165  1 !
;  509          1166  1 !       NONE
;  510          1167  1 !
;  511          1168  1 !--
;  512          1169  1
;  513          1170  2     BEGIN
;  514          1171  2
;  515          1172  2     MAP
```

```
;  516          1173  2          RADIX_PT : REF BLOCK [, BYTE],
;  517          1174  2          CURRENCY : REF BLOCK [, BYTE],
;  518          1175  2          DIGIT_SEP : REF BLOCK [, BYTE],
;  519          1176  2          BAS_SCALE_FAC : VECTOR [1, BYTE, SIGNED],
;  520          1177  2          BAS_OUT_STR_LEN : REF VECTOR,
;  521          1178  2          VALUE : REF VECTOR,
;  522          1179  2          BAS_OUT_STR : REF VECTOR [2, LONG];
;  523          1180  2
;  524          1181  2      LITERAL
;  525          1182  2          K_ARG_COUNT = 10;
;  526          1183  2
;  527          1184  2      BUILTIN
;  528          1185  2          ACTUALCOUNT;
;  529          1186  2
;  530          1187  2      LOCAL
;  531          1188  2          STATUS;
;  532          1189  2
;  533          1190  2      IF ACTUALCOUNT () LSS K_ARG_COUNT THEN LIB$STOP (OTS$_FATINTERR);
;  534          1191  2
;  535          1192  2      STATUS = COMMON_E (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS, BAS_OUT_STR_LEN [0],
;  536          1193  2          BAS_OUT_STR [0], .BAS_SCALE_FAC, .CURRENCY, .DIGIT_SEP, .RADIX_PT,
;  537          1194  2          DSC$K_DTYPE_D);
;  538          1195  2
;  539          1196  2      RETURN .STATUS;
;  540          1197  1      END;                                        ! End of BAS$CVT_OUT_D_E


                                 0FFC 00000        .ENTRY  BAS$CVT_OUT_D_E, Save R2,R3,R4,R5,R6,R7,R8,-; 1116
                                                           R9,R10,R11                                 ;
                  0A           6C  91 00002        CMPB    (AP), #10                                  ; 1190
                               0D  1E 00005        BGEQU   1$                                         ;
                     00000000G 8F  DD 00007        PUSHL   #OTS$_FATINTERR                            ;
         00000000G 00          01  FB 0000D        CALLS   #1, LIB$STOP                               ;
                  5A           0B  D0 00014 1$:    MOVL    #11, R10                                   ; 1193
                  58       24  AC  7D 00017        MOVQ    DIGIT_SEP, R8                              ;
                  56       1C  AC  7D 0001B        MOVQ    BAS_SCALE_FAC, R6                          ;
                  54       14  AC  7D 0001F        MOVQ    BAS_OUT_STR_LEN, R4                        ;
                  52       0C  AC  7D 00023        MOVQ    NO_FRAC_DIGITS, R2                         ;
                  50       04  AC  7D 00027        MOVQ    VALUE, R0                                  ;
                            0000V  30 0002B        BSBW    COMMON_E                                   ;
                                   04 0002E        RET                                                ; 1197

; Routine Size:  47 bytes,    Routine Base:  _BAS$CODE + 0041


;  541          1198  1
```

```
;  543          1199  1 GLOBAL ROUTINE BAS$CVT_OUT_F_F (                                      !
;  544          1200  1         VALUE,                                                        !
;  545          1201  1         NO_INT_DIGITS,                                                !
;  546          1202  1         NO_FRAC_DIGITS,                                               !
;  547          1203  1         FLAGS,                                                        !
;  548          1204  1         BAS_OUT_STR_LEN,                                              !
;  549          1205  1         BAS_OUT_STR,                                                  !
;  550          1206  1         CURRENCY,                                                     !
;  551          1207  1         DIGIT_SEP,                                                    !
;  552          1208  1         RADIX_PT                                                      !
;  553          1209  1     ) =                                                               !
;  554          1210  1
;  555          1211  1 !++
;  556          1212  1 ! FUNCTIONAL DESCRIPTION:
;  557          1213  1 !
;  558          1214  1 !       This routine converts a single precision number to explicit point un-
;  559          1215  1 !       scaled notation (F format) right justified.  This routine supports PRINT
;  560          1216  1 !       USING and NUM1$.  It will return up to 6 significant digits.
;  561          1217  1 !
;  562          1218  1 ! FORMAL PARAMETERS:
;  563          1219  1 !
;  564          1220  1 !       VALUE.rf.r              value to convert
;  565          1221  1 !       NO_INT_DIGITS.rlu.v     number of integer digits
;  566          1222  1 !       NO_FRAC_DIGITS.rlu.v    number of fraction digits
;  567          1223  1 !       FLAGS.rlu.v             = 1, no leading or trailing space
;  568          1224  1 !                               = 2, trailing sign
;  569          1225  1 !                               = 4, insert commas
;  570          1226  1 !                               = 8, float currency symbol
;  571          1227  1 !                               = 16, * fill to left of most significant digit
;  572          1228  1 !                               = 32, floating decimal point.  Disregard
;  573          1229  1 !                                     NO_INT_DIGITS and NO_FRAC_DIGITS.
;  574          1230  1 !                               = 64, a decimal point is required
;  575          1231  1 !       BAS_OUT_STR_LEN.wlu.r   length of return string
;  576          1232  1 !       BAS_OUT_STR.wt.dx       the return string
;  577          1233  1 !       [CURRENCY.rt.dx]        currency symbol
;  578          1234  1 !       [DIGIT_SEP.rt.dx]       digit group separator
;  579          1235  1 !       [RADIX_PT.rt.dx]        radix point
;  580          1236  1 !
;  581          1237  1 ! IMPLICIT INPUTS:
;  582          1238  1 !
;  583          1239  1 !       NONE
;  584          1240  1 !
;  585          1241  1 ! IMPLICIT OUTPUTS:
;  586          1242  1 !
;  587          1243  1 !       NONE
;  588          1244  1 !
;  589          1245  1 ! COMPLETION CODES:
;  590          1246  1 !
;  591          1247  1 !       success         the value could be converted as required
;  592          1248  1 !       failure         The value did not fit in the field described
;  593          1249  1 !                       the return string was truncated
;  594          1250  1 !
;  595          1251  1 ! SIDE EFFECTS:
;  596          1252  1 !
;  597          1253  1 !       NONE
;  598          1254  1 !
;  599          1255  1 !--
```

```
;  600          1256  1
;  601          1257  2      BEGIN
;  602          1258  2
;  603          1259  2      MAP
;  604          1260  2          RADIX_PT : REF BLOCK [, BYTE],
;  605          1261  2          CURRENCY : REF BLOCK [, BYTE],
;  606          1262  2          DIGIT_SEP : REF BLOCK [, BYTE],
;  607          1263  2          BAS_OUT_STR_LEN : REF VECTOR,
;  608          1264  2          VALUE : REF VECTOR,
;  609          1265  2          BAS_OUT_STR : REF VECTOR [2, LONG];
;  610          1266  2
;  611          1267  2      BUILTIN
;  612          1268  2          ACTUALCOUNT;
;  613          1269  2
;  614          1270  2      LOCAL
;  615          1271  2          STATUS;
;  616          1272  2
;  617          1273  2      LITERAL
;  618          1274  2          K_ARG_COUNT = 9;
;  619          1275  2
;  620          1276  2      IF ACTUALCOUNT () NEQ K_ARG_COUNT
;  621          1277  2      THEN
;  622          1278  2          STATUS = COMMON_F (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS,
;  623          1279  2                  BAS_OUT_STR_LEN [0], BAS_OUT_STR [0], 0, DSC$K_DTYPE_F, 9)
;  624          1280  2      ELSE
;  625          1281  2          STATUS = COMMON_F (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS, BAS_OUT_STR_LEN [0],
;  626          1282  2                  BAS_OUT_STR [0], 0, DSC$K_DTYPE_F, 12, .CURRENCY, .DIGIT_SEP,
;  627          1283  2                   .RADIX_PT);
;  628          1284  2
;  629          1285  2      RETURN .STATUS;
;  630          1286  1      END;                                    !End of BAS$CVT_OUT_F_F


                              0FFC 00000          .ENTRY  BAS$CVT_OUT_F_F, Save R2,R3,R4,R5,R6,R7,R8,-;  1199
                                                          R9,R10,R11
                    5E            04  C2 00002          SUBL2   #4, SP
                    09            6C  91 00005          CMPB    (AP), #9                                1276
                                  05  13 00008          BEQL    1$
                    58            09  D0 0000A          MOVL    #9, R8                                  1279
                                  0B  11 0000D          BRB     2$
                    5A      20    AC  7D 0000F  1$:      MOVQ    DIGIT_SEP, R10                          1282
                    59      1C    AC  D0 00013          MOVL    CURRENCY, R9
                    58            0C  D0 00017          MOVL    #12, R8
                    57            0A  D0 0001A  2$:      MOVL    #10, R7
                                  56  D4 0001D          CLRL    R6
                    54      14    AC  7D 0001F          MOVQ    BAS_OUT_STR_LEN, R4
                    52      0C    AC  7D 00023          MOVQ    NO_FRAC_DIGITS, R2
                    50      04    AC  7D 00027          MOVQ    VALUE, R0
                                0000V 30 0002B          BSBW    COMMON_F
                    6E            50  D0 0002E          MOVL    R0, STATUS
                                      04 00031          RET                                             1286

; Routine Size:  50 bytes,    Routine Base:  _BAS$CODE + 0070
```

; 631 1287 1

```
;  633          1288  1 GLOBAL ROUTINE BAS$CVT_OUT_D_F (
;  634          1289  1         VALUE,
;  635          1290  1         NO_INT_DIGITS,
;  636          1291  1         NO_FRAC_DIGITS,
;  637          1292  1         FLAGS,
;  638          1293  1         BAS_OUT_STR_LEN,
;  639          1294  1         BAS_OUT_STR,
;  640          1295  1         BAS_SCALE_FAC,
;  641          1296  1         CURRENCY,
;  642          1297  1         DIGIT_SEP,
;  643          1298  1         RADIX_PT
;  644          1299  1     ) =
;  645          1300  1
;  646          1301  1 !++
;  647          1302  1 ! FUNCTIONAL DESCRIPTION:
;  648          1303  1 !
;  649          1304  1 !       This routine converts a double precision number to explicit point un-
;  650          1305  1 !       scaled notation (F format) right justified.  This routine supports PRINT
;  651          1306  1 !       USING and NUM1$.  It will return up to 16 significant digits.
;  652          1307  1 !
;  653          1308  1 ! FORMAL PARAMETERS:
;  654          1309  1 !
;  655          1310  1 !       VALUE.rd.r              value to convert
;  656          1311  1 !       NO_INT_DIGITS.r.lu.v    number of integer digits
;  657          1312  1 !       NO_FRAC_DIGITS.rlu.v    number of fraction digits
;  658          1313  1 !       FLAGS.rlu.v             = 1, no leading or trailing space
;  659          1314  1 !                               = 2, trailing sign
;  660          1315  1 !                               = 4, insert commas
;  661          1316  1 !                               = 8, float currency symbol
;  662          1317  1 !                               = 16, * fill to left of most significant digit
;  663          1318  1 !                               = 32, floating decimal point.  Disregard
;  664          1319  1 !                                  NO_INT_DIGITS and NO_FRAC_DIGITS.
;  665          1320  1 !                               = 64, a decimal point is required
;  666          1321  1 !       BAS_OUT_STR_LEN.wlu.r   length of return string
;  667          1322  1 !       BAS_OUT_STR.wt.dx       the return string
;  668          1323  1 !       [BAS_SCALE_FAC.rb.v]    Basic scale factor from 0 to -6
;  669          1324  1 !       [CURRENCY.rt.dx]        currency symbol
;  670          1325  1 !       [DIGIT_SEP.rt.dx]       digit group separator
;  671          1326  1 !       [RADIX_PT.rt.dx]        radix point
;  672          1327  1 !
;  673          1328  1 !
;  674          1329  1 ! IMPLICIT INPUTS:
;  675          1330  1 !
;  676          1331  1 !       NONE
;  677          1332  1 !
;  678          1333  1 ! IMPLICIT OUTPUTS:
;  679          1334  1 !
;  680          1335  1 !       NONE
;  681          1336  1 !
;  682          1337  1 ! COMPLETION CODES:
;  683          1338  1 !
;  684          1339  1 !       success         the value could be converted as required
;  685          1340  1 !       failure         the value did not fit in the field described
;  686          1341  1 !
;  687          1342  1 ! SIDE EFFECTS:
;  688          1343  1 !
;  689          1344  1 !       NONE
```

```
;  690          1345  1 !
;  691          1346  1 !--
;  692          1347  1
;  693          1348  2     BEGIN
;  694          1349  2
;  695          1350  2     MAP
;  696          1351  2         RADIX_PT : REF BLOCK [, BYTE],
;  697          1352  2         BAS_SCALE_FAC : VECTOR [1, BYTE, SIGNED],
;  698          1353  2         CURRENCY : REF BLOCK [, BYTE],
;  699          1354  2         DIGIT_SEP : REF BLOCK [, BYTE],
;  700          1355  2         BAS_OUT_STR_LEN : REF VECTOR,
;  701          1356  2         VALUE : REF VECTOR,
;  702          1357  2         BAS_OUT_STR : REF VECTOR [2, LONG];
;  703          1358  2
;  704          1359  2     BUILTIN
;  705          1360  2         ACTUALCOUNT;
;  706          1361  2
;  707          1362  2     LOCAL
;  708          1363  2         STATUS;
;  709          1364  2
;  710          1365  2     LITERAL
;  711          1366  2         K_SCALE = 7,
;  712          1367  2         K_ARG_COUNT = 10;
;  713          1368  2
;  714          1369  2     IF ACTUALCOUNT () LSS K_SCALE THEN BAS_SCALE_FAC = 0;
;  715          1370  2
;  716          1371  2     IF ACTUALCOUNT () NEQ K_ARG_COUNT
;  717          1372  2     THEN
;  718          1373  2         STATUS = COMMON_F (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS,
;  719          1374  2                  BAS_OUT_STR_LEN [0], BAS_OUT_STR [0], .BAS_SCALE_FAC, DSC$K_DTYPE_D,
;  720          1375  2                  9)
;  721          1376  2     ELSE
;  722          1377  2         STATUS = COMMON_F (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS, BAS_OUT_STR_LEN [0],
;  723          1378  2                  BAS_OUT_STR [0], .BAS_SCALE_FAC, DSC$K_DTYPE_D, 12, .CURRENCY,
;  724          1379  2                  .DIGIT_SEP, .RADIX_PT);
;  725          1380  2
;  726          1381  2     RETURN .STATUS;
;  727          1382  1     END;                                    !End of BAS$CVT_OUT_D_F


                        0FFC 00000          .ENTRY  BAS$CVT_OUT_D_F, Save R2,R3,R4,R5,R6,R7,R8,-  ; 1288
                                                    R9,R10,R11                                    ;
5E                  04  C2 00002            SUBL2   #4, SP                                        ;
07                  6C  91 00005            CMPB    (AP), #7                                      ; 1369
                    03  1E 00008            BGEQU   1$                                            ;
                1C  AC  D4 0000A            CLRL    BAS_SCALE_FAC                                 ;
0A                  6C  91 0000D 1$:        CMPB    (AP), #10                                     ; 1371
                    05  13 00010            BEQL    2$                                            ;
58                  09  D0 00012            MOVL    #9, R8                                        ; 1374
                    0B  11 00015            BRB     3$                                            ;
5A              24  AC  7D 00017 2$:        MOVQ    DIGIT_SEP, R10                                ; 1378
59              20  AC  D0 0001B            MOVL    CURRENCY, R9                                  ;
58                  0C  D0 0001F            MOVL    #12, R8                                       ;
57                  0B  D0 00022 3$:        MOVL    #11, R7                                       ;
```

```
                                55       18    AC  7D 00025          MOVQ    BAS_OUT_STR, R5                                 ;
                                53       10    AC  7D 00029          MOVQ    FLAGS, R3                                       ;
                                51       08    AC  7D 0002D          MOVQ    NO_INT_DIGITS, R1                               ;
                                50       04    AC  D0 00031          MOVL    VALUE, R0                                       ;
                                                 0000V 30 00035          BSBW    COMMON_F                                        ;
                                6E                 50  D0 00038          MOVL    R0, STATUS                                      ;
                                                       04 0003B          RET                                                     ; 1382

; Routine Size:  60 bytes,    Routine Base:  _BAS$CODE + 00A2

;  728          1383  1
```

```
;  730          1384  1 GLOBAL ROUTINE BAS$CVT_OUT_D_G (
;  731          1385  1         VALUE,
;  732          1386  1         FLAGS,
;  733          1387  1         BAS_OUT_STR_LEN,
;  734          1388  1         BAS_OUT_STR,
;  735          1389  1         BAS_SCALE_FAC,
;  736          1390  1         NUM_DIGITS
;  737          1391  1     ) =
;  738          1392  1
;  739          1393  1 !++
;  740          1394  1 ! FUNCTIONAL DESCRIPTION:
;  741          1395  1 !
;  742          1396  1 !       This routine converts a double precision number to either E or F format
;  743          1397  1 !       depending on its value.  This routine supports PRINT, PRINT USING, NUM$,
;  744          1398  1 !       and STR$.  If a number can be represented in six digits without any loss
;  745          1399  1 !       of accuracy then it is returned in F format.  Otherwise, it is returned
;  746          1400  1 !       in E format.  If the optional argument NUM_DIGITS is included, then the
;  747          1401  1 !       number is returned in F format if it can be represented no less accur-
;  748          1402  1 !       ately in NUM_DIGITS.
;  749          1403  1 !
;  750          1404  1 ! FORMAL PARAMETERS:
;  751          1405  1 !
;  752          1406  1 !       VALUE.rd.r              the value to convert
;  753          1407  1 !       FLAGS.rlu.v             = 1, strip leading and trailing space
;  754          1408  1 !       BAS_OUT_STR_LEN.ww.r    length of the string returned.  Useful if return
;  755          1409  1 !                               string is static.
;  756          1410  1 !       BAS_OUT_STR.wt.dx       return string
;  757          1411  1 !       [BAS_SCALE_FAC.rb.v]    the scale factor for double precision values.
;  758          1412  1 !       [NUM_DIGITS.rl.v]       number of digits in F format.
;  759          1413  1 !
;  760          1414  1 ! IMPLICIT INPUTS:
;  761          1415  1 !
;  762          1416  1 !       NONE
;  763          1417  1 !
;  764          1418  1 ! IMPLICIT OUTPUTS:
;  765          1419  1 !
;  766          1420  1 !       NONE
;  767          1421  1 !
;  768          1422  1 ! COMPLETION CODES:
;  769          1423  1 !
;  770          1424  1 !       success         the value could be converted as required
;  771          1425  1 !       failure         the value did not fit in the field described
;  772          1426  1 !
;  773          1427  1 ! SIDE EFFECTS:
;  774          1428  1 !
;  775          1429  1 !       NONE
;  776          1430  1 !
;  777          1431  1 !--
;  778          1432  1
;  779          1433  2     BEGIN
;  780          1434  2
;  781          1435  2     MAP
;  782          1436  2         BAS_SCALE_FAC : VECTOR [1, BYTE, SIGNED],
;  783          1437  2         BAS_OUT_STR_LEN : REF VECTOR,
;  784          1438  2         VALUE : REF VECTOR,
;  785          1439  2         BAS_OUT_STR : REF VECTOR [2, LONG];
;  786          1440  2
```

```
;  787          1441  2     LITERAL
;  788          1442  2         K_NUM_DIGITS = 6,                       ! Arg. position of number of digits
;  789          1443  2         K_SCALE = 5;                            ! Arg. position of scale factor
;  790          1444  2
;  791          1445  2     LOCAL
;  792          1446  2         SIG_DIG;
;  793          1447  2
;  794          1448  2     BUILTIN
;  795          1449  2         ACTUALCOUNT;
;  796          1450  2
;  797          1451  2     LOCAL
;  798          1452  2         STATUS;
;  799          1453  2
;  800          1454  2     IF ACTUALCOUNT () LSS K_SCALE
;  801          1455  2     THEN
;  802          1456  3         BEGIN
;  803          1457  3         BAS_SCALE_FAC = 0;
;  804          1458  3         SIG_DIG = K_NUM_F_SIG_DIG;
;  805          1459  2         END;
;  806          1460  2
;  807          1461  2     IF ACTUALCOUNT () NEQ K_NUM_DIGITS
;  808          1462  2     THEN
;  809          1463  2         SIG_DIG  = K_NUM_F_SIG_DIG
;  810          1464  2     ELSE
;  811          1465  2         SIG_DIG = .NUM_DIGITS;
;  812          1466  2
;  813          1467  2     STATUS = COMMON_G (.VALUE, .FLAGS, BAS_OUT_STR_LEN [0], BAS_OUT_STR [0],
;  814          1468  2         .BAS_SCALE_FAC, .SIG_DIG, DSC$K_DTYPE_D);
;  815          1469  2
;  816          1470  2     RETURN .STATUS;
;  817          1471  2
;  818          1472  1     END;                                        ! end of BAS$CVT_OUT_D_G

                                  0FFC 00000          .ENTRY  BAS$CVT_OUT_D_G, Save R2,R3,R4,R5,R6,R7,R8,-  ; 1384
                                                              R9,R10,R11
                            05        6C  91 00002          CMPB    (AP), #5                                  ; 1454
                                      06  1E 00005          BGEQU   1$
                                14    AC  D4 00007          CLRL    BAS_SCALE_FAC                             ; 1457
                            55        06  D0 0000A          MOVL    #6, SIG_DIG                               ; 1458
                            06        6C  91 0000D 1$:      CMPB    (AP), #6                                  ; 1461
                                      05  13 00010          BEQL    2$
                            55        06  D0 00012          MOVL    #6, SIG_DIG                               ; 1463
                                      04  11 00015          BRB     3$
                            55    18  AC  D0 00017 2$:      MOVL    NUM_DIGITS, SIG_DIG                       ; 1465
                            56        0B  D0 0001B 3$:      MOVL    #11, R6                                   ; 1467
                            53    10  AC  7D 0001E          MOVQ    BAS_OUT_STR, R3
                            51    08  AC  7D 00022          MOVQ    FLAGS, R1
                            50    04  AC  D0 00026          MOVL    VALUE, R0
                                0000V     30 0002A          BSBW    COMMON_G
                                          04 0002D          RET                                               ; 1472

; Routine Size:  46 bytes,    Routine Base:  _BAS$CODE + 00DE
```

```
;  820          1473  1 GLOBAL ROUTINE BAS$CVT_OUT_G_E (
;  821          1474  1         VALUE,
;  822          1475  1         NO_INT_DIGITS,
;  823          1476  1         NO_FRAC_DIGITS,
;  824          1477  1         FLAGS,
;  825          1478  1         BAS_OUT_STR_LEN,
;  826          1479  1         BAS_OUT_STR,
;  827          1480  1         CURRENCY,
;  828          1481  1         DIGIT_SEP,
;  829          1482  1         RADIX_PT
;  830          1483  1     ) =
;  831          1484  1
;  832          1485  1 !++
;  833          1486  1 ! FUNCTIONAL DESCRIPTION:
;  834          1487  1 !
;  835          1488  1 !       Write out a G floating number in explicit point scaled notation
;  836          1489  1 !       (E format).  This routine is called by the PRINT USING support of the
;  837          1490  1 !       Basic language.  It will return up to 15 significant digits.
;  838          1491  1 !
;  839          1492  1 ! FORMAL PARAMETERS:
;  840          1493  1 !
;  841          1494  1 !       VALUE.rd.r              the value to be converted
;  842          1495  1 !       NO_INT_DIGITS.rlu.r     number of integer digits in output string
;  843          1496  1 !       NO_FRAC_DIGITS.rlu.r    number of fraction digits in output string
;  844          1497  1 !       FLAGS.rlu.v             no flags are applicable
;  845          1498  1 !       BAS_OUT_STR_LEN.wlu.v   length of output string (handy for fixed length
;  846          1499  1 !                               output strings
;  847          1500  1 !       BAS_OUT_STR.wt.dx       the output string
;  848          1501  1 !       [CURRENCY.rt.dx]        currency symbol
;  849          1502  1 !       [DIGIT_SEP.rt.dx]       digit group separator
;  850          1503  1 !       [RADIX_POINT.rt.dx]     radix point
;  851          1504  1 !
;  852          1505  1 ! IMPLICIT INPUTS:
;  853          1506  1 !
;  854          1507  1 !       NONE
;  855          1508  1 !
;  856          1509  1 ! IMPLICIT OUTPUTS:
;  857          1510  1 !
;  858          1511  1 !       NONE
;  859          1512  1 !
;  860          1513  1 ! COMPLETION CODES:
;  861          1514  1 !
;  862          1515  1 !       success         the value could be converted as required
;  863          1516  1 !       failure         the value did not fit in the field described
;  864          1517  1 !                       the return string was truncated
;  865          1518  1 !
;  866          1519  1 ! SIDE EFFECTS:
;  867          1520  1 !
;  868          1521  1 !       NONE
;  869          1522  1 !
;  870          1523  1 !--
;  871          1524  1
;  872          1525  2     BEGIN
;  873          1526  2
;  874          1527  2     MAP
;  875          1528  2         RADIX_PT : REF BLOCK [, BYTE],
;  876          1529  2         CURRENCY : REF BLOCK [, BYTE],
```

```
;  877          1530  2          DIGIT_SEP : REF BLOCK [, BYTE],
;  878          1531  2          BAS_OUT_STR_LEN : REF VECTOR,
;  879          1532  2          VALUE : REF VECTOR,
;  880          1533  2          BAS_OUT_STR : REF VECTOR [2, LONG];
;  881          1534  2
;  882          1535  2      LITERAL
;  883          1536  2          K_ARG_COUNT = 9;
;  884          1537  2
;  885          1538  2      BUILTIN
;  886          1539  2          ACTUALCOUNT;
;  887          1540  2
;  888          1541  2      LOCAL
;  889          1542  2          STATUS;
;  890          1543  2
;  891          1544  2      IF ACTUALCOUNT () LSS K_ARG_COUNT THEN LIB$STOP (OTS$_FATINTERR);
;  892          1545  2
;  893          1546  2      STATUS = COMMON_E (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS, BAS_OUT_STR_LEN [0],
;  894          1547  2          BAS_OUT_STR [0], 0, .CURRENCY, .DIGIT_SEP, .RADIX_PT, DSC$K_DTYPE_G);
;  895          1548  2
;  896          1549  2      RETURN .STATUS;
;  897          1550  1      END;                                    !End of BAS$CVT_OUT_G_E


                                 0FFC 00000          .ENTRY  BAS$CVT_OUT_G_E, Save R2,R3,R4,R5,R6,R7,R8,-  ; 1473
                                                             R9,R10,R11                                        ;
                      09          6C  91 00002       CMPB    (AP), #9                                          ; 1544
                                  0D  1E 00005       BGEQU   1$                                                ;
                          00000000G   8F  DD 00007       PUSHL   #OTS$_FATINTERR                               ;
                 00000000G    00      01  FB 0000D       CALLS   #1, LIB$STOP                                  ;
                              5A      1B  D0 00014 1$:   MOVL    #27, R10                                       ; 1547
                              58  20  AC  7D 00017       MOVQ    DIGIT_SEP, R8                                  ;
                              57  1C  AC  D0 0001B       MOVL    CURRENCY, R7                                   ;
                                      56  D4 0001F       CLRL    R6                                             ;
                              54  14  AC  7D 00021       MOVQ    BAS_OUT_STR_LEN, R4                            ;
                              52  0C  AC  7D 00025       MOVQ    NO_FRAC_DIGITS, R2                             ;
                              50  04  AC  7D 00029       MOVQ    VALUE, R0                                      ;
                                  0000V   30  0002D      BSBW    COMMON_E                                       ;
                                          04  00030      RET                                                    ; 1550

; Routine Size:  49 bytes,    Routine Base:  _BAS$CODE + 010C

;  898          1551  1
;  899          1552  1
```

```
;  901          1553  1 GLOBAL ROUTINE BAS$CVT_OUT_G_F (
;  902          1554  1         VALUE,
;  903          1555  1         NO_INT_DIGITS,
;  904          1556  1         NO_FRAC_DIGITS,
;  905          1557  1         FLAGS,
;  906          1558  1         BAS_OUT_STR_LEN,
;  907          1559  1         BAS_OUT_STR,
;  908          1560  1         CURRENCY,
;  909          1561  1         DIGIT_SEP,
;  910          1562  1         RADIX_PT
;  911          1563  1     ) =
;  912          1564  1
;  913          1565  1 !++
;  914          1566  1 ! FUNCTIONAL DESCRIPTION:
;  915          1567  1 !
;  916          1568  1 !       This routine converts a G floating number to explicit point un-
;  917          1569  1 !       scaled notation (F format) right justified.  This routine supports PRINT
;  918          1570  1 !       USING and NUM1$.  It will return up to 15 significant digits.
;  919          1571  1 !
;  920          1572  1 ! FORMAL PARAMETERS:
;  921          1573  1 !
;  922          1574  1 !       VALUE.rd.r              value to convert
;  923          1575  1 !       NO_INT_DIGITS.rlu.v     number of integer digits
;  924          1576  1 !       NO_FRAC_DIGITS.rlu.v    number of fraction digits
;  925          1577  1 !       FLAGS.rlu.v             = 1, no leading or trailing space
;  926          1578  1 !                               = 2, trailing sign
;  927          1579  1 !                               = 4, insert commas
;  928          1580  1 !                               = 8, float currency symbol
;  929          1581  1 !                               = 16, * fill to left of most significant digit
;  930          1582  1 !                               = 32, floating decimal point.  Disregard
;  931          1583  1 !                                     NO_INT_DIGITS and NO_FRAC_DIGITS.
;  932          1584  1 !                               = 64, a decimal point is required
;  933          1585  1 !       BAS_OUT_STR_LEN.wlu.r   length of return string
;  934          1586  1 !       BAS_OUT_STR.wt.dx       the return string
;  935          1587  1 !       [CURRENCY.rt.dx]        currency symbol
;  936          1588  1 !       [DIGIT_SEP.rt.dx]       digit group separator
;  937          1589  1 !       [RADIX_PT.rt.dx]        radix point
;  938          1590  1 !
;  939          1591  1 !
;  940          1592  1 ! IMPLICIT INPUTS:
;  941          1593  1 !
;  942          1594  1 !       NONE
;  943          1595  1 !
;  944          1596  1 ! IMPLICIT OUTPUTS:
;  945          1597  1 !
;  946          1598  1 !       NONE
;  947          1599  1 !
;  948          1600  1 ! COMPLETION CODES:
;  949          1601  1 !
;  950          1602  1 !       success         the value could be converted as required
;  951          1603  1 !       failure         the value did not fit in the field described
;  952          1604  1 !
;  953          1605  1 ! SIDE EFFECTS:
;  954          1606  1 !
;  955          1607  1 !       NONE
;  956          1608  1 !
;  957          1609  1 !--
```

```
;  958          1610  1
;  959          1611  2      BEGIN
;  960          1612  2
;  961          1613  2      MAP
;  962          1614  2          RADIX_PT : REF BLOCK [, BYTE],
;  963          1615  2          CURRENCY : REF BLOCK [, BYTE],
;  964          1616  2          DIGIT_SEP : REF BLOCK [, BYTE],
;  965          1617  2          BAS_OUT_STR_LEN : REF VECTOR,
;  966          1618  2          VALUE : REF VECTOR,
;  967          1619  2          BAS_OUT_STR : REF VECTOR [2, LONG];
;  968          1620  2
;  969          1621  2      BUILTIN
;  970          1622  2          ACTUALCOUNT;
;  971          1623  2
;  972          1624  2      LOCAL
;  973          1625  2          STATUS;
;  974          1626  2
;  975          1627  2      LITERAL
;  976          1628  2          K_ARG_COUNT = 9;
;  977          1629  2
;  978          1630  2      IF ACTUALCOUNT () NEQ K_ARG_COUNT
;  979          1631  2      THEN
;  980          1632  2          STATUS = COMMON_F (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS,
;  981          1633  2                  BAS_OUT_STR_LEN [0], BAS_OUT_STR [0], 0, DSC$K_DTYPE_G, 9)
;  982          1634  2      ELSE
;  983          1635  2          STATUS = COMMON_F (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS, BAS_OUT_STR_LEN [0],
;  984          1636  2                  BAS_OUT_STR [0], 0, DSC$K_DTYPE_G, 12, .CURRENCY, .DIGIT_SEP,
;  985          1637  2                  .RADIX_PT);
;  986          1638  2
;  987          1639  2      RETURN .STATUS;
;  988          1640  1      END;                                    !End of BAS$CVT_OUT_G_F


                                    0FFC 00000          .ENTRY  BAS$CVT_OUT_G_F, Save R2,R3,R4,R5,R6,R7,R8,-  ; 1553
                                                                R9,R10,R11
                                5E          04  C2 00002        SUBL2   #4, SP                                  ;
                                09          6C  91 00005        CMPB    (AP), #9                                ; 1630
                                            05  13 00008        BEQL    1$                                      ;
                                58          09  D0 0000A        MOVL    #9, R8                                  ; 1633
                                            0B  11 0000D        BRB     2$                                      ;
                                5A    20    AC  7D 0000F  1$:   MOVQ    DIGIT_SEP, R10                          ; 1636
                                59    1C    AC  D0 00013        MOVL    CURRENCY, R9                            ;
                                58          0C  D0 00017        MOVL    #12, R8                                 ;
                                57          1B  D0 0001A  2$:   MOVL    #27, R7                                 ;
                                            56  D4 0001D        CLRL    R6                                      ;
                                54    14    AC  7D 0001F        MOVQ    BAS_OUT_STR_LEN, R4                     ;
                                52    0C    AC  7D 00023        MOVQ    NO_FRAC_DIGITS, R2                      ;
                                50    04    AC  7D 00027        MOVQ    VALUE, R0                               ;
                                      0000V 30  00002B        BSBW    COMMON_F                                ;
                                6E          50  D0 0002E        MOVL    R0, STATUS                              ;
                                                04 00031        RET                                             ; 1640

; Routine Size:  50 bytes,    Routine Base:  _BAS$CODE + 013D
```

; 989   1641 1

```
;  991          1642  1 GLOBAL ROUTINE BAS$CVT_OUT_H_E (
;  992          1643  1          VALUE,
;  993          1644  1          NO_INT_DIGITS,
;  994          1645  1          NO_FRAC_DIGITS,
;  995          1646  1          FLAGS,
;  996          1647  1          BAS_OUT_STR_LEN,
;  997          1648  1          BAS_OUT_STR,
;  998          1649  1          CURRENCY,
;  999          1650  1          DIGIT_SEP,
; 1000          1651  1          RADIX_PT
; 1001          1652  1      ) =
; 1002          1653  1
; 1003          1654  1 !++
; 1004          1655  1 ! FUNCTIONAL DESCRIPTION:
; 1005          1656  1 !
; 1006          1657  1 !       Write out an H floating number in explicit point scaled notation
; 1007          1658  1 !       (E format).  This routine is called by the PRINT USING support of the
; 1008          1659  1 !       Basic language.  It will return up to 33 significant digits.
; 1009          1660  1 !
; 1010          1661  1 ! FORMAL PARAMETERS:
; 1011          1662  1 !
; 1012          1663  1 !       VALUE.rd.r              the value to be converted
; 1013          1664  1 !       NO_INT_DIGITS.rlu.r     number of integer digits in output string
; 1014          1665  1 !       NO_FRAC_DIGITS.rlu.r    number of fraction digits in output string
; 1015          1666  1 !       FLAGS.rlu.v             no flags are applicable
; 1016          1667  1 !       BAS_OUT_STR_LEN.wlu.v   length of output string (handy for fixed length
; 1017          1668  1 !                               output strings
; 1018          1669  1 !       BAS_OUT_STR.wt.dx       the output string
; 1019          1670  1 !       [CURRENCY.rt.dx]        currency symbol
; 1020          1671  1 !       [DIGIT_SEP.rt.dx]       digit group separator
; 1021          1672  1 !       [RADIX_POINT.rt.dx]     radix point
; 1022          1673  1 !
; 1023          1674  1 ! IMPLICIT INPUTS:
; 1024          1675  1 !
; 1025          1676  1 !       NONE
; 1026          1677  1 !
; 1027          1678  1 ! IMPLICIT OUTPUTS:
; 1028          1679  1 !
; 1029          1680  1 !       NONE
; 1030          1681  1 !
; 1031          1682  1 ! COMPLETION CODES:
; 1032          1683  1 !
; 1033          1684  1 !       success         the value could be converted as required
; 1034          1685  1 !       failure         the value did not fit in the field described
; 1035          1686  1 !                       the return string was truncated
; 1036          1687  1 !
; 1037          1688  1 ! SIDE EFFECTS:
; 1038          1689  1 !
; 1039          1690  1 !       NONE
; 1040          1691  1 !
; 1041          1692  1 !--
; 1042          1693  1
; 1043          1694  2     BEGIN
; 1044          1695  2
; 1045          1696  2     MAP
; 1046          1697  2         RADIX_PT : REF BLOCK [, BYTE],
; 1047          1698  2         CURRENCY : REF BLOCK [, BYTE],
```

```
;  1048          1699  2          DIGIT_SEP : REF BLOCK [, BYTE],
;  1049          1700  2          BAS_OUT_STR_LEN : REF VECTOR,
;  1050          1701  2          VALUE : REF VECTOR,
;  1051          1702  2          BAS_OUT_STR : REF VECTOR [2, LONG];
;  1052          1703  2
;  1053          1704  2      BUILTIN
;  1054          1705  2          ACTUALCOUNT;
;  1055          1706  2
;  1056          1707  2      LOCAL
;  1057          1708  2          STATUS;
;  1058          1709  2
;  1059          1710  2      LITERAL
;  1060          1711  2          K_ARG_COUNT = 9;
;  1061          1712  2
;  1062          1713  2      IF ACTUALCOUNT () LSS K_ARG_COUNT THEN LIB$STOP (OTS$_FATINTERR);
;  1063          1714  2
;  1064          1715  2      STATUS = COMMON_E (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS, BAS_OUT_STR_LEN [0],
;  1065          1716  2          BAS_OUT_STR [0], 0, .CURRENCY, .DIGIT_SEP, .RADIX_PT, DSC$K_DTYPE_H,
;  1066          1717  2          ACTUALCOUNT ());
;  1067          1718  2
;  1068          1719  2      RETURN .STATUS;
;  1069          1720  1      END;                                    !End of BAS$CVT_OUT_H_E


                                        0FFC 00000          .ENTRY  BAS$CVT_OUT_H_E, Save R2,R3,R4,R5,R6,R7,R8,-  ; 1642
                                                                    R9,R10,R11                                        ;
                              09        6C  91 00002        CMPB    (AP), #9                                          ; 1713
                                        0D  1E 00005        BGEQU   1$                                                ;
                    00000000G           8F  DD 00007        PUSHL   #OTS$_FATINTERR                                   ;
          00000000G   00                01  FB 0000D        CALLS   #1, LIB$STOP                                      ;
                      7E                6C  9A 00014 1$:    MOVZBL  (AP), -(SP)                                       ; 1717
                      5A                1C  D0 00017        MOVL    #28, R10                                          ;
                      58        20      AC  7D 0001A        MOVQ    DIGIT_SEP, R8                                     ;
                      57        1C      AC  D0 0001E        MOVL    CURRENCY, R7                                      ;
                                        56  D4 00022        CLRL    R6                                                ;
                      54        14      AC  7D 00024        MOVQ    BAS_OUT_STR_LEN, R4                               ;
                      52        0C      AC  7D 00028        MOVQ    NO_FRAC_DIGITS, R2                                ;
                      50        04      AC  7D 0002C        MOVQ    VALUE, R0                                         ;
                                  0000V 30  00030           BSBW    COMMON_E                                          ;
                                        04  00033           RET                                                       ; 1720

; Routine Size:  52 bytes,    Routine Base:  _BAS$CODE + 016F
```

```
;  1071          1721  1 GLOBAL ROUTINE BAS$CVT_OUT_G_G (
;  1072          1722  1         VALUE,
;  1073          1723  1         FLAGS,
;  1074          1724  1         BAS_OUT_STR_LEN,
;  1075          1725  1         BAS_OUT_STR,
;  1076          1726  1         NUM_DIGITS
;  1077          1727  1     ) =
;  1078          1728  1
;  1079          1729  1 !++
;  1080          1730  1 ! FUNCTIONAL DESCRIPTION:
;  1081          1731  1 !
;  1082          1732  1 !       This routine converts a G floating number to either E or F format
;  1083          1733  1 !       depending on its value.  This routine supports PRINT, PRINT USING, NUM$,
;  1084          1734  1 !       and STR$.  If a number can be represented in six digits without any loss
;  1085          1735  1 !       of accuracy then it is returned in F format.  Otherwise, it is returned
;  1086          1736  1 !       in E format.  If the optional argument NUM_DIGITS is included, then the
;  1087          1737  1 !       number is returned in F format if it can be represented no less accur-
;  1088          1738  1 !       ately in NUM_DIGITS.
;  1089          1739  1 !
;  1090          1740  1 ! FORMAL PARAMETERS:
;  1091          1741  1 !
;  1092          1742  1 !       VALUE.rd.r              the value to convert
;  1093          1743  1 !       FLAGS.rlu.v             = 1, strip leading and trailing space
;  1094          1744  1 !       BAS_OUT_STR_LEN.ww.r    length of the string returned.  Useful if return
;  1095          1745  1 !                               string is static.
;  1096          1746  1 !       BAS_OUT_STR.wt.dx       return string
;  1097          1747  1 !       [NUM_DIGITS.rl.v]       number of digits in F format.
;  1098          1748  1 !
;  1099          1749  1 ! IMPLICIT INPUTS:
;  1100          1750  1 !
;  1101          1751  1 !       NONE
;  1102          1752  1 !
;  1103          1753  1 ! IMPLICIT OUTPUTS:
;  1104          1754  1 !
;  1105          1755  1 !       NONE
;  1106          1756  1 !
;  1107          1757  1 ! COMPLETION CODES:
;  1108          1758  1 !
;  1109          1759  1 !       success         the value could be converted as required
;  1110          1760  1 !       failure         the value did not fit in the field described
;  1111          1761  1 !
;  1112          1762  1 ! SIDE EFFECTS:
;  1113          1763  1 !
;  1114          1764  1 !       NONE
;  1115          1765  1 !
;  1116          1766  1 !--
;  1117          1767  1
;  1118          1768  2     BEGIN
;  1119          1769  2
;  1120          1770  2     MAP
;  1121          1771  2         BAS_OUT_STR_LEN : REF VECTOR,
;  1122          1772  2         VALUE : REF VECTOR,
;  1123          1773  2         BAS_OUT_STR : REF VECTOR [2, LONG];
;  1124          1774  2
;  1125          1775  2     LITERAL
;  1126          1776  2         K_NUM_DIGITS = 5;                       ! Arg. position of number of digits
;  1127          1777  2
```

```
;  1128          1778  2     BUILTIN
;  1129          1779  2         ACTUALCOUNT;
;  1130          1780  2
;  1131          1781  2     LOCAL
;  1132          1782  2         STATUS,
;  1133          1783  2         SIG_DIG;
;  1134          1784  2
;  1135          1785  2     IF ACTUALCOUNT () NEQ K_NUM_DIGITS
;  1136          1786  2     THEN
;  1137          1787  2         SIG_DIG = K_NUM_F_SIG_DIG
;  1138          1788  2     ELSE
;  1139          1789  2         SIG_DIG = .NUM_DIGITS;
;  1140          1790  2
;  1141          1791  2     STATUS = COMMON_G (.VALUE, .FLAGS, BAS_OUT_STR_LEN [0], BAS_OUT_STR [0],
;  1142          1792  2         0, .SIG_DIG, DSC$K_DTYPE_G);
;  1143          1793  2
;  1144          1794  2     RETURN .STATUS;
;  1145          1795  2
;  1146          1796  1     END;                                    !End of BAS$CVT_OUT_G_G


                      0FFC 00000          .ENTRY  BAS$CVT_OUT_G_G, Save R2,R3,R4,R5,R6,R7,R8,-  ; 1721
                                                  R9,R10,R11                                    ;
                   6C  91 00002           CMPB    (AP), #5                                     ; 1785
                   05  13 00005           BEQL    1$                                           ;
                   06  D0 00007           MOVL    #6, SIG_DIG                                  ; 1787
                   04  11 0000A           BRB     2$                                           ;
                   AC  D0 0000C 1$:       MOVL    NUM_DIGITS, SIG_DIG                          ; 1789
                   1B  D0 00010 2$:       MOVL    #27, R6                                      ; 1791
                   54  D4 00013           CLRL    R4                                           ;
                0C AC  7D 00015           MOVQ    BAS_OUT_STR_LEN, R2                          ;
                04 AC  7D 00019           MOVQ    VALUE, R0                                    ;
                 0000V 30 0001D           BSBW    COMMON_G                                     ;
                       04 00020           RET                                                  ; 1796

; Routine Size:  33 bytes,    Routine Base:  _BAS$CODE + 01A3
```

```
;1148     1797  1 GLOBAL ROUTINE BAS$CVT_OUT_H_G (                                      !
;1149     1798  1         VALUE,                                                        !
;1150     1799  1         FLAGS,                                                        !
;1151     1800  1         BAS_OUT_STR_LEN,                                              !
;1152     1801  1         BAS_OUT_STR,                                                  !
;1153     1802  1         NUM_DIGITS                                                    !
;1154     1803  1     ) =                                                               !
;1155     1804  1
;1156     1805  1 !++
;1157     1806  1 ! FUNCTIONAL DESCRIPTION:
;1158     1807  1 !
;1159     1808  1 !       This routine converts an H floating number to either E or F format
;1160     1809  1 !       depending on its value.  This routine supports PRINT, PRINT USING, NUM$,
;1161     1810  1 !       and STR$.  If a number can be represented in six digits without any loss
;1162     1811  1 !       of accuracy then it is returned in F format.  Otherwise, it is returned
;1163     1812  1 !       in E format.  If the optional argument NUM_DIGITS is included, then the
;1164     1813  1 !       number is returned in F format if it can be represented no less accur-
;1165     1814  1 !       ately in NUM_DIGITS.
;1166     1815  1 !
;1167     1816  1 ! FORMAL PARAMETERS:
;1168     1817  1 !
;1169     1818  1 !       VALUE.rd.r              the value to convert
;1170     1819  1 !       FLAGS.rlu.v             = 1, strip leading and trailing space
;1171     1820  1 !       BAS_OUT_STR_LEN.ww.r    length of the string returned.  Useful if return
;1172     1821  1 !                               string is static.
;1173     1822  1 !       BAS_OUT_STR.wt.dx       return string
;1174     1823  1 !       [NUM_DIGITS.rl.v]       number of digits
;1175     1824  1 !
;1176     1825  1 ! IMPLICIT INPUTS:
;1177     1826  1 !
;1178     1827  1 !       NONE
;1179     1828  1 !
;1180     1829  1 ! IMPLICIT OUTPUTS:
;1181     1830  1 !
;1182     1831  1 !       NONE
;1183     1832  1 !
;1184     1833  1 ! COMPLETION CODES:
;1185     1834  1 !
;1186     1835  1 !       success         the value could be converted as required
;1187     1836  1 !       failure         the value did not fit in the field described
;1188     1837  1 !
;1189     1838  1 ! SIDE EFFECTS:
;1190     1839  1 !
;1191     1840  1 !       NONE
;1192     1841  1 !
;1193     1842  1 !--
;1194     1843  1
;1195     1844  2     BEGIN
;1196     1845  2
;1197     1846  2     MAP
;1198     1847  2         BAS_OUT_STR_LEN : REF VECTOR,
;1199     1848  2         VALUE : REF VECTOR,
;1200     1849  2         BAS_OUT_STR : REF VECTOR [2, LONG];
;1201     1850  2
;1202     1851  2     LITERAL
;1203     1852  2         K_NUM_DIGITS = 5,                           ! Arg. position of number of digits
;1204     1853  2         K_NUM_H_E_DIGS = 5;                         ! Only 5 digits in fraction for
```

```
;  1205          1854  2                                        ! h floating (6 for other data types)
;  1206          1855  2
;  1207          1856  2     BUILTIN
;  1208          1857  2         ACTUALCOUNT;
;  1209          1858  2
;  1210          1859  2     LOCAL
;  1211          1860  2         STATUS,
;  1212          1861  2         SIG_DIG;
;  1213          1862  2
;  1214          1863  2     IF ACTUALCOUNT () NEQ K_NUM_DIGITS
;  1215          1864  2     THEN
;  1216          1865  2         SIG_DIG = K_NUM_F_SIG_DIG
;  1217          1866  2     ELSE
;  1218          1867  2         SIG_DIG = .NUM_DIGITS;
;  1219          1868  2
;  1220          1869  2     STATUS = COMMON_G (.VALUE, .FLAGS, BAS_OUT_STR_LEN [0], BAS_OUT_STR [0],
;  1221          1870  2         0, .SIG_DIG, DSC$K_DTYPE_H);
;  1222          1871  2
;  1223          1872  2     RETURN .STATUS;
;  1224          1873  2
;  1225          1874  1     END;                                 !End of BAS$CVT_OUT_H_G


                              0FFC 00000          .ENTRY  BAS$CVT_OUT_H_G, Save R2,R3,R4,R5,R6,R7,R8,-; 1797
                                                          R9,R10,R11
                        05    6C  91 00002          CMPB    (AP), #5                                    ; 1863
                              05  13 00005          BEQL    1$
                        55    06  D0 00007          MOVL    #6, SIG_DIG                                 ; 1865
                              04  11 0000A          BRB     2$
                        55  14  AC  D0 0000C  1$:    MOVL    NUM_DIGITS, SIG_DIG                         ; 1867
                        56    1C  D0 00010  2$:     MOVL    #28, R6                                     ; 1869
                              54  D4 00013          CLRL    R4
                        52  0C  AC  7D 00015          MOVQ    BAS_OUT_STR_LEN, R2
                        50  04  AC  7D 00019          MOVQ    VALUE, R0
                            0000V 30 0001D          BSBW    COMMON_G
                                  04 00020          RET                                                 ; 1874

; Routine Size:  33 bytes,    Routine Base:  _BAS$CODE + 01C4
```

```
;1227     1875  1 GLOBAL ROUTINE BAS$CVT_OUT_H_F (
;1228     1876  1         VALUE,
;1229     1877  1         NO_INT_DIGITS,
;1230     1878  1         NO_FRAC_DIGITS,
;1231     1879  1         FLAGS,
;1232     1880  1         BAS_OUT_STR_LEN,
;1233     1881  1         BAS_OUT_STR,
;1234     1882  1         CURRENCY,
;1235     1883  1         DIGIT_SEP,
;1236     1884  1         RADIX_PT
;1237     1885  1     ) =
;1238     1886  1
;1239     1887  1 !++
;1240     1888  1 ! FUNCTIONAL DESCRIPTION:
;1241     1889  1 !
;1242     1890  1 !       This routine converts an H floating number to explicit point un-
;1243     1891  1 !       scaled notation (F format) right justified.  This routine supports PRINT
;1244     1892  1 !       USING and NUM1$.  It will return up to 33 significant digits.
;1245     1893  1 !
;1246     1894  1 ! FORMAL PARAMETERS:
;1247     1895  1 !
;1248     1896  1 !       VALUE.rd.r              value to convert
;1249     1897  1 !       NO_INT_DIGITS.rlu.v     number of integer digits
;1250     1898  1 !       NO_FRAC_DIGITS.rlu.v    number of fraction digits
;1251     1899  1 !       FLAGS.rlu.v             = 1, no leading or trailing space
;1252     1900  1 !                               = 2, trailing sign
;1253     1901  1 !                               = 4, insert commas
;1254     1902  1 !                               = 8, float currency symbol
;1255     1903  1 !                               = 16, * fill to left of most significant digit
;1256     1904  1 !                               = 32, floating decimal point.  Disregard
;1257     1905  1 !                                     NO_INT_DIGITS and NO_FRAC_DIGITS.
;1258     1906  1 !                               = 64, a decimal point is required
;1259     1907  1 !       BAS_OUT_STR_LEN.wlu.r   length of return string
;1260     1908  1 !       BAS_OUT_STR.wt.dx       the return string
;1261     1909  1 !       [CURRENCY.rt.dx]        currency symbol
;1262     1910  1 !       [DIGIT_SEP.rt.dx]       digit group separator
;1263     1911  1 !       [RADIX_PT.rt.dx]        radix point
;1264     1912  1 !
;1265     1913  1 !
;1266     1914  1 ! IMPLICIT INPUTS:
;1267     1915  1 !
;1268     1916  1 !       NONE
;1269     1917  1 !
;1270     1918  1 ! IMPLICIT OUTPUTS:
;1271     1919  1 !
;1272     1920  1 !       NONE
;1273     1921  1 !
;1274     1922  1 ! COMPLETION CODES:
;1275     1923  1 !
;1276     1924  1 !       success         the value could be converted as required
;1277     1925  1 !       failure         the value did not fit in the field described
;1278     1926  1 !
;1279     1927  1 ! SIDE EFFECTS:
;1280     1928  1 !
;1281     1929  1 !       NONE
;1282     1930  1 !
;1283     1931  1 !--
```

```
;  1284          1932  1
;  1285          1933  2      BEGIN
;  1286          1934  2
;  1287          1935  2      MAP
;  1288          1936  2          RADIX_PT : REF BLOCK [, BYTE],
;  1289          1937  2          CURRENCY : REF BLOCK [, BYTE],
;  1290          1938  2          DIGIT_SEP : REF BLOCK [, BYTE],
;  1291          1939  2          BAS_OUT_STR_LEN : REF VECTOR,
;  1292          1940  2          VALUE : REF VECTOR,
;  1293          1941  2          BAS_OUT_STR : REF VECTOR [2, LONG];
;  1294          1942  2
;  1295          1943  2      BUILTIN
;  1296          1944  2          ACTUALCOUNT;
;  1297          1945  2
;  1298          1946  2      LOCAL
;  1299          1947  2          STATUS;
;  1300          1948  2
;  1301          1949  2      LITERAL
;  1302          1950  2          K_ARG_COUNT = 9;
;  1303          1951  2
;  1304          1952  2      IF ACTUALCOUNT () NEQ K_ARG_COUNT
;  1305          1953  2      THEN
;  1306          1954  2          STATUS = COMMON_F (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS,
;  1307          1955  2                  BAS_OUT_STR_LEN [0], BAS_OUT_STR [0], 0, DSC$K_DTYPE_H, 9)
;  1308          1956  2      ELSE
;  1309          1957  2          STATUS = COMMON_F (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS, BAS_OUT_STR_LEN [0],
;  1310          1958  2                  BAS_OUT_STR [0], 0, DSC$K_DTYPE_H, 12, .CURRENCY, .DIGIT_SEP,
;  1311          1959  2                   .RADIX_PT);
;  1312          1960  2
;  1313          1961  2      RETURN .STATUS;
;  1314          1962  1      END;                                    !End of BAS$CVT_OUT_H_F

                              0FFC 00000          .ENTRY  BAS$CVT_OUT_H_F, Save R2,R3,R4,R5,R6,R7,R8,-  ; 1875
                                                          R9,R10,R11
                                                                                                        ;
                   5E            04  C2 00002           SUBL2   #4, SP                                 ;
                   09            6C  91 00005           CMPB    (AP), #9                               ; 1952
                                 05  13 00008           BEQL    1$                                     ;
                   58            09  D0 0000A           MOVL    #9, R8                                 ; 1955
                                 0B  11 0000D           BRB     2$                                     ;
                   5A      20    AC  7D 0000F  1$:      MOVQ    DIGIT_SEP, R10                         ; 1958
                   59      1C    AC  D0 00013           MOVL    CURRENCY, R9                           ;
                   58            0C  D0 00017           MOVL    #12, R8                                ;
                   57            1C  D0 0001A  2$:      MOVL    #28, R7                                ;
                                 56  D4 0001D           CLRL    R6                                     ;
                   54      14    AC  7D 0001F           MOVQ    BAS_OUT_STR_LEN, R4                    ;
                   52      0C    AC  7D 00023           MOVQ    NO_FRAC_DIGITS, R2                     ;
                   50      04    AC  7D 00027           MOVQ    VALUE, R0                              ;
                              0000V 30 0002B           BSBW    COMMON_F                               ;
                   6E            50  D0 0002E           MOVL    R0, STATUS                             ;
                                     04 00031           RET                                            ; 1962

; Routine Size:  50 bytes,    Routine Base:  _BAS$CODE + 01E5
```

```
;1316     1963  1 GLOBAL ROUTINE BAS$CVT_OUT_P_F (                                  !
;1317     1964  1         VALUE,                                                    !
;1318     1965  1         NO_INT_DIGITS,                                            !
;1319     1966  1         NO_FRAC_DIGITS,                                           !
;1320     1967  1         FLAGS,                                                    !
;1321     1968  1         BAS_OUT_STR_LEN,                                          !
;1322     1969  1         BAS_OUT_STR,                                              !
;1323     1970  1         CURRENCY,                                                 !
;1324     1971  1         DIGIT_SEP,                                                !
;1325     1972  1         RADIX_PT                                                  !
;1326     1973  1     ) =                                                           !
;1327     1974  1
;1328     1975  1 !++
;1329     1976  1 ! FUNCTIONAL DESCRIPTION:
;1330     1977  1 !
;1331     1978  1 !       This routine converts a packed decimal number to explicit point un-
;1332     1979  1 !       scaled notation (F format) right justified.  This routine supports PRINT
;1333     1980  1 !       USING and NUM1$.  It will return up to 31 significant digits.
;1334     1981  1 !
;1335     1982  1 ! FORMAL PARAMETERS:
;1336     1983  1 !
;1337     1984  1 !       VALUE.rt.dx             packed decimal string to convert
;1338     1985  1 !       NO_INT_DIGITS.rlu.v     number of integer digits
;1339     1986  1 !       NO_FRAC_DIGITS.rlu.v    number of fraction digits
;1340     1987  1 !       FLAGS.rlu.v             = 1, no leading or trailing space
;1341     1988  1 !                               = 2, trailing sign
;1342     1989  1 !                               = 4, insert commas
;1343     1990  1 !                               = 8, float currency symbol
;1344     1991  1 !                               = 16, * fill to left of most significant digit
;1345     1992  1 !                               = 32, floating decimal point.  Disregard
;1346     1993  1 !                                     NO_INT_DIGITS and NO_FRAC_DIGITS.
;1347     1994  1 !                               = 64, a decimal point is required
;1348     1995  1 !       BAS_OUT_STR_LEN.wlu.r   length of return string
;1349     1996  1 !       BAS_OUT_STR.wt.dx       the return string
;1350     1997  1 !       [CURRENCY.rt.dx]        currency symbol
;1351     1998  1 !       [DIGIT_SEP.rt.dx]       digit group separator
;1352     1999  1 !       [RADIX_PT.rt.dx]        radix point
;1353     2000  1 !
;1354     2001  1 ! IMPLICIT INPUTS:
;1355     2002  1 !
;1356     2003  1 !       NONE
;1357     2004  1 !
;1358     2005  1 ! IMPLICIT OUTPUTS:
;1359     2006  1 !
;1360     2007  1 !       NONE
;1361     2008  1 !
;1362     2009  1 ! COMPLETION CODES:
;1363     2010  1 !
;1364     2011  1 !       success         the value could be converted as required
;1365     2012  1 !       failure         The value did not fit in the field described
;1366     2013  1 !                       the return string was truncated
;1367     2014  1 !
;1368     2015  1 ! SIDE EFFECTS:
;1369     2016  1 !
;1370     2017  1 !       NONE
;1371     2018  1 !
;1372     2019  1 !--
```

```
;  1373          2020  1
;  1374          2021  2      BEGIN
;  1375          2022  2
;  1376          2023  2      MAP
;  1377          2024  2          RADIX_PT : REF BLOCK [, BYTE],
;  1378          2025  2          CURRENCY : REF BLOCK [, BYTE],
;  1379          2026  2          DIGIT_SEP : REF BLOCK [, BYTE],
;  1380          2027  2          BAS_OUT_STR_LEN : REF VECTOR,
;  1381          2028  2          VALUE : REF VECTOR,
;  1382          2029  2          BAS_OUT_STR : REF VECTOR [2, LONG];
;  1383          2030  2
;  1384          2031  2      BUILTIN
;  1385          2032  2          ACTUALCOUNT;
;  1386          2033  2
;  1387          2034  2      LOCAL
;  1388          2035  2          STATUS;
;  1389          2036  2
;  1390          2037  2      LITERAL
;  1391          2038  2          K_ARG_COUNT = 9;
;  1392          2039  2
;  1393          2040  2      IF ACTUALCOUNT () NEQ K_ARG_COUNT
;  1394          2041  2      THEN
;  1395          2042  2          STATUS = COMMON_F (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS,
;  1396          2043  2                  BAS_OUT_STR_LEN [0], BAS_OUT_STR [0], 0, DSC$K_DTYPE_P, 9)
;  1397          2044  2      ELSE
;  1398          2045  2          STATUS = COMMON_F (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS, BAS_OUT_STR_LEN [0],
;  1399          2046  2                  BAS_OUT_STR [0], 0, DSC$K_DTYPE_P, 12, .CURRENCY, .DIGIT_SEP,
;  1400          2047  2                   .RADIX_PT);
;  1401          2048  2
;  1402          2049  2      RETURN .STATUS;
;  1403          2050  1      END;                                    !End of BAS$CVT_OUT_P_F


                                        0FFC 00000          .ENTRY  BAS$CVT_OUT_P_F, Save R2,R3,R4,R5,R6,R7,R8,-  ; 1963
                                                                    R9,R10,R11
                                    5E      04  C2 00002            SUBL2   #4, SP
                                    09      6C  91 00005            CMPB    (AP), #9                                  ; 2040
                                            05  13 00008            BEQL    1$
                                    58      09  D0 0000A            MOVL    #9, R8                                    ; 2043
                                            0B  11 0000D            BRB     2$
                                    5A  20  AC  7D 0000F 1$:        MOVQ    DIGIT_SEP, R10                            ; 2046
                                    59  1C  AC  D0 00013            MOVL    CURRENCY, R9
                                    58      0C  D0 00017            MOVL    #12, R8
                                    57      15  D0 0001A 2$:        MOVL    #21, R7
                                            56  D4 0001D            CLRL    R6
                                    54  14  AC  7D 0001F            MOVQ    BAS_OUT_STR_LEN, R4
                                    52  0C  AC  7D 00023            MOVQ    NO_FRAC_DIGITS, R2
                                    50  04  AC  7D 00027            MOVQ    VALUE, R0
                                        0000V 30 0002B            BSBW    COMMON_F
                                    6E      50  D0 0002E            MOVL    R0, STATUS
                                                04 00031            RET                                               ; 2050

; Routine Size:  50 bytes,    Routine Base:  _BAS$CODE + 0217
```

; 1404 2051 1

```
;1406     2052  1 GLOBAL ROUTINE BAS$CVT_OUT_P_E (
;1407     2053  1         VALUE,
;1408     2054  1         NO_INT_DIGITS,
;1409     2055  1         NO_FRAC_DIGITS,
;1410     2056  1         FLAGS,
;1411     2057  1         BAS_OUT_STR_LEN,
;1412     2058  1         BAS_OUT_STR,
;1413     2059  1         CURRENCY,
;1414     2060  1         DIGIT_SEP,
;1415     2061  1         RADIX_PT
;1416     2062  1     ) =
;1417     2063  1
;1418     2064  1 !++
;1419     2065  1 ! FUNCTIONAL DESCRIPTION:
;1420     2066  1 !
;1421     2067  1 !       Write out a packed decimal number in explicit point scaled notation
;1422     2068  1 !       (E format).  This routine is called by the PRINT USING support of the
;1423     2069  1 !       Basic language.  It will return up to 6 significant digits.
;1424     2070  1 !
;1425     2071  1 ! FORMAL PARAMETERS:
;1426     2072  1 !
;1427     2073  1 !       VALUE.rt.dx             the packed decimal string to be converted
;1428     2074  1 !       NO_INT_DIGITS.rlu.r     number of integer digits in output string
;1429     2075  1 !       NO_FRAC_DIGITS.rlu.r    number of fraction digits in output string
;1430     2076  1 !       FLAGS.rlu.v             no flags are applicable
;1431     2077  1 !       BAS_OUT_STR_LEN.wlu.v   length of output string (handy for fixed length
;1432     2078  1 !                               output strings
;1433     2079  1 !       BAS_OUT_STR.wt.dx       the output string
;1434     2080  1 !       [CURRENCY.rt.dx]        currency symbol
;1435     2081  1 !       [DIGIT_SEP.rt.dx]       digit group separator
;1436     2082  1 !       [RADIX_PT.rt.dx]        radix point
;1437     2083  1 !
;1438     2084  1 ! IMPLICIT INPUTS:
;1439     2085  1 !
;1440     2086  1 !       NONE
;1441     2087  1 !
;1442     2088  1 ! IMPLICIT OUTPUTS:
;1443     2089  1 !
;1444     2090  1 !       NONE
;1445     2091  1 !
;1446     2092  1 ! COMPLETION CODES:
;1447     2093  1 !
;1448     2094  1 !       success         the value could be converted as required
;1449     2095  1 !       failure         the value did not fit in the field described
;1450     2096  1 !                       the return string was truncated.
;1451     2097  1 !
;1452     2098  1 ! SIDE EFFECTS:
;1453     2099  1 !
;1454     2100  1 !       NONE
;1455     2101  1 !
;1456     2102  1 !--
;1457     2103  1
;1458     2104  2     BEGIN
;1459     2105  2
;1460     2106  2     LITERAL
;1461     2107  2         K_ARG_COUNT = 9;
;1462     2108  2
```

```
;  1463          2109  2     BUILTIN
;  1464          2110  2         ACTUALCOUNT;
;  1465          2111  2
;  1466          2112  2     MAP
;  1467          2113  2         RADIX_PT : REF BLOCK [, BYTE],
;  1468          2114  2         CURRENCY : REF BLOCK [, BYTE],
;  1469          2115  2         DIGIT_SEP : REF BLOCK [, BYTE],
;  1470          2116  2         BAS_OUT_STR_LEN : REF VECTOR,
;  1471          2117  2         VALUE : REF VECTOR,
;  1472          2118  2         BAS_OUT_STR : REF VECTOR [2, LONG];
;  1473          2119  2
;  1474          2120  2     LOCAL
;  1475          2121  2         STATUS;
;  1476          2122  2
;  1477          2123  2     IF ACTUALCOUNT () LSS K_ARG_COUNT THEN LIB$STOP (OTS$_FATINTERR);
;  1478          2124  2
;  1479          2125  2     STATUS = COMMON_E (.VALUE, .NO_INT_DIGITS, .NO_FRAC_DIGITS, .FLAGS, BAS_OUT_STR_LEN [0],
;  1480          2126  2         BAS_OUT_STR [0], 0, .CURRENCY, .DIGIT_SEP, .RADIX_PT,
;  1481          2127  2         DSC$K_DTYPE_P);
;  1482          2128  2
;  1483          2129  2     RETURN .STATUS;
;  1484          2130  1     END;                                    .End of BAS$CVT_OUT_P_E

                                    0FFC 00000          .ENTRY  BAS$CVT_OUT_P_E, Save R2,R3,R4,R5,R6,R7,R8,-; 2052
                                                                R9,R10,R11
                          09        6C  91 00002          CMPB    (AP), #9                                      ; 2123
                                    0D  1E 00005          BGEQU   1$
                          0000000CG 8F  DD 00007          PUSHL   #OTS$_FATINTERR
                00000000G 00        01  FB 0000D          CALLS   #1, LIB$STOP
                          5A            15  D0 00014 1$:  MOVL    #21, R10                                      ; 2126
                          58      20    AC  7D 00017          MOVQ    DIGIT_SEP, R8
                          57      1C    AC  D0 0001B          MOVL    CURRENCY, R7
                                        56  D4 0001F          CLRL    R6
                          54      14    AC  7D 00021          MOVQ    BAS_OUT_STR_LEN, R4
                          52      0C    AC  7D 00025          MOVQ    NO_FRAC_DIGITS, R2
                          50      04    AC  7D 00029          MOVQ    VALUE, R0
                                  0000V 30  0002D          BSBW    COMMON_E
                                        04  00030          RET                                               ; 2130

; Routine Size:  49 bytes,    Routine Base:  _BAS$CODE + 0249

;  1485          2131  1
```

```
;  1487          2132  1 GLOBAL ROUTINE BAS$CVT_OUT_P_G (
;  1488          2133  1         VALUE,
;  1489          2134  1         FLAGS,
;  1490          2135  1         BAS_OUT_STR_LEN,
;  1491          2136  1         BAS_OUT_STR,
;  1492          2137  1         NUM_DIGITS
;  1493          2138  1     ) =
;  1494          2139  1
;  1495          2140  1 !++
;  1496          2141  1 ! FUNCTIONAL DESCRIPTION:
;  1497          2142  1 !
;  1498          2143  1 !       This routine converts a packed decimal number to F format.  (Packed decimal
;  1499          2144  1 !       will never print in E format since it is defined to work the same as text.
;  1500          2145  1 !       This routine supports PRINT, PRINT USING, NUM$, and STR$.
;  1501          2146  1 !
;  1502          2147  1 ! FORMAL PARAMETERS:
;  1503          2148  1 !
;  1504          2149  1 !       VALUE.rt.dx             the value to convert
;  1505          2150  1 !       FLAGS.rlu.v             = 1, strip leading and trailing space
;  1506          2151  1 !       BAS_OUT_STR_LEN.ww.r    length of the string returned.  Useful if return
;  1507          2152  1 !                               string is static.
;  1508          2153  1 !       BAS_OUT_STR.wt.dx       return string
;  1509          2154  1 !       [NUM_DIGITS.rl.v]       number of digits in F format.
;  1510          2155  1 !
;  1511          2156  1 ! IMPLICIT INPUTS:
;  1512          2157  1 !
;  1513          2158  1 !       NONE
;  1514          2159  1 !
;  1515          2160  1 ! IMPLICIT OUTPUTS:
;  1516          2161  1 !
;  1517          2162  1 !       NONE
;  1518          2163  1 !
;  1519          2164  1 ! COMPLETION CODES:
;  1520          2165  1 !
;  1521          2166  1 !       success         the va  e could be converted as required
;  1522          2167  1 !       failure         the value did not fit in the field described
;  1523          2168  1 !
;  1524          2169  1 ! SIDE EFFECTS:
;  1525          2170  1 !
;  1526          2171  1 !       NONE
;  1527          2172  1 !
;  1528          2173  1 !--
;  1529          2174  1
;  1530          2175  2     BEGIN
;  1531          2176  2
;  1532          2177  2     MAP
;  1533          2178  2         BAS_OUT_STR_LEN : REF VECTOR,
;  1534          2179  2         VALUE : REF VECTOR,
;  1535          2180  2         BAS_OUT_STR : REF VECTOR [2, LONG];
;  1536          2181  2
;  1537          2182  2     LITERAL
;  1538          2183  2         K_NUM_DIGITS = 5;                       ! Arg. position of number of digits
;  1539          2184  2
;  1540          2185  2     BUILTIN
;  1541          2186  2         ACTUALCOUNT;
;  1542          2187  2
;  1543          2188  2     LOCAL
```

```
;  1544          2189  2          STATUS,
;  1545          2190  2          DIGITS;
;  1546          2191  2
;  1547          2192  2      IF ACTUALCOUNT () NEQ K_NUM_DIGITS
;  1548          2193  2      THEN
;  1549          2194  2          DIGITS = K_NUM_P_SIG_DIG
;  1550          2195  2      ELSE
;  1551          2196  2          DIGITS = .NUM_DIGITS;
;  1552          2197  2
;  1553          2198  2      STATUS = COMMON_G (.VALUE, .FLAGS, BAS_OUT_STR_LEN [0], BAS_OUT_STR [0],
;  1554          2199  2          0, .DIGITS, DSC$K_DTYPE_P);
;  1555          2200  2
;  1556          2201  2      RETURN .STATUS;
;  1557          2202  2
;  1558          2203  1      END;                                    ! end of BAS$CVT_OUT_P_G


                                    0FFC 00000          .ENTRY  BAS$CVT_OUT_P_G, Save R2,R3,R4,R5,R6,R7,R8,-; 2132
                                                                R9,R10,R11
                              05        6C  91 00002      CMPB    (AP), #5                                      ; 2192
                                        05  13 00005      BEQL    1$
                              55        1F  D0 00007      MOVL    #31, DIGITS                                   ; 2194
                                        04  11 0000A      BRB     2$
                              55    14  AC  D0 0000C 1$:  MOVL    NUM_DIGITS, DIGITS                            ; 2196
                              56        15  D0 00010 2$:  MOVL    #21, R6                                       ; 2198
                                        54  D4 00013      CLRL    R4
                              52    0C  AC  7D 00015      MOVQ    BAS_OUT_STR_LEN, R2
                              50    04  AC  7D 00019      MOVQ    VALUE, R0
                                    0000V 30 0001D      BSBW    COMMON_G
                                          04 00020      RET                                                   ; 2203

; Routine Size:  33 bytes,    Routine Base:  _BAS$CODE + 027A
```

```
;1560     2204  1 ROUTINE COMMON_E (
;1561     2205  1          VALUE,
;1562     2206  1          NO_INT_DIGITS,
;1563     2207  1          NO_FRAC_DIGITS,
;1564     2208  1          FLAGS,
;1565     2209  1          BAS_OUT_STR_LEN,
;1566     2210  1          BAS_OUT_STR,
;1567     2211  1          BAS_SCALE_FAC,
;1568     2212  1          CURRENCY,
;1569     2213  1          DIGIT_SEP,
;1570     2214  1          RADIX_PT,
;1571     2215  1          DATA_TYPE
;1572     2216  1      ): JSB_E_FORMAT =
;1573     2217  1
;1574     2218  1 !++
;1575     2219  1 ! FUNCTIONAL DESCRIPTION:
;1576     2220  1 !
;1577     2221  1 !       Write out a number in explicit point scaled notation
;1578     2222  1 !       (E format).  This routine is called by the PRINT USING support of the
;1579     2223  1 !       Basic language.  The number of significant digits returned depends on
;1580     2224  1 !       the data type.
;1581     2225  1 !
;1582     2226  1 ! FORMAL PARAMETERS:
;1583     2227  1 !
;1584     2228  1 !       VALUE.rz.r              the value to be converted
;1585     2229  1 !       NO_INT_DIGITS.rlu.r     number of integer digits in output string
;1586     2230  1 !       NO_FRAC_DIGITS.rlu.r    number of fraction digits in output string
;1587     2231  1 !       FLAGS.rlu.v             no flags are applicable
;1588     2232  1 !       BAS_OUT_STR_LEN.wlu.v   length of output string (handy for fixed length
;1589     2233  1 !                               output strings
;1590     2234  1 !       BAS_OUT_STR.wt.dx       the output string
;1591     2235  1 !       BAS_SCALE_FAC.rb.v      Basic scale factor in a range of 0 to -6
;1592     2236  1 !       CURRENCY.rt.dx          currency symbol
;1593     2237  1 !       DIGIT_SEP.rt.dx         digit group separator
;1594     2238  1 !       RADIX_POINT.rt.dx       radix point
;1595     2239  1 !       DATA_TYPE.rl.v          data type of element
;1596     2240  1 !
;1597     2241  1 ! IMPLICIT INPUTS:
;1598     2242  1 !
;1599     2243  1 !       NONE
;1600     2244  1 !
;1601     2245  1 ! IMPLICIT OUTPUTS:
;1602     2246  1 !
;1603     2247  1 !       NONE
;1604     2248  1 !
;1605     2249  1 ! COMPLETION CODES:
;1606     2250  1 !
;1607     2251  1 !       success         the value could be converted as required
;1608     2252  1 !       failure         the value did not fit in the field described
;1609     2253  1 !                       the return string was truncated
;1610     2254  1 !
;1611     2255  1 ! SIDE EFFECTS:
;1612     2256  1 !
;1613     2257  1 !       NONE
;1614     2258  1 !
;1615     2259  1 !--
;1616     2260  1
```

```
; 1617          2261  2     BEGIN
; 1618          2262  2
; 1619          2263  2     MAP
; 1620          2264  2         RADIX_PT : REF BLOCK [, BYTE],
; 1621          2265  2         CURRENCY : REF BLOCK [, BYTE],
; 1622          2266  2         DIGIT_SEP : REF BLOCK [, BYTE],
; 1623          2267  2         BAS_SCALE_FAC : VECTOR [1, BYTE, SIGNED],
; 1624          2268  2         BAS_OUT_STR_LEN : REF VECTOR,
; 1625          2269  2         VALUE : REF VECTOR,
; 1626          2270  2         BAS_OUT_STR : REF VECTOR [2, LONG];
; 1627          2271  2
; 1628          2272  2     LOCAL
; 1629          2273  2         TEMP_STRING : VECTOR [K_TMP_STR_LEN_H, BYTE],   ! temp string for kernel conversion result
; 1630          2274  2         TEMP_PTR : REF VECTOR [, BYTE],                 ! pointer into temp string
; 1631          2275  2         NO_DIGITS,                                      ! no. of actual significant digits returned
; 1632          2276  2                                                         ! by the kernel conversion routine
; 1633          2277  2         KERNEL_BLOCK : BLOCK [K_KERNEL_BLK_SZ, BYTE]
; 1634          2278  2                         INITIAL (REP (K_KERNEL_BLK_SZ/4) OF (0)),
; 1635          2279  2         SIG_DIG,
; 1636          2280  2         RESULT;
; 1637          2281  2
; 1638          2282  2     KERNEL_BLOCK [STRING_ADDR] = TEMP_STRING;
; 1639          2283  2
; 1640          2284  2     SELECTONE .DATA_TYPE OF
; 1641          2285  2         SET
; 1642          2286  2             [DSC$K_DTYPE_P]:
; 1643          2287  3                 BEGIN
; 1644          2288  3                 MAP
; 1645          2289  3                     VALUE : REF BLOCK [12,BYTE];
; 1646          2290  3                 RESULT = CMPP (VALUE [DSC$W_LENGTH], .VALUE [DSC$A_POINTER],
; 1647          2291  3                         %REF (15), PACKED_ZERO);
; 1648          2292  2                 END;
; 1649          2293  2             [OTHERWISE]:
; 1650          2294  2                 RESULT = .VALUE [0];
; 1651          2295  2         TES;
; 1652          2296  2
; 1653          2297  2     IF .RESULT EQL 0
; 1654          2298  2     THEN
; 1655          2299  2 !+
; 1656          2300  2 ! If a zero value should be suppressed, then complete processing here, without
; 1657          2301  2 ! calling PLAIN_ or FANCY_ formatting routines.  All that needs to be output
; 1658          2302  2 ! is a string of blanks.
; 1659          2303  2 !-
; 1660          2304  3         BEGIN
; 1661          2305  4         IF ((.FLAGS AND V_SUPPRESS_ZERO) EQL 0)
; 1662          2306  3         THEN
; 1663          2307  4             PUT_OUT_ZERO
; 1664          2308  3         ELSE
; 1665          2309  4             BEGIN
; 1666          2310  4             LOCAL BLANKS_STR_DESC : BLOCK [8, BYTE];
; 1667          2311  4
; 1668          2312  4             INITIALIZE_DESC;
; 1669          2313  4             BLANKS_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
; 1670          2314  4             BLANKS_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
; 1671          2315  4             BLANKS_STR_DESC [DSC$A_POINTER] = BLANKS;
; 1672          2316  4             BLANKS_STR_DESC [DSC$W_LENGTH] = .NO_INT_DIGITS + .NO_FRAC_DIGITS
; 1673          2317  5                 + (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN 1 ELSE 0)
```

```
;  1674          2318  5                 + (IF (.FLAGS AND V_CURRENCY) NEQ 0
;  1675          2319  5                     THEN .CURRENCY [DSC$W_LENGTH] ELSE 0)
;  1676          2320  5                 + (IF (.FLAGS AND V_COMMA) NEQ 0
;  1677          2321  5                     THEN (.KERNEL_BLOCK [DEC_EXP]-1)/3 * .DIGIT_SEP [DSC$W_LENGTH]
;  1678          2322  5                     ELSE 0)
;  1679          2323  4                 + .RADIX_PT [DSC$W_LENGTH];
;  1680          2324  4             STR$CONCAT (BAS_OUT_STR [0], BLANKS_STR_DESC);
;  1681          2325  4             RETURN 1;
;  1682          2326  3             END;
;  1683          2327  3         END
;  1684          2328  3
;  1685          2329  2     ELSE
;  1686          2330  3         BEGIN
;  1687          2331  3 !+
;  1688          2332  3 ! Figure out the number of significant digits to ask for.  It is the minimum of
;  1689          2333  3 ! the maximum that the data type can accurately hold or the total number of
;  1690          2334  3 ! digits asked for subtracting one if the value is negative.
;  1691          2335  3 !-
;  1692          2336  3         SELECTONE .DATA_TYPE OF
;  1693          2337  3             SET
;  1694          2338  3                 [DSC$K_DTYPE_F]:
;  1695          2339  3                     SIG_DIG = K_NUM_F_SIG_DIG;
;  1696          2340  3                 [DSC$K_DTYPE_D]:
;  1697          2341  3                     SIG_DIG = K_NUM_D_SIG_DIG;
;  1698          2342  3                 [DSC$K_DTYPE_P]:
;  1699          2343  3                     SIG_DIG = K_NUM_P_SIG_DIG;
;  1700          2344  3                 [DSC$K_DTYPE_G]:
;  1701          2345  3                     SIG_DIG = K_NUM_G_SIG_DIG;
;  1702          2346  3                 [DSC$K_DTYPE_H]:
;  1703          2347  3                     SIG_DIG = K_NUM_H_SIG_DIG;
;  1704          2348  3                 [OTHERWISE]:
;  1705          2349  3                     LIB$STOP (OTS$_FATINTERR);
;  1706          2350  3             TES;
;  1707          2351  3         KERNEL_BLOCK [SIG_DIGITS] = MIN (.SIG_DIG,
;  1708          2352  3             .NO_INT_DIGITS + .NO_FRAC_DIGITS - (IF (.(VALUE [0])<15, 1> EQL 1) THEN 1 ELSE 0));
;  1709          2353  3         KERNEL_BLOCK [CONV_FLAGS] = KERNEL_BLOCK [RT_RND] = 0;
;  1710          2354  3         SELECTONE .DATA_TYPE OF
;  1711          2355  3         SET
;  1712          2356  3             [DSC$K_DTYPE_F]:
;  1713          2357  3                 OTS$$CVT_D_T_R8 (.VALUE, KERNEL_BLOCK [KERNEL_PTR]);
;  1714          2358  3             [DSC$K_DTYPE_D]:
;  1715          2359  3                 OTS$$CVT_D_T_R8 (.VALUE, KERNEL_BLOCK [KERNEL_PTR]);
;  1716          2360  3             [DSC$K_DTYPE_P]:
;  1717          2361  4                 BEGIN
;  1718          2362  4                 !+
;  1719          2363  4                 ! No kernel conversion routine so convert to text here and
;  1720          2364  4                 ! set some parameters.
;  1721          2365  4                 !-
;  1722          2366  4                 MAP
;  1723          2367  4                     VALUE : REF BLOCK [12, BYTE];
;  1724          2368  4                 LOCAL
;  1725          2369  4                     ZERO_CTR : INITIAL (0),
;  1726          2370  4                     INT_DIGITS;
;  1727          2371  4
;  1728          2372  4                 CVTPS (VALUE [DSC$W_LENGTH], .VALUE [DSC$A_POINTER],
;  1729          2373  4                        VALUE [DSC$W_LENGTH], TEMP_STRING);
;  1730          2374  4                 !+
```

```
;  1731          2375  4                ! Calculate the number of integer digits, excluding leading
;  1732          2376  4                ! zeroes.
;  1733          2377  4                !-
;  1734          2378  4                INT_DIGITS = .VALUE [DSC$W_LENGTH] + .VALUE [DSC$B_SCALE];
;  1735          2379  4                INCR I FROM 1 TO .INT_DIGITS DO
;  1736          2380  4                    IF .TEMP_STRING [.I] EQL %C'0' THEN ZERO_CTR = .ZERO_CTR + 1
;  1737          2381  4                    ELSE EXITLOOP;
;  1738          2382  4                KERNEL_BLOCK [DEC_EXP] = .INT_DIGITS - .ZERO_CTR;
;  1739          2383  4                KERNEL_BLOCK [OFFSET] = 1 + .ZERO_CTR;
;  1740          2384  4                IF .TEMP_STRING [0] EQL %C'-'
;  1741          2385  4                THEN
;  1742          2386  5                    BEGIN
;  1743          2387  5                    KERNEL_BLOCK [SIGN] = -1;
;  1744          2388  5                    KERNEL_BLOCK [SIG_DIGITS] = .KERNEL_BLOCK [SIG_DIGITS] -
;  1745          2389  5                        (IF (.FLAGS AND V_TRAILING_SIGN) EQL 0 THEN 1 ELSE 0);
;  1746          2390  5                    END
;  1747          2391  4                ELSE
;  1748          2392  4                    KERNEL_BLOCK [SIGN] = 1;
;  1749          2393  3                END;
;  1750          2394  3            [DSC$K_DTYPE_G]:
;  1751          2395  3                OTS$$CVT_G_T_R8 (.VALUE, KERNEL_BLOCK [KERNEL_PTR]);
;  1752          2396  3            [DSC$K_DTYPE_H]:
;  1753          2397  3                OTS$$CVT_H_T_R8 (.VALUE, KERNEL_BLOCK [KERNEL_PTR]);
;  1754          2398  3            [OTHERWISE]:
;  1755          2399  3                LIB$STOP (OTS$_FATINTERR);
;  1756          2400  3            TES;
;  1757          2401  3        !+
;  1758          2402  3        ! Do some error checking.  Are there enough digits to return this number?
;  1759          2403  3        !-
;  1760          2404  3
;  1761          2405  4            IF (.KERNEL_BLOCK [SIGN] LSS 0)
;  1762          2406  3            THEN
;  1763          2407  3
;  1764          2408  3                IF (.NO_INT_DIGITS LSS 1) OR ((.NO_INT_DIGITS EQL 1) AND (.NO_FRAC_DIGITS LSS 1)) THEN RETURN 0;
;  1765          2409  3
;  1766          2410  3        !+
;  1767          2411  3        ! Adjust for the scale factor.  The scale factor will be zero for data types
;  1768          2412  3        ! other than double.
;  1769          2413  3        !-
;  1770          2414  3            KERNEL_BLOCK [DEC_EXP] = .KERNEL_BLOCK [DEC_EXP] + .BAS_SCALE_FAC [0];
;  1771          2415  3        !+
;  1772          2416  3        ! Check for the actual number of significant digits returned.
;  1773          2417  3        !-
;  1774          2418  3            IF .DATA_TYPE EQL DSC$K_DTYPE_P
;  1775          2419  3            THEN
;  1776          2420  4                BEGIN
;  1777          2421  4                MAP
;  1778          2422  4                    VALUE: REF BLOCK [12,BYTE];
;  1779          2423  4                TEMP_PTR = TEMP_STRING + .VALUE [DSC$W_LENGTH];
;  1780          2424  4                END
;  1781          2425  3            ELSE
;  1782          2426  3                TEMP_PTR = .KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET] + .KERNEL_BLOCK [SIG_DIGITS] - 1;
;  1783          2427  3
;  1784          2428  3                WHILE .TEMP_PTR [0] EQL %C'0' DO
;  1785          2429  3                    TEMP_PTR = .TEMP_PTR - 1;
;  1786          2430  3
;  1787          2431  3                NO_DIGITS = .TEMP_PTR - (.KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET]) + 1;
```

```
; 1788          2432  2           END;
; 1789          2433  2
; 1790          2434  2       FANCY_E_FORM_11 (.NO_INT_DIGITS, .NO_FRAC_DIGITS, .NO_DIGITS, .FLAGS, KERNEL_BLOCK, .CURRENCY,
; 1791          2435  2           .DIGIT_SEP, .RADIX_PT, BAS_OUT_STR [0], BAS_OUT_STR_LEN [0], .DATA_TYPE);
; 1792          2436  2       RETURN 1;
; 1793          2437  1       END;                                        !End of COMMON_E

                    0029B              .BLKB     1
         00000000#  0029C  P.AAB:      .LONG     0[9]                                                   ;
               2E   002C0  P.AAC:      .ASCII    \.\                                                    ;
                    002C1              .BLKB     3
               2A   002C4  P.AAD:      .ASCII    \*\                                                    ;
               2A   002C5              .ASCII    \*\                                                    ;
               2A   002C6              .ASCII    \*\                                                    ;
               2A   002C7              .ASCII    \*\                                                    ;
               2A   002C8              .ASCII    \*\                                                    ;
               2A   002C9              .ASCII    \*\                                                    ;
               2A   002CA              .ASCII    \*\                                                    ;
               2A   002CB              .ASCII    \*\                                                    ;
               2A   002CC              .ASCII    \*\                                                    ;
               2A   002CD              .ASCII    \*\                                                    ;
               2A   002CE              .ASCII    \*\                                                    ;
               2A   002CF              .ASCII    \*\                                                    ;
               2A   002D0              .ASCII    \*\                                                    ;
               2A   002D1              .ASCII    \*\                                                    ;
               2A   002D2              .ASCII    \*\                                                    ;
               2A   002D3              .ASCII    \*\                                                    ;
               2A   002D4              .ASCII    \*\                                                    ;
               2A   002D5              .ASCII    \*\                                                    ;
               2A   002D6              .ASCII    \*\                                                    ;
               2A   002D7              .ASCII    \*\                                                    ;
               2A   002D8              .ASCII    \*\                                                    ;
               2A   002D9              .ASCII    \*\                                                    ;
               2A   002DA              .ASCII    \*\                                                    ;
               2A   002DB              .ASCII    \*\                                                    ;
               2A   002DC              .ASCII    \*\                                                    ;
               2A   002DD              .ASCII    \*\                                                    ;
               2A   002DE              .ASCII    \*\                                                    ;
               2A   002DF              .ASCII    \*\                                                    ;
               2A   002E0              .ASCII    \*\                                                    ;
               2A   002E1              .ASCII    \*\                                                    ;
               2A   002E2              .ASCII    \*\                                                    ;
               2A   002E3              .ASCII    \*\                                                    ;
               2A   002E4              .ASCII    \*\                                                    ;
               2A   002E5              .ASCII    \*\                                                    ;
               2A   002E6              .ASCII    \*\                                                    ;
               2A   002E7              .ASCII    \*\                                                    ;
               2A   002E8              .ASCII    \*\                                                    ;
               2A   002E9              .ASCII    \*\                                                    ;
                    002EA              .BLKB     2
               2D   002EC  P.AAE:      .ASCII    \-\                                                    ;
                    002ED              .BLKB     3
               20   002F0  P.AAF:      .ASCII    \ \                                                    ;
                    002F1              .BLKB     3
               45   002F4  P.AAG:      .ASCII    \E\                                                    ;
```

```
                002F5                   .BLKB   3
20              002F8 P.AAH:            .ASCII  \ \
20              002F9                   .ASCII  \ \
20              002FA                   .ASCII  \ \
20              002FB                   .ASCII  \ \
20              002FC                   .ASCII  \ \
20              002FD                   .ASCII  \ \
20              002FE                   .ASCII  \ \
20              002FF                   .ASCII  \ \
20              00300                   .ASCII  \ \
20              00301                   .ASCII  \ \
20              00302                   .ASCII  \ \
20              00303                   .ASCII  \ \
20              00304                   .ASCII  \ \
20              00305                   .ASCII  \ \
20              00306                   .ASCII  \ \
20              00307                   .ASCII  \ \
20              00308                   .ASCII  \ \
20              00309                   .ASCII  \ \
20              0030A                   .ASCII  \ \
20              0030B                   .ASCII  \ \
20              0030C                   .ASCII  \ \
20              0030D                   .ASCII  \ \
20              0030E                   .ASCII  \ \
20              0030F                   .ASCII  \ \
20              00310                   .ASCII  \ \
20              00311                   .ASCII  \ \
20              00312                   .ASCII  \ \
20              00313                   .ASCII  \ \
20              00314                   .ASCII  \ \
20              00315                   .ASCII  \ \
20              00316                   .ASCII  \ \
20              00317                   .ASCII  \ \
20              00318                   .ASCII  \ \
20              00319                   .ASCII  \ \
20              0031A                   .ASCII  \ \
20              0031B                   .ASCII  \ \
20              0031C                   .ASCII  \ \
20              0031D                   .ASCII  \ \
                0031E                   .BLKB   2
30              00320 P.AAI:            .ASCII  \0\
30              00321                   .ASCII  \0\
30              00322                   .ASCII  \0\
30              00323                   .ASCII  \0\
30              00324                   .ASCII  \0\
30              00325                   .ASCII  \0\
30              00326                   .ASCII  \0\
30              00327                   .ASCII  \0\
30              00328                   .ASCII  \0\
30              00329                   .ASCII  \0\
30              0032A                   .ASCII  \0\
30              0032B                   .ASCII  \0\
30              0032C                   .ASCII  \0\
30              0032D                   .ASCII  \0\
30              0032E                   .ASCII  \0\
30              0032F                   .ASCII  \0\
30              00330                   .ASCII  \0\
```

```
                                        30  00331              .ASCII   \0\                        ;
                                        30  00332              .ASCII   \0\                        ;
                                        30  00333              .ASCII   \0\                        ;
                                        30  00334              .ASCII   \0\                        ;
                                        30  00335              .ASCII   \0\                        ;
                                        30  00336              .ASCII   \0\                        ;
                                        30  00337              .ASCII   \0\                        ;
                                        30  00338              .ASCII   \0\                        ;
                                        30  00339              .ASCII   \0\                        ;
                                        30  0033A              .ASCII   \0\                        ;
                                        30  0033B              .ASCII   \0\                        ;
                                        30  0033C              .ASCII   \0\                        ;
                                        30  0033D              .ASCII   \0\                        ;
                                        30  0033E              .ASCII   \0\                        ;
                                        30  0033F              .ASCII   \0\                        ;
                                        30  00340              .ASCII   \0\                        ;
                                        30  00341              .ASCII   \0\                        ;
                                        30  00342              .ASCII   \0\                        ;
                                        30  00343              .ASCII   \0\                        ;
                                        30  00344              .ASCII   \0\                        ;
                                        30  00345              .ASCII   \0\                        ;

                                                  DOT=                   P.AAC
                                                  STAR=                  P.AAD
                                                  MINUS=                 P.AAE
                                                  BLANK=                 P.AAF
                                                  E=                     P.AAG
                                                  BLANKS=                P.AAH
                                                  ZEROES=                P.AAI

                                 0BFC   8F  BB 00000 COMMON_E:
                                                           PUSHR    #^M<R2,R3,R4,R5,R6,R7,R8,R9,R11>   ; 2204
                           5E     8C    AE  9E 00004       MOVAB    -116(SP), SP                       ;
                                0380    8F  BB 00008       PUSHR    #^M<R7,R8,R9>                      ;
                     10    AE           56  D0 0000C       MOVL     R6, BAS_SCALE_FAC                  ;
                                        30  BB 00010       PUSHR    #^M<R4,R5>                         ;
                     20    AE           53  D0 00012       MOVL     R3, FLAGS                          ;
                     14    AE           52  D0 00016       MOVL     R2, NO_FRAC_DIGITS                 ;
                     1C    AE           51  D0 0001A       MOVL     R1, NO_INT_DIGITS                  ;
                           5B           50  D0 0001E       MOVL     R0, R11                            ;
          30    AE  FF30   CF           24  28 00021       MOVC3    #36, P.AAB, KERNEL_BLOCK           ; 2278
                     40    AE     54    AE  9E 00028       MOVAB    TEMP_STRING, KERNEL_BLOCK+16       ; 2282
                           15           5A  D1 0002D       CMPL     DATA_TYPE, #21                     ; 2286
                                        15  12 00030       BNEQ     1$                                 ;
FC80  CF        0F   04    BB           6B  37 00032       CMPP4    (VALUE), @4(VALUE), #15, PACKED_ZERO ; 2290
                                        54  DC 0003A       MOVPSL   R4                                 ;
      54        54         02           02  EF 0003C       EXTZV    #2, #2, R4, R4                     ;
                50         01           54  C3 00041       SUBL3    R4, #1, RESULT                     ;
                                        03  11 00045       BRB      2$                                 ; 2284
                           50           6B  D0 00047 1$:   MOVL     (VALUE), RESULT                    ; 2294
                56   1C    AE     14    AE  C1 0004A 2$:   ADDL3    NO_FRAC_DIGITS, NO_INT_DIGITS, R6  ; 2306
                                        50  D5 00050       TSTL     RESULT                             ; 2297
                                        03  13 00052       BEQL     3$                                 ;
                                      00AA  31 00054       BRW      12$                                ;
                1D   20    AE           0C  E0 00057 3$:   BBS      #12, FLAGS, 4$                     ; 2305
                                  34    AE  D4 0005C       CLRL     KERNEL_BLOCK+4                     ; 2306
```

```
                        38  AE          01  7D 0005F           MOVQ    #1, KERNEL_BLOCK+8                      ;
                        44  AE          01  D0 00063           MOVL    #1, KERNEL_BLOCK+20                     ;
                        24  AE          01  D0 00067           MOVL    #1, NO_DIGITS                           ;
                        40  BE          30  90 0006B           MOVB    #48, @KERNEL_BLOCK+16                   ;
     56          30         6E          00  2C 0006F           MOVC5   #0, (SP), #48, R6, @KERNEL_BLOCK+16     ;
                                  40    BE     00074                                                           ;
                                      01DE  31 00076           BRW     39$                                     ; 2305
                           00000000'    EF  D5 00079 4$:       TSTL    E_D                                     ; 2310
                                        24  12 0007F           BNEQ    5$                                      ;
                 00000000'  EF    FF21  CF  9E 00081           MOVAB   MINUS, M_D                              ;
                 00000000'  EF    FEEC  CF  9E 0008A           MOVAB   DOT, D_D                                ;
                 00000000'  EF    FF13  CF  9E 00093           MOVAB   BLANK, B_D                              ;
                 00000000'  EF    FF0E  CF  9E 0009C           MOVAB   E, E_D                                  ;
                        2A  AE    010E  8F  B0 000A5 5$:       MOVW    #270, BLANKS_STR_DESC+2                 ; 2313
                        2C  AE    FF03  CF  9E 000AB           MOVAB   BLANKS, BLANKS_STR_DESC+4               ; 2315
                 05     20  AE          01  E1 000B1           BBC     #1, FLAGS, 6$                           ; 2317
                            50          01  D0 000B6           MOVL    #1, R0                                  ;
                                        02  11 000B9           BRB     7$                                      ;
                                        50  D4 000BB 6$:       CLRL    R0                                      ;
                            50          56  C0 000BD 7$:       ADDL2   R6, R0                                  ;
                 06     20  AE          03  E1 000C0           BBC     #3, FLAGS, 8$                           ; 2318
                            57    08    BE  3C 000C5           MOVZWL  @CURRENCY, R7                           ; 2319
                                        02  11 000C9           BRB     9$                                      ;
                                        57  D4 000CB 8$:       CLRL    R7                                      ; 2318
                            50          57  C0 000CD 9$:       ADDL2   R7, R0                                  ;
                 11     20  AE          02  E1 000D0           BBC     #2, FLAGS, 10$                          ; 2320
                 51     38  AE          01  C3 000D5           SUBL3   #1, KERNEL_BLOCK+8, R1                  ; 2321
                            51          03  C6 000DA           DIVL2   #3, R1                                  ;
                            52    0C    BE  3C 000DD           MOVZWL  @DIGIT_SEP, R2                          ;
                            51          52  C4 000E1           MULL2   R2, R1                                  ;
                                        02  11 000E4           BRB     11$                                     ;
                                        51  D4 000E6 10$:      CLRL    R1                                      ; 2320
                            50          51  C0 000E8 11$:      ADDL2   R1, R0                                  ;
           28    AE         50    10    BE  A1 000EB           ADDW3   @RADIX_PT, R0, BLANKS_STR_DESC          ; 2323
                                  28    AE  9F 000F1           PUSHAB  BLANKS_STR_DESC                         ; 2324
                                  08    AE  DD 000F4           PUSHL   BAS_OUT_STR                             ;
                 00000000G  00          02  FB 000F7           CALLS   #2, STR$CONCAT                          ;
                                      017C  31 000FE           BRW     40$                                     ; 2325
                            0A          5A  D1 00101 12$:      CMPL    DATA_TYPE, #10                          ; 2338
                                        05  12 00104           BNEQ    13$                                     ;
                            52          06  D0 00106           MOVL    #6, SIG_DIG                             ; 2339
                                        35  11 00109           BRB     18$                                     ;
                            0B          5A  D1 0010B 13$:      CMPL    DATA_TYPE, #11                          ; 2340
                                        05  12 0010E           BNEQ    14$                                     ;
                            52          10  D0 00110           MOVL    #16, SIG_DIG                            ; 2341
                                        2B  11 00113           BRB     18$                                     ;
                            15          5A  D1 00115 14$:      CMPL    DATA_TYPE, #21                          ; 2342
                                        05  12 00118           BNEQ    15$                                     ;
                            52          1F  D0 0011A           MOVL    #31, SIG_DIG                            ; 2343
                                        21  11 0011D           BRB     18$                                     ;
                            1B          5A  D1 0011F 15$:      CMPL    DATA_TYPE, #27                          ; 2344
                                        05  12 00122           BNEQ    16$                                     ;
                            52          0F  D0 00124           MOVL    #15, SIG_DIG                            ; 2345
                                        17  11 00127           BRB     18$                                     ;
                            1C          5A  D1 00129 16$:      CMPL    DATA_TYPE, #28                          ; 2346
                                        05  12 0012C           BNEQ    17$                                     ;
                            52          21  D0 0012E           MOVL    #33, SIG_DIG                            ; 2347
```

```
                                                 0D  11 00131          BRB     18$
                                00000000G        8F  DD 00133 17$:     PUSHL   #OTS$_FATINTERR                                          ; 2349
                                00000000G     00 01  FB 00139          CALLS   #1, LIB$STOP
                                               6B  B5 00140 18$:     TSTW    (VALUE)                                                  ; 2352
                                               05  18 00142          BGEQ    19$
                                            50 01  D0 00144          MOVL    #1, R0
                                               02  11 00147          BRB     20$
                                               50  D4 00149 19$:     CLRL    R0
                                         51    56  50 C3 0014B 20$:     SUBL3   R0, R6, R1
                                            51 52  D1 0014F          CMPL    R2, R1
                                               03  15 00152          BLEQ    21$
                                            52 51  D0 00154          MOVL    R1, R2
                                      44    AE 52  D0 00157 21$:     MOVL    R2, KERNEL_BLOCK+20                                      ; 2351
                                         30 AE  D4 0015B          CLRL    KERNEL_BLOCK                                             ; 2353
                                         48 AE  D4 0015E          CLRL    KERNEL_BLOCK+24
                                         0A 5A  D1 00161          CMPL    DATA_TYPE, #10                                           ; 2356
                                               05  13 00164          BEQL    22$
                                         0B 5A  D1 00166          CMPL    DATA_TYPE, #11                                           ; 2358
                                               0F  12 00169          BNEQ    23$
                                         51 54  AE 9E 0016B 22$:     MOVAB   KERNEL_BLOCK+36, R1                                      ; 2359
                                         50 5B  D0 0016F          MOVL    VALUE, R0
                                00000000G     00  16 00172          JSB     OTS$$CVT_D_T_R8
                                               79  11 00178          BRB     32$
                                         15 5A  D1 0017A 23$:     CMPL    DATA_TYPE, #21                                           ; 2360
                                               50  12 0017D          BNEQ    30$
                                               54  D4 0017F          CLRL    ZERO_CTR                                                 ; 2361
54    AE        6B        04  BB 6B  08 00181          CVTPS   (VALUE), @4(VALUE), (VALUE), TEMP_STRING                          ; 2373
                                         51 6B  3C 00188          MOVZWL  (VALUE), INT_DIGITS                                      ; 2378
                                         50 08  AB 98 0018B          CVTBL   8(VALUE), R0
                                         51 50  C0 0018F          ADDL2   R0, INT_DIGITS
                                               50  D4 00192          CLRL    I                                                        ; 2379
                                               09  11 00194          BRB     25$
                                      30    54 AE40  91 00196 24$:     CMPB    TEMP_STRING[I], #48                                      ; 2380
                                               06  12 0019B          BNEQ    26$
                                               54  D6 0019D          INCL    ZERO_CTR
                                F3             50  51 F3 0019F 25$:     AOBLEQ  INT_DIGITS, I, 24$
38    AE                        51  54 C3 001A3 26$:     SUBL3   ZERO_CTR, INT_DIGITS, KERNEL_BLOCK+8                             ; 2382
                      34    AE 01  A4 9E 001A8          MOVAB   1(R4), KERNEL_BLOCK+4                                            ; 2383
                         2D 54  AE 91 001AD          CMPB    TEMP_STRING, #45                                                 ; 2384
                                               16  12 001B1          BNEQ    29$
                      3C    AE                 01  CE 001B3          MNEGL   #1, KERNEL_BLOCK+12                                      ; 2387
                05    20    AE                 01  E0 001B7          BBS     #1, FLAGS, 27$                                           ; 2389
                                            50 01  D0 001BC          MOVL    #1, R0
                                               02  11 001BF          BRB     28$
                                               50  D4 001C1 27$:     CLRL    R0
                      44    AE                 50  C2 001C3 28$:     SUBL2   R0, KERNEL_BLOCK+20
                                               3B  11 001C7          BRB     34$                                                      ; 2384
                      3C    AE                 01  D0 001C9 29$:     MOVL    #1, KERNEL_BLOCK+12                                      ; 2392
                                               35  11 001CD          BRB     34$                                                      ; 2354
                                         1B 5A  D1 001CF 30$:     CMPL    DATA_TYPE, #27                                           ; 2394
                                               0F  12 001D2          BNEQ    31$
                                         51 54  AE 9E 001D4          MOVAB   KERNEL_BLOCK+36, R1                                      ; 2395
                                         50 5B  D0 001D8          MOVL    VALUE, R0
                                00000000G     00  16 001DB          JSB     OTS$$CVT_G_T_R8
                                               21  11 001E1          BRB     34$
                                         1C 5A  D1 001E3 31$:     CMPL    DATA_TYPE, #28                                           ; 2396
                                               0F  12 001E6          BNEQ    33$
```

```
                        51        54    AF  9E 001E8          MOVAB   KERNEL_BLOCK+36, R1                      ; 2397
                        50              5B  D0 001EC          MOVL    VALUE, R0
                   00000000G            00  16 001EF          JSB     OTS$$CVT_H_T_R8
                                        0D  11 001F5 32$:      BRB     34$
                   00000000G            8F  DD 001F7 33$:      PUSHL   #OTS$_FATINTERR                         ; 2399
                00000000G   00          01  FB 001FD          CALLS   #1, LIB$STOP
                                    3C  AE  D5 00204 34$:      TSTL    KERNEL_BLOCK+12                         ; 2405
                                        10  18 00207          BGEQ    35$
                                    1C  AE  D5 00209          TSTL    NO_INT_DIGITS                           ; 2408
                                        74  15 0020C          BLEQ    41$
                                01      1C  AE  D1 0020E          CMPL    NO_INT_DIGITS, #1
                                        05  12 00212          BNEQ    35$
                                    14  AE  D5 00214          TSTL    NO_FRAC_DIGITS
                                        69  15 00217          BLEQ    41$
                                50      18  AE  98 00219 35$:      CVTBL   BAS_SCALE_FAC, R0                       ; 2414
                             38 AE              50  C0 0021D          ADDL2   R0, KERNEL_BLOCK+8
                                15              5A  D1 00221          CMPL    DATA_TYPE, #21                  ; 2418
                                                0C  12 00224          BNEQ    36$
                                50      54  AE  9E 00226          MOVAB   TEMP_STRING, R0                 ; 2423
                                51              6B  3C 0022A          MOVZWL  (VALUE), R1
                                50              51  C0 0022D          ADDL2   R1, TEMP_PTR
                                                0E  11 00230          BRB     37$                             ; 2418
                        51      40  AE      34  AE  C1 00232 36$:      ADDL3   KERNEL_BLOCK+4, KERNEL_BLOCK+16, R1     ; 2426
                                51      44  AE  C0 00238          ADDL2   KERNEL_BLOCK+20, R1
                                50      FF  A1  9E 0023C          MOVAB   -1(R1), TEMP_PTR
                                30              60  91 00240 37$:      CMPB    (TEMP_PTR), #48                 ; 2428
                                                04  12 00243          BNEQ    38$
                                                50  D7 00245          DECL    TEMP_PTR                        ; 2429
                                                F7  11 00247          BRB     37$
                        52      40  AE      34  AE  C1 00249 38$:      ADDL3   KERNEL_BLOCK+4, KERNEL_BLOCK+16, R2     ; 2431
                                52              50  C2 0024F          SUBL2   TEMP_PTR, R2
                             24 AE              01  52  C3 00252          SUBL3   R2, #1, NO_DIGITS
                                54      30  AE  9E 00257 39$:      MOVAB   KERNEL_BLOCK, R4                ; 2434
                                59              6E  D0 0025B          MOVL    BAS_OUT_STR_LEN, R9             ; 2435
                                58      04  AE  D0 0025E          MOVL    BAS_OUT_STR, R8
                                56      0C  AE  7D 00262          MOVQ    DIGIT_SEP, R6
                                55      08  AE  D0 00266          MOVL    CURRENCY, R5
                                53      20  AE  D0 0026A          MOVL    FLAGS, R3
                                52      24  AE  D0 0026E          MOVL    NO_DIGITS, R2
                                51      14  AE  D0 00272          MOVL    NO_FRAC_DIGITS, R1
                                50      1C  AE  D0 00276          MOVL    NO_INT_DIGITS, R0
                                            0000V 30 0027A          BSBW    FANCY_E_FORM_11
                                50              01  D0 0027D 40$:      MOVL    #1, R0                          ; 2436
                                                02  11 00280          BRB     42$
                                                50  D4 00282 41$:      CLRL    R0                              ; 2437
                                5E      0088    CE  9E 00284 42$:      MOVAB   136(SP), SP
                                        0BFC    8F  BA 00289          POPR    #^M<R2,R3,R4,R5,R6,R7,R8,R9,R11>
                                                    05 0028D          RSB

; Routine Size:  654 bytes,    Routine Base:  _BAS$CODE + 0346

; 1794          2438  1
```

```
;1796     2439  1  ROUTINE COMMON_F (                                    !
;1797     2440  1          VALUE,                                        !
;1798     2441  1          NO_INT_DIGITS,                                !
;1799     2442  1          NO_FRAC_DIGITS,                               !
;1800     2443  1          FLAGS,                                        !
;1801     2444  1          BAS_OUT_STR_LEN,                              !
;1802     2445  1          BAS_OUT_STR,                                  !
;1803     2446  1          BAS_SCALE_FAC,                                !
;1804     2447  1          DATA_TYPE,                                    !
;1805     2448  1          ARG_COUNT,                                    !
;1806     2449  1          CURRENCY,                                     !
;1807     2450  1          DIGIT_SEP,                                    !
;1808     2451  1          RADIX_PT                                      !
;1809     2452  1      ): JSB_F_FORMAT =                                 !
;1810     2453  1
;1811     2454  1  !++
;1812     2455  1  ! FUNCTIONAL DESCRIPTION:
;1813     2456  1  !
;1814     2457  1  !       This routine converts a number to explicit point unscaled
;1815     2458  1  !       notation (F format) right justified.  This routine supports PRINT
;1816     2459  1  !       USING and NUM1$.  The number of significant digits returned
;1817     2460  1  !       depends on the data type.
;1818     2461  1  !
;1819     2462  1  ! FORMAL PARAMETERS:
;1820     2463  1  !
;1821     2464  1  !       VALUE.rz.r              value to convert
;1822     2465  1  !       NO_INT_DIGITS.rlu.v     number of integer digits
;1823     2466  1  !       NO_FRAC_DIGITS.rlu.v    number of fraction digits
;1824     2467  1  !       FLAGS.rlu.v             = 1, no leading or trailing space
;1825     2468  1  !                               = 2, trailing sign
;1826     2469  1  !                               = 4, insert commas
;1827     2470  1  !                               = 8, float currency symbol
;1828     2471  1  !                               = 16, * fill to left of most significant digit
;1829     2472  1  !                               = 32, floating decimal point.  Disregard
;1830     2473  1  !                                     NO_INT_DIGITS and NO_FRAC_DIGITS.
;1831     2474  1  !                               = 64, a decimal point is required
;1832     2475  1  !       BAS_OUT_STR_LEN.wlu.r   length of return string
;1833     2476  1  !       BAS_OUT_STR.wt.dx       the return string
;1834     2477  1  !       BAS_SCALE_FAC.rb.v      Basic scale factor from 0 to -6
;1835     2478  1  !       DATA_TYPE.rl.v          data type of element
;1836     2479  1  !       ARG_COUNT.ri.v          number of arguments passed
;1837     2480  1  !       [CURRENCY.rt.dx]        currency symbol
;1838     2481  1  !       [DIGIT_SEP.rt.dx]       digit group separator
;1839     2482  1  !       [RADIX_PT.rt.dx]        radix point
;1840     2483  1  !
;1841     2484  1  ! IMPLICIT INPUTS:
;1842     2485  1  !
;1843     2486  1  !       NONE
;1844     2487  1  !
;1845     2488  1  ! IMPLICIT OUTPUTS:
;1846     2489  1  !
;1847     2490  1  !       NONE
;1848     2491  1  !
;1849     2492  1  ! COMPLETION CODES:
;1850     2493  1  !
;1851     2494  1  !       success         the value could be converted as required
;1852     2495  1  !       failure         the value did not fit in the field described
```

```
;1853     2496  1 !
;1854     2497  1 ! SIDE EFFECTS:
;1855     2498  1 !
;1856     2499  1 !       NONE
;1857     2500  1 !--
;1858     2501  1 !
;1859     2502  1
;1860     2503  2     BEGIN
;1861     2504  2
;1862     2505  2     MAP
;1863     2506  2         RADIX_PT : REF BLOCK [, BYTE],
;1864     2507  2         BAS_SCALE_FAC : VECTOR [1, BYTE, SIGNED],
;1865     2508  2         CURRENCY : REF BLOCK [, BYTE],
;1866     2509  2         DIGIT_SEP : REF BLOCK [, BYTE],
;1867     2510  2         BAS_OUT_STR_LEN : REF VECTOR,
;1868     2511  2         VALUE : REF VECTOR,
;1869     2512  2         BAS_OUT_STR : REF VECTOR [2, LONG];
;1870     2513  2
;1871     2514  2     LOCAL
;1872     2515  2         TEMP_STRING : VECTOR [K_TMP_STR_LEN_H, BYTE],    ! string for kernel conversion to
;1873     2516  2                                                          ! return the result in
;1874     2517  2         TEMP_PTR : REF VECTOR [, BYTE],                  ! pointer into TEMP_STR
;1875     2518  2         NO_DIGITS,                                       ! no. of actual, useful digits returned
;1876     2519  2         KERNEL_BLOCK : BLOCK [K_KERNEL_BLK_SZ, BYTE]
;1877     2520  2                         INITIAL (REP (K_KERNEL_BLK_SZ/4) OF (0)),
;1878     2521  2         LARGEST_EXP,
;1879     2522  2         SIG_DIG,
;1880     2523  2         RESULT;
;1881     2524  2
;1882     2525  2     LITERAL
;1883     2526  2         K_ARG_COUNT = 12;
;1884     2527  2
;1885     2528  2 !+
;1886     2529  2 ! Convert this number and then check it to see if it is effectively
;1887     2530  2 ! zero after rounding to the number of places required.
;1888     2531  2 !-
;1889     2532  2     KERNEL_BLOCK [STRING_ADDR] = TEMP_STRING;
;1890     2533  2 !+
;1891     2534  2 ! Set the number of significant digits to the minimum of the number of total
;1892     2535  2 ! digits requested (minus one if negative) and the number of accurate significant digits that
;1893     2536  2 ! the data type can return.  This is done so that the kernel routine can do the
;1894     2537  2 ! rounding.
;1895     2538  2 !-
;1896     2539  2
;1897     2540  2   SELECTONE .DATA_TYPE OF
;1898     2541  2         SET
;1899     2542  2             [DSC$K_DTYPE_P]:
;1900     2543  3                 BEGIN
;1901     2544  3                     MAP
;1902     2545  3                         VALUE : REF BLOCK [12,BYTE];
;1903     2546  3                     RESULT = CMPP (VALUE [DSC$W_LENGTH], .VALUE [DSC$A_POINTER],
;1904     2547  3                         %REF(15), PACKED_ZERO)
;1905     2548  2                 END;
;1906     2549  2             [OTHERWISE]:
;1907     2550  2                 RESULT = .VALUE [0];
;1908     2551  2         TES;
;1909     2552  2
```

```
;  1910          2553  2      IF .RESULT EQL 0
;  1911          2554  2      THEN
;  1912          2555  2  !+
;  1913          2556  2  ! If a zero value should be suppressed, then complete processing here, without
;  1914          2557  2  ! calling PLAIN_ or FANCY_ formatting routines.  All that needs to be output
;  1915          2558  2  ! is a string of blanks.
;  1916          2559  2  !-
;  1917          2560  3          BEGIN
;  1918          2561  4          IF ((.FLAGS AND V_SUPPRESS_ZERO) EQL 0)
;  1919          2562  3          THEN
;  1920          2563  4              IF (.NO_INT_DIGITS GEQ 1)
;  1921          2564  3              THEN
;  1922          2565  4                  PUT_OUT_ZERO
;  1923          2566  3              ELSE
;  1924          2567  3                  !+
;  1925          2568  3                  ! Kludge for NUM1$.  NUM1$ always passes 0 for no_int_digits
;  1926          2569  3                  ! and no_frac_digits, but should print "0" not ".0" for
;  1927          2570  3                  ! NUM1$(0).  So make sure 0 looks like an integer to the
;  1928          2571  3                  ! formatting routine.
;  1929          2572  3                  !-
;  1930          2573  4                  BEGIN
;  1931          2574  5                  IF (.NO_INT_DIGITS EQL 0 AND .NO_FRAC_DIGITS EQL 0)
;  1932          2575  4                  THEN
;  1933          2576  5                      PUT_OUT_ZERO
;  1934          2577  4                  ELSE
;  1935          2578  5                      PUT_OUT_ZERO_FRACT
;  1936          2579  4                  END
;  1937          2580  3          ELSE
;  1938          2581  4              BEGIN
;  1939          2582  4              LOCAL BLANKS_STR_DESC : BLOCK [8, BYTE];
;  1940          2583  4
;  1941          2584  4              INITIALIZE_DESC;
;  1942          2585  4              BLANKS_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  1943          2586  4              BLANKS_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  1944          2587  4              BLANKS_STR_DESC [DSC$A_POINTER] = BLANKS;
;  1945          2588  4              BLANKS_STR_DESC [DSC$W_LENGTH] = .NO_INT_DIGITS + .NO_FRAC_DIGITS
;  1946          2589  5                  + (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN 1 ELSE 0)
;  1947          2590  5                  + (IF (.FLAGS AND V_COMMA) NEQ 0
;  1948          2591  5                      THEN (.KERNEL_BLOCK [DEC_EXP]-1)/3 * .DIGIT_SEP [DSC$W_LENGTH]
;  1949          2592  5                      ELSE 0)
;  1950          2593  5                  + (IF (.FLAGS AND V_PERIOD) NEQ 0 THEN .RADIX_PT [DSC$W_LENGTH]
;  1951          2594  4                      ELSE 0);
;  1952          2595  4              STR$CONCAT (BAS_OUT_STR [0], BLANKS_STR_DESC);
;  1953          2596  4              RETURN 1;
;  1954          2597  3              END;
;  1955          2598  3          END
;  1956          2599  3
;  1957          2600  2      ELSE
;  1958          2601  3          BEGIN
;  1959          2602  3          SELECTONE .DATA_TYPE OF
;  1960          2603  3              SET
;  1961          2604  3              [DSC$K_DTYPE_F]:
;  1962          2605  3                  SIG_DIG = K_NUM_F_SIG_DIG;
;  1963          2606  3              [DSC$K_DTYPE_D]:
;  1964          2607  3                  SIG_DIG = K_NUM_D_SIG_DIG;
;  1965          2608  3              [DSC$K_DTYPE_P]:
;  1966          2609  4                  BEGIN
```

```
;  1967          2610  4                    MAP
;  1968          2611  4                        VALUE : REF BLOCK [12,BYTE];
;  1969          2612  4
;  1970          2613  4                    SIG_DIG = .VALUE [DSC$W_LENGTH] + .VALUE [DSC$B_SCALE];
;  1971          2614  4                    END;
;  1972          2615  3                [DSC$K_DTYPE_G]:
;  1973          2616  3                    SIG_DIG = K_NUM_G_SIG_DIG;
;  1974          2617  3                [DSC$K_DTYPE_H]:
;  1975          2618  3                    SIG_DIG = K_NUM_H_SIG_DIG;
;  1976          2619  3                [OTHERWISE]:
;  1977          2620  3                    LIB$STOP (OTS$_FATINTERR);
;  1978          2621  2                TES;
;  1979          2622  2            KERNEL_BLOCK [SIG_DIGITS] =
;  1980          2623  2            (IF ((.FLAGS AND V_FLT_DEC_PT) EQL 0) THEN MIN (.NO_INT_DIGITS + .NO_FRAC_DIGITS -
;  1981          2624  3                BEGIN
;  1982          2625  3
;  1983          2626  3                IF ((.(VALUE [0])<15, 1> EQL 1) AND ((.FLAGS AND V_TRAILING_SIGN) EQL 0)) THEN 1 ELSE 0
;  1984          2627  3
;  1985          2628  3                END
;  1986          2629  3                , .SIG_DIG) ELSE .SIG_DIG);
;  1987          2630  2            !+
;  1988          2631  2            ! Do not set the right round flag if this is a floating decimal point number.
;  1989          2632  2            !-
;  1990          2633  2            KERNEL_BLOCK [CONV_FLAGS] = (IF (.FLAGS AND V_FLT_DEC_PT) EQL 0 THEN V_RT_RND ELSE 0);
;  1991          2634  2            !+
;  1992          2635  2            ! The scale factor will be zero if this is not a double precision value.
;  1993          2636  2            !-
;  1994          2637  2            KERNEL_BLOCK [RT_RND] = .BAS_SCALE_FAC + .NO_FRAC_DIGITS;
;  1995          2638  2            SELECTONE .DATA_TYPE OF
;  1996          2639  2            SET
;  1997          2640  3                [DSC$K_DTYPE_F]:
;  1998          2641  3                    OTS$$CVT_D_T_R8 (.VALUE, KERNEL_BLOCK [KERNEL_PTR]);
;  1999          2642  3                [DSC$K_DTYPE_D]:
;  2000          2643  3                    OTS$$CVT_D_T_R8 (.VALUE, KERNEL_BLOCK [KERNEL_PTR]);
;  2001          2644  3                [DSC$K_DTYPE_P]:
;  2002          2645  4                    BEGIN
;  2003          2646  4                    !+
;  2004          2647  4                    ! No kernel conversion routine so convert to text here and
;  2005          2648  4                    ! set some parameters.
;  2006          2649  4                    !-
;  2007          2650  4                    MAP
;  2008          2651  4                        VALUE : REF BLOCK [12, BYTE];
;  2009          2652  4                    LOCAL
;  2010          2653  4                        ZERO_CTR : INITIAL (0),
;  2011          2654  4                        INT_DIGITS;
;  2012          2655  4
;  2013          2656  4                    CVTPS (VALUE [DSC$W_LENGTH], .VALUE [DSC$A_POINTER],
;  2014          2657  4                            VALUE [DSC$W_LENGTH], TEMP_STRING);
;  2015          2658  4                    !+
;  2016          2659  4                    ! Calculate the number of integer digits, excluding leading
;  2017          2660  4                    ! zeroes.
;  2018          2661  4                    !-
;  2019          2662  4                    INT_DIGITS = .VALUE [DSC$W_LENGTH] + .VALUE [DSC$B_SCALE];
;  2020          2663  4                    INCR I FROM 1 TO .INT_DIGITS DO
;  2021          2664  4                        IF .TEMP_STRING [.I] EQL %C'0' THEN ZERO_CTR = .ZERO_CTR + 1
;  2022          2665  4                        ELSE EXITLOOP;
;  2023          2666  4                    KERNEL_BLOCK [DEC_EXP] = .INT_DIGITS - .ZERO_CTR;
```

```
;  2024          2667  4                KERNEL_BLOCK [OFFSET] = 1 + .ZERO_CTR;
;  2025          2668  4                !+
;  2026          2669  4                ! Set the significant digits to the actual number of digits.
;  2027          2670  4                !-
;  2028          2671  4                SIG_DIG = .KERNEL_BLOCK [DEC_EXP] + ABS(.VALUE [DSC$B_SCALE]);
;  2029          2672  4                IF .TEMP_STRING [0] EQL %C'-'
;  2030          2673  4                THEN
;  2031          2674  5                    BEGIN
;  2032          2675  5                    KERNEL_BLOCK [SIGN] = -1;
;  2033          2676  5                    KERNEL_BLOCK [SIG_DIGITS] = .KERNEL_BLOCK [SIG_DIGITS] -
;  2034          2677  5                        (IF (.FLAGS AND V_TRAILING_SIGN) EQL 0 THEN 1 ELSE 0);
;  2035          2678  5                    END
;  2036          2679  4                ELSE
;  2037          2680  4                    KERNEL_BLOCK [SIGN] = 1;
;  2038          2681  3                END;
;  2039          2682  3            [DSC$K_DTYPE_G]:
;  2040          2683  3                OTS$$CVT_G_T_R8 (.VALUE, KERNEL_BLOCK [KERNEL_PTR]);
;  2041          2684  3            [DSC$K_DTYPE_H]:
;  2042          2685  3                OTS$$CVT_H_T_R8 (.VALUE, KERNEL_BLOCK [KERNEL_PTR]);
;  2043          2686  3            [OTHERWISE]:
;  2044          2687  3                LIB$STOP (OTS$_FATINTERR);
;  2045          2688  3        TES;
;  2046          2689  3        !+
;  2047          2690  3        ! Adjust for the scale factor.  The scale factor will be zero if this is
;  2048          2691  3        ! not a double precisio value.
;  2049          2692  3        !-
;  2050          2693  3        KERNEL_BLOCK [DEC_EXP] = .KERNEL_BLOCK [DEC_EXP] + .BAS_SCALE_FAC [0];
;  2051          2694  3        !+
;  2052          2695  3        ! Check to see if this number is effectively zero
;  2053          2696  3        !-
;  2054          2697  3
;  2055          2698  5        IF (((.KERNEL_BLOCK [DEC_EXP] LSS 0) AND (ABS (.KERNEL_BLOCK [DEC_EXP]) GEQ .NO_FRAC_DIGITS))   !
;  2056          2699  4            OR ((.KERNEL_BLOCK [DEC_EXP] EQL 0) AND (.NO_FRAC_DIGITS EQL 0)))   !
;  2057          2700  4            AND ((.FLAGS AND V_FLT_DEC_PT) EQL 0)       !
;  2058          2701  4           ! AND (.NO_INT_DIGITS GTR 0)
;  2059          2702  3        THEN
;  2060          2703  3        !+
;  2061          2704  3        ! If a zero value should be suppressed, then complete processing here, without
;  2062          2705  3        ! calling PLAIN_ or FANCY_ formatting routines.  All that needs to be output
;  2063          2706  3        ! is a string of blanks.
;  2064          2707  3        !-
;  2065          2708  4        BEGIN
;  2066          2709  5        IF ((.FLAGS AND V_SUPPRESS_ZERO) EQL 0)
;  2067          2710  4        THEN
;  2068          2711  5            BEGIN
;  2069          2712  5            IF .NO_INT_DIGITS GEQ 1
;  2070          2713  5            THEN
;  2071          2714  6                PUT_OUT_ZERO
;  2072          2715  5            ELSE
;  2073          2716  5                !+
;  2074          2717  5                ! Kludge for NUM1$.  NUM1$ always passes 0 for no_int_digits
;  2075          2718  5                ! and no_frac_digits, but should print "0" not ".0" for
;  2076          2719  5                ! NUM1$(0).  So make sure 0 looks like an integer to the
;  2077          2720  5                ! formatting routine.
;  2078          2721  5                !-
;  2079          2722  6                BEGIN
;  2080          2723  7                IF (.NO_INT_DIGITS EQL 0 AND .NO_FRAC_DIGITS EQL 0)
```

```
;  2081          2724  6                 THEN
;  2082          2725  7                     PUT_OUT_ZERO
;  2083          2726  6                 ELSE
;  2084          2727  7                     PUT_OUT_ZERO_FRACT
;  2085          2728  6                 END
;  2086          2729  5             END
;  2087          2730  4         ELSE
;  2088          2731  5             BEGIN
;  2089          2732  5             LOCAL BLANKS_STR_DESC : BLOCK [8, BYTE];
;  2090          2733  5
;  2091          2734  5             INITIALIZE_DESC;
;  2092          2735  5             BLANKS_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  2093          2736  5             BLANKS_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  2094          2737  5             BLANKS_STR_DESC [DSC$A_POINTER] = BLANKS;
;  2095          2738  5             BLANKS_STR_DESC [DSC$W_LENGTH] = .NO_INT_DIGITS + .NO_FRAC_DIGITS
;  2096          2739  6                 + (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN 1 ELSE 0)
;  2097          2740  6                 + (IF (.FLAGS AND V_COMMA) NEQ 0
;  2098          2741  6                     THEN (.KERNEL_BLOCK [DEC_EXP]-1)/3 * .DIGIT_SEP [DSC$W_LENGTH]
;  2099          2742  6                     ELSE 0)
;  2100          2743  6                 + (IF (.FLAGS AND V_PERIOD) NEQ 0 THEN .RADIX_PT [DSC$W_LENGTH]
;  2101          2744  5                     ELSE 0);
;  2102          2745  5             STR$CONCAT (BAS_OUT_STR [0], BLANKS_STR_DESC);
;  2103          2746  5             RETURN 1;
;  2104          2747  4             END;
;  2105          2748  4         END
;  2106          2749  4
;  2107          2750  3         ELSE
;  2108          2751  3 !+
;  2109          2752  3 ! PRINT USING specific error checking.  Not performed for floating dec. pt.
;  2110          2753  3 ! Error checking.  If this is a negative number with '**' or '$$' then it must
;  2111          2754  3 ! also have trailing minus or a trailing <CD>.
;  2112          2755  3 !-
;  2113          2756  3
;  2114          2757  4             IF ((.FLAGS AND V_FLT_DEC_PT) EQL 0)
;  2115          2758  3             THEN
;  2116          2759  3
;  2117          2760  3                 !+
;  2118          2761  3                 ! this IF statement reads: if the number is less than zero,
;  2119          2762  3                 ! and this IS an asterisk or a currency field, and the field
;  2120          2763  3                 ! DOESN'T end with either a minus sign or a <CD>, then it's
;  2121          2764  3                 ! a print-using format error.
;  2122          2765  3                 !-
;  2123          2766  4                 IF (
;  2124          2767  4                     (.KERNEL_BLOCK [SIGN] LSS 0) AND
;  2125          2768  5                     (
;  2126          2769  6                      (
;  2127          2770  6                       (.FLAGS AND V_STAR) NEQ 0
;  2128          2771  5                      ) OR
;  2129          2772  6                      (
;  2130          2773  6                       (.FLAGS AND V_CURRENCY) NEQ 0
;  2131          2774  6                      )
;  2132          2775  4                     ) AND
;  2133          2776  5                     (
;  2134          2777  5                      (.FLAGS AND V_TRAILING_SIGN) EQL 0          AND
;  2135          2778  5                      (.FLAGS AND V_CR_DR) EQL 0
;  2136          2779  5                     )
;  2137          2780  4                    )
```

```
;2138     2781  3                THEN
;2139     2782  3                    BAS$$STOP (BAS$K_PRIUSIFOR);
;2140     2783  3
;2141     2784  4            BEGIN
;2142     2785  4
;2143     2786  4 !+
;2144     2787  4 ! Find the last non-zero byte
;2145     2788  4 !-
;2146     2789  4            TEMP_PTR = .KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET] +
;2147     2790  5            BEGIN
;2148     2791  5
;2149     2792  6            IF ((.FLAGS AND V_FLT_DEC_PT) EQL 0)
;2150     2793  5            THEN
;2151     2794  5                MIN (.SIG_DIG,
;2152     2795  5                    .KERNEL_BLOCK [DEC_EXP] + .NO_FRAC_DIGITS)
;2153     2796  5            ELSE
;2154     2797  5                .SIG_DIG
;2155     2798  5
;2156     2799  5            END
;2157     2800  4            - 1;
;2158     2801  4
;2159     2802  5            IF (.TEMP_PTR GTRA (.KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET]))
;2160     2803  4            THEN
;2161     2804  4
;2162     2805  6                WHILE (.TEMP_PTR GEQA (.KERNEL_BLOCK [STRING_ADDR] +
;2163     2806  4                       .KERNEL_BLOCK [OFFSET])) AND
;2164     2807  4                      .TEMP_PTR [0] EQL %C'0'
;2165     2808  4                DO TEMP_PTR = .TEMP_PTR - 1;
;2166     2809  4
;2167     2810  4                NO_DIGITS = .TEMP_PTR - (.KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET]) + 1;
;2168     2811  3                END;
;2169     2812  3
;2170     2813  2            END;
;2171     2814  2
;2172     2815  2 !+
;2173     2816  2 ! Check to see if this is the floating decimal point situation.  The only other
;2174     2817  2 ! valid flag bit is V_STRIP_SPACES.  All other bits are ignored.
;2175     2818  2 !-
;2176     2819  2        SELECTONE .DATA_TYPE OF
;2177     2820  2        SET
;2178     2821  2            [DSC$K_DTYPE_F]:
;2179     2822  2                LARGEST_EXP = K_RANGE;
;2180     2823  2            [DSC$K_DTYPE_D]:
;2181     2824  2                LARGEST_EXP = K_RANGE;
;2182     2825  2            [DSC$K_DTYPE_P]:
;2183     2826  2                LARGEST_EXP = K_P_RANGE;
;2184     2827  2            [DSC$K_DTYPE_G]:
;2185     2828  2                LARGEST_EXP = K_G_RANGE;
;2186     2829  2            [DSC$K_DTYPE_H]:
;2187     2830  2                LARGEST_EXP = K_H_RANGE;
;2188     2831  2            [OTHERWISE]:
;2189     2832  2                LIB$STOP (OTS$_FATINTERR);
;2190     2833  2        TES;
;2191     2834  2
;2192     2835  3        IF ((.FLAGS AND V_FLT_DEC_PT) NEQ 0)
;2193     2836  2        THEN
;2194     2837  2            PLAIN_F_FORM_11 (.NO_DIGITS, .FLAGS, KERNEL_BLOCK,
```

```
;2195     2838  2            BAS_OUT_STR [0], BAS_OUT_STR_LEN [0], .LARGEST_EXP)
;2196     2839  2        ELSE
;2197     2840  3            BEGIN
;2198     2841  3    !+
;2199     2842  3    ! This is a more conventional F format.  Try formatting the number as
;2200     2843  3    ! requested.
;2201     2844  3    ! Do some error checking to see if the number will fit into the field width
;2202     2845  3    ! provided.
;2203     2846  3    ! 1.) Are there enough integer digits
;2204     2847  3    !       a.) Need 1 if negative number and no trailing sign
;2205     2848  3    !       b.) Need length of currency symbol if floating currency symbol
;2206     2849  3    !       c.) enough for the commas if required
;2207     2850  3    !       d.) enough for the digits based on the exponent
;2208     2851  3    !       e.) Need 1 for a zero if magnitude of number is less than 1.
;2209     2852  3    ! 2.) Are there enough fractional digits
;2210     2853  3    !-
;2211     2854  3
;2212     2855  3            LOCAL
;2213     2856  3                INT_DIGITS_NEED;
;2214     2857  3
;2215     2858  3            IF .ARG_COUNT NEQ K_ARG_COUNT THEN LIB$STOP (OTS$_FATINTERR);
;2216     2859  3                                                    ! consistency check
;2217     2860  3            INT_DIGITS_NEED =
;2218     2861  4            BEGIN
;2219     2862  4
;2220     2863  5            IF ((.KERNEL_BLOCK [SIGN] LSS 0) AND (.FLAGS AND V_TRAILING_SIGN) EQL 0)
;2221     2864  5                                                    ! Negative numbers and trailing sign
;2222     2865  4            THEN
;2223     2866  4                1
;2224     2867  4            ELSE
;2225     2868  4                0
;2226     2869  4
;2227     2870  4            END
;2228     2871  3            +
;2229     2872  4            BEGIN
;2230     2873  4
;2231     2874  5            IF (.KERNEL_BLOCK [DEC_EXP] GTR 0)
;2232     2875  4            THEN
;2233     2876  4                .KERNEL_BLOCK [DEC_EXP]
;2234     2877  4            ELSE
;2235     2878  4                0
;2236     2879  4
;2237     2880  4            END
;2238     2881  3            +
;2239     2882  4            BEGIN
;2240     2883  4
;2241     2884  5            IF ((.FLAGS AND V_CURRENCY) NEQ 0)      ! floating currency
;2242     2885  4            THEN
;2243     2886  4                .CURRENCY [DSC$W_LENGTH]
;2244     2887  4            ELSE
;2245     2888  4                0
;2246     2889  4
;2247     2890  4            END
;2248     2891  3            +
;2249     2892  4            BEGIN
;2250     2893  4
;2251     2894  5            IF (((.FLAGS AND V_COMMA) NEQ 0) AND (.KERNEL_BLOCK [DEC_EXP] GTR 0))   ! number of commas needed
```

```
4          THEN
4              (.KERNEL_BLOCK [DEC_EXP] - 1)/3*.DIGIT_SEP [DSC$W_LENGTH]
4          ELSE
4              0
4
3          END;
3  !+
3  ! Check to see if enough integer and fraction digits have been reserved
3  !-
3
5          IF (.INT_DIGITS_NEED GTR .NO_INT_DIGITS) OR ((.KERNEL_BLOCK [DEC_EXP] LSS 0) AND (ABS (.KERNEL_BLOCK
4                  [DEC_EXP]) GTR .NO_FRAC_DIGITS))
3          THEN
3              RETURN 0;
3
3          FANCY_F_FORM_11 (.NO_INT_DIGITS, .NO_FRAC_DIGITS, .NO_DIGITS, .FLAGS, KERNEL_BLOCK, .CURRENCY,
3              .DIGIT_SEP, .RADIX_PT, BAS_OUT_STR [0], BAS_OUT_STR_LEN [0],
3              .LARGEST_EXP);
2          END;
2
2      RETURN 1;
1      END;                                      !End of COMMON_F


00000000# 005D4 P.AAJ:  .LONG   0[9]
       2E 005F8 P.AAK:  .ASCII  \.\
          005F9         .BLKB   3
       2A 005FC P.AAL:  .ASCII  \*\
       2A 005FD         .ASCII  \*\
       2A 005FE         .ASCII  \*\
       2A 005FF         .ASCII  \*\
       2A 00600         .ASCII  \*\
       2A 00601         .ASCII  \*\
       2A 00602         .ASCII  \*\
       2A 00603         .ASCII  \*\
       2A 00604         .ASCII  \*\
       2A 00605         .ASCII  \*\
       2A 00606         .ASCII  \*\
       2A 00607         .ASCII  \*\
       2A 00608         .ASCII  \*\
       2A 00609         .ASCII  \*\
       2A 0060A         .ASCII  \*\
       2A 0060B         .ASCII  \*\
       2A 0060C         .ASCII  \*\
       2A 0060D         .ASCII  \*\
       2A 0060E         .ASCII  \*\
       2A 0060F         .ASCII  \*\
       2A 00610         .ASCII  \*\
       2A 00611         .ASCII  \*\
       2A 00612         .ASCII  \*\
       2A 00613         .ASCII  \*\
       2A 00614         .ASCII  \*\
       2A 00615         .ASCII  \*\
       2A 00616         .ASCII  \*\
       2A 00617         .ASCII  \*\
       2A 00618         .ASCII  \*\
```

```
2A  00619                .ASCII  \*\                                    ;
2A  0061A                .ASCII  \*\                                    ;
2A  0061B                .ASCII  \*\                                    ;
2A  0061C                .ASCII  \*\                                    ;
2A  0061D                .ASCII  \*\                                    ;
2A  0061E                .ASCII  \*\                                    ;
2A  0061F                .ASCII  \*\                                    ;
2A  00620                .ASCII  \*\                                    ;
2A  00621                .ASCII  \*\                                    ;
    00622                .BLKB   2
2D  00624 P.AAM:         .ASCII  \-\                                    ;
    00625                .BLKB   3
20  00628 P.AAN:         .ASCII  \ \                                    ;
    00629                .BLKB   3
45  0062C P.AAO:         .ASCII  \E\                                    ;
    0062D                .BLKB   3
20  00630 P.AAP:         .ASCII  \ \                                    ;
20  00631                .ASCII  \ \                                    ;
20  00632                .ASCII  \ \                                    ;
20  00633                .ASCII  \ \                                    ;
20  00634                .ASCII  \ \                                    ;
20  00635                .ASCII  \ \                                    ;
20  00636                .ASCII  \ \                                    ;
20  00637                .ASCII  \ \                                    ;
20  00638                .ASCII  \ \                                    ;
20  00639                .ASCII  \ \                                    ;
20  0063A                .ASCII  \ \                                    ;
20  0063B                .ASCII  \ \                                    ;
20  0063C                .ASCII  \ \                                    ;
20  0063D                .ASCII  \ \                                    ;
20  0063E                .ASCII  \ \                                    ;
20  0063F                .ASCII  \ \                                    ;
20  00640                .ASCII  \ \                                    ;
20  00641                .ASCII  \ \                                    ;
20  00642                .ASCII  \ \                                    ;
20  00643                .ASCII  \ \                                    ;
20  00644                .ASCII  \ \                                    ;
20  00645                .ASCII  \ \                                    ;
20  00646                .ASCII  \ \                                    ;
20  00647                .ASCII  \ \                                    ;
20  00648                .ASCII  \ \                                    ;
20  00649                .ASCII  \ \                                    ;
20  0064A                .ASCII  \ \                                    ;
20  0064B                .ASCII  \ \                                    ;
20  0064C                .ASCII  \ \                                    ;
20  0064D                .ASCII  \ \                                    ;
20  0064E                .ASCII  \ \                                    ;
20  0064F                .ASCII  \ \                                    ;
20  00650                .ASCII  \ \                                    ;
20  00651                .ASCII  \ \                                    ;
20  00652                .ASCII  \ \                                    ;
20  00653                .ASCII  \ \                                    ;
20  00654                .ASCII  \ \                                    ;
20  00655                .ASCII  \ \                                    ;
    00656                .BLKB   2
30  00658 P.AAQ:         .ASCII  \0\                                    ;
30  00659                .ASCII  \0\                                    ;
```

```
30  0065A          .ASCII  \0\
30  0065B          .ASCII  \0\
30  0065C          .ASCII  \0\
30  0065D          .ASCII  \0\
30  0065E          .ASCII  \0\
30  0065F          .ASCII  \0\
30  00660          .ASCII  \0\
30  00661          .ASCII  \0\
30  00662          .ASCII  \0\
30  00663          .ASCII  \0\
30  00664          .ASCII  \0\
30  00665          .ASCII  \0\
30  00666          .ASCII  \0\
30  00667          .ASCII  \0\
30  00668          .ASCII  \0\
30  00669          .ASCII  \0\
30  0066A          .ASCII  \0\
30  0066B          .ASCII  \0\
30  0066C          .ASCII  \0\
30  0066D          .ASCII  \0\
30  0066E          .ASCII  \0\
30  0066F          .ASCII  \0\
30  00670          .ASCII  \0\
30  00671          .ASCII  \0\
30  00672          .ASCII  \0\
30  00673          .ASCII  \0\
30  00674          .ASCII  \0\
30  00675          .ASCII  \0\
30  00676          .ASCII  \0\
30  00677          .ASCII  \0\
30  00678          .ASCII  \0\
30  00679          .ASCII  \0\
30  0067A          .ASCII  \0\
30  0067B          .ASCII  \0\
30  0067C          .ASCII  \0\
30  0067D          .ASCII  \0\
    0067E          .BLKB   2
2E  00680 P.AAR:   .ASCII  \.\
    00681          .BLKB   3
2A  00684 P.AAS:   .ASCII  \*\
2A  00685          .ASCII  \*\
2A  00686          .ASCII  \*\
2A  00687          .ASCII  \*\
2A  00688          .ASCII  \*\
2A  00689          .ASCII  \*\
2A  0068A          .ASCII  \*\
2A  0068B          .ASCII  \*\
2A  0068C          .ASCII  \*\
2A  0068D          .ASCII  \*\
2A  0068E          .ASCII  \*\
2A  0068F          .ASCII  \*\
2A  00690          .ASCII  \*\
2A  00691          .ASCII  \*\
2A  00692          .ASCII  \*\
2A  00693          .ASCII  \*\
2A  00694          .ASCII  \*\
2A  00695          .ASCII  \*\
```

```
2A  00696                .ASCII  \*\
2A  00697                .ASCII  \*\
2A  00698                .ASCII  \*\
2A  00699                .ASCII  \*\
2A  0069A                .ASCII  \*\
2A  0069B                .ASCII  \*\
2A  0069C                .ASCII  \*\
2A  0069D                .ASCII  \*\
2A  0069E                .ASCII  \*\
2A  0069F                .ASCII  \*\
2A  006A0                .ASCII  \*\
2A  006A1                .ASCII  \*\
2A  006A2                .ASCII  \*\
2A  006A3                .ASCII  \*\
2A  006A4                .ASCII  \*\
2A  006A5                .ASCII  \*\
2A  006A6                .ASCII  \*\
2A  006A7                .ASCII  \*\
2A  006A8                .ASCII  \*\
2A  006A9                .ASCII  \*\
    006AA                .BLKB   2
2D  006AC  P.AAT:        .ASCII  \-\
    006AD                .BLKB   3
20  006B0  P.AAU:        .ASCII  \ \
    006B1                .BLKB   3
45  006B4  P.AAV:        .ASCII  \E\
    006B5                .BLKB   3
20  006B8  P.AAW:        .ASCII  \ \
20  006B9                .ASCII  \ \
20  006BA                .ASCII  \ \
20  006BB                .ASCII  \ \
20  006BC                .ASCII  \ \
20  006BD                .ASCII  \ \
20  006BE                .ASCII  \ \
20  006BF                .ASCII  \ \
20  006C0                .ASCII  \ \
20  006C1                .ASCII  \ \
20  006C2                .ASCII  \ \
20  006C3                .ASCII  \ \
20  006C4                .ASCII  \ \
20  006C5                .ASCII  \ \
20  006C6                .ASCII  \ \
20  006C7                .ASCII  \ \
20  006C8                .ASCII  \ \
20  006C9                .ASCII  \ \
20  006CA                .ASCII  \ \
20  006CB                .ASCII  \ \
20  006CC                .ASCII  \ \
20  006CD                .ASCII  \ \
20  006CE                .ASCII  \ \
20  006CF                .ASCII  \ \
20  006D0                .ASCII  \ \
20  006D1                .ASCII  \ \
20  006D2                .ASCII  \ \
20  006D3                .ASCII  \ \
20  006D4                .ASCII  \ \
20  006D5                .ASCII  \ \
```

```
20  006D6            .ASCII  \ \
20  006D7            .ASCII  \ \
20  006D8            .ASCII  \ \
20  006D9            .ASCII  \ \
20  006DA            .ASCII  \ \
20  006DB            .ASCII  \ \
20  006DC            .ASCII  \ \
20  006DD            .ASCII  \ \
    006DE            .BLKB   2
30  006E0 P.AAX:     .ASCII  \0\
30  006E1            .ASCII  \0\
30  006E2            .ASCII  \0\
30  006E3            .ASCII  \0\
30  006E4            .ASCII  \0\
30  006E5            .ASCII  \0\
30  006E6            .ASCII  \0\
30  006E7            .ASCII  \0\
30  006E8            .ASCII  \0\
30  006E9            .ASCII  \0\
30  006EA            .ASCII  \0\
30  006EB            .ASCII  \0\
30  006EC            .ASCII  \0\
30  006ED            .ASCII  \0\
30  006EE            .ASCII  \0\
30  006EF            .ASCII  \0\
30  006F0            .ASCII  \0\
30  006F1            .ASCII  \0\
30  006F2            .ASCII  \0\
30  006F3            .ASCII  \0\
30  006F4            .ASCII  \0\
30  006F5            .ASCII  \0\
30  006F6            .ASCII  \0\
30  006F7            .ASCII  \0\
30  006F8            .ASCII  \0\
30  006F9            .ASCII  \0\
30  006FA            .ASCII  \0\
30  006FB            .ASCII  \0\
30  006FC            .ASCII  \0\
30  006FD            .ASCII  \0\
30  006FE            .ASCII  \0\
30  006FF            .ASCII  \0\
30  00700            .ASCII  \0\
30  00701            .ASCII  \0\
30  00702            .ASCII  \0\
30  00703            .ASCII  \0\
30  00704            .ASCII  \0\
30  00705            .ASCII  \0\

DOT=                P.AAK
STAR=               P.AAL
MINUS=              P.AAM
BLANK=              P.AAN
E=                  P.AAO
BLANKS=             P.AAP
ZEROES=             P.AAQ
DOT=                P.AAR
STAR=               P.AAS
```

```
MINUS=                  P.AAT
BLANK=                  P.AAU
E=                      P.AAV
BLANKS=                 P.AAW
ZEROES=                 P.AAX

                        0FFC   8F   BB 00000 COMMON_F:
                                                PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>        ; 2439
                     5E FF74   CE   9E 00004            MOVAB   -140(SP), SP
                  04 AE        5B   D0 00009            MOVL    R11, RADIX_PT
                  0C AE        5A   D0 0000D            MOVL    R10, DIGIT_SEP
                               59   DD 00011            PUSHL   R9
                  04 AE        58   D0 00013            MOVL    R8, ARG_COUNT
                  24 AE        57   D0 00017            MOVL    R7, DATA_TYPE
                  14 AE        56   D0 0001B            MOVL    R6, BAS_SCALE_FAC
                  0C AE        55   D0 0001F            MOVL    R5, BAS_OUT_STR
                               54   DD 00023            PUSHL   R4
                  1C AE        52   7D 00025            MOVQ    R2, NO_FRAC_DIGITS
                  24 AE        51   D0 00029            MOVL    R1, NO_INT_DIGITS
                  2C AE        50   D0 0002D            MOVL    R0, VALUE
               3C AE FE98      CF   28 00031            MOVC3   #36, P.AAJ, KERNEL_BLOCK            ; 2520
                  4C AE     60 AE   9E 00038            MOVAB   TEMP_STRING, KERNEL_BLOCK+16        ; 2532
                  15     28 AE      D1 0003D            CMPL    DATA_TYPE, #21                      ; 2542
                                    1A 12 00041         BNEQ    1$
                  55     2C AE      04 C1 00043         ADDL3   #4, VALUE, R5                       ; 2546
     F8AA CF      0F        95   2C BE 37 00048         CMPP4   @VALUE, @(R5)+, #15, PACKED_ZERO
          54      54           02    54 DC 00050         MOVPSL  R4
                  54           02    02 EF 00052         EXTZV   #2, #2, R4, R4
                  50           01    54 C3 00057         SUBL3   R4, #1, RESULT
                                     04 11 0005B         BRB     2$                                  ; 2543
                               50 2C BE D0 0005D 1$:     MOVL    @VALUE, RESULT                      ; 2550
                                  03 13 00061 2$:        BEQL    3$                                  ; 2553
                                  0A9 31 00063           BRW     12$
                  44 20 AE         0C E0 00066 3$:        BBS     #12, FLAGS, 7$                      ; 2561
                              40 AE D4 0006B             CLRL    KERNEL_BLOCK+4                      ; 2564
                  30 AE           01 D0 0006E            MOVL    #1, NO_DIGITS
                              24 AE D5 00072             TSTL    NO_INT_DIGITS                       ; 2563
                                    07 14 00075         BGTR    4$
                                    20 12 00077         BNEQ    5$                                  ; 2574
                              1C AE D5 00079             TSTL    NO_FRAC_DIGITS
                                    1B 12 0007C         BNEQ    5$
                  44 AE             01 7D 0007E 4$:     MOVQ    #1, KERNEL_BLOCK+8                  ; 2575
                  50 AE             01 D0 00082         MOVL    #1, KERNEL_BLOCK+20
                  4C BE             30 90 00086         MOVB    #48, @KERNEL_BLOCK+16
               51 24 AE     1C AE   C1 0008A            ADDL3   NO_FRAC_DIGITS, NO_INT_DIGITS, R1
          51   30           6E      00 2C 00090         MOVC5   #0, (SP), #48, R1, @KERNEL_BLOCK+16
                              4C BE    00095
                                    13 11 00097         BRB     6$                                  ; 2573
                              44 AE 7C 00099 5$:        CLRQ    KERNEL_BLOCK+8                      ; 2577
                  50 AE             01 D0 0009C         MOVL    #1, KERNEL_BLOCK+20
                  4C BE             30 90 000A0         MOVB    #48, @KERNEL_BLOCK+16
     1C AE        30           6E      00 2C 000A4         MOVC5   #0, (SP), #48, NO_FRAC_DIGITS, -
                              4C BE    000AA                       @KERNEL_BLOCK+16
                                  0327 31 000AC 6$:      BRW     62$                                 ; 2563
                        00000000' EF   D5 000AF 7$:      TSTL    E_D                                 ; 2582
                                    24 12 000B5         BNEQ    8$
```

```
                00000000'  EF    FE63    CF  9E 000B7          MOVAB   MINUS, M_D                              ;
                00000000'  EF    FE2E    CF  9E 000C0          MOVAB   DOT, D_D                                ;
                00000000'  EF    FE55    CF  9E 000C9          MOVAB   BLANK, B_D                              ;
                00000000'  EF    FE50    CF  9E 000D2          MOVAB   E, E_D                                  ;
                      36   AE    010E    8F  B0 000DB 8$:      MOVW    #270, BLANKS_STR_DESC+2                 ; 2585
                      38   AE    FE45    CF  9E 000E1          MOVAB   BLANKS, BLANKS_STR_DESC+4               ; 2587
                           51      24    AE  D0 000E7          MOVL    NO_INT_DIGITS, R1                       ; 2588
           50              51      1C    AE  C1 000EB          ADDL3   NO_FRAC_DIGITS, R1, R0                  ;
           05        20    AE            01  E1 000F0          BBC     #1, FLAGS, 9$                           ; 2589
                           51            01  D0 000F5          MOVL    #1, R1                                  ;
                                         02  11 000F8          BRB     10$                                     ;
                                         51  D4 000FA 9$:      CLRL    R1                                      ;
                           50            51  C0 000FC 10$:     ADDL2   R1, R0                                  ;
           03        20    AE            02  E0 000FF          BBS     #2, FLAGS, 11$                          ; 2590
                                     0244  31 00104          BRW     51$                                     ;
           51        44    AE            01  C3 00107 11$:     SUBL3   #1, KERNEL_BLOCK+8, R1                  ; 2591
                                     0230  31 0010C          BRW     50$                                     ;
                           0A      28    AE  D1 0010F 12$:     CMPL    DATA_TYPE, #10                          ; 2604
                                         05  12 00113          BNEQ    13$                                     ;
                           59            06  D0 00115          MOVL    #6, SIG_DIG                             ; 2605
                                         45  11 00118          BRB     18$                                     ;
                           0B      28    AE  D1 0011A 13$:     CMPL    DATA_TYPE, #11                          ; 2606
                                         05  12 0011E          BNEQ    14$                                     ;
                           59            10  D0 00120          MOVL    #16, SIG_DIG                            ; 2607
                                         3A  11 00123          BRB     18$                                     ;
                           15      28    AE  D1 00125 14$:     CMPL    DATA_TYPE, #21                          ; 2608
                                         11  12 00129          BNEQ    15$                                     ;
                           50      2C    BE  3C 0012B          MOVZWL  @VALUE, R0                              ; 2613
           51        2C    AE            08  C1 0012F          ADDL3   #8, VALUE, R1                           ;
                           59            61  98 00134          CVTBL   (R1), SIG_DIG                           ;
                           59            50  C0 00137          ADDL2   R0, SIG_DIG                             ;
                                         23  11 0013A          BRB     18$                                     ; 2602
                           1B      28    AE  D1 0013C 15$:     CMPL    DATA_TYPE, #27                          ; 2615
                                         05  12 00140          BNEQ    16$                                     ;
                           59            0F  D0 00142          MOVL    #15, SIG_DIG                            ; 2616
                                         18  11 00145          BRB     18$                                     ;
                           1C      28    AE  D1 00147 16$:     CMPL    DATA_TYPE, #28                          ; 2617
                                         05  12 0014B          BNEQ    17$                                     ;
                           59            21  D0 0014D          MOVL    #33, SIG_DIG                            ; 2618
                                         0D  11 00150          BRB     18$                                     ;
                                00000000G  8F  DD 00152 17$:     PUSHL   #OTS$_FATINTERR                         ; 2620
                00000000G  00            01  FB 00158          CALLS   #1, LIB$STOP                            ;
                                         5A  D4 0015F 18$:     CLRL    R10                                     ; 2623
           24        20    AE            05  E0 00161          BBS     #5, FLAGS, 21$                          ;
                                         5A  D6 00166          INCL    R10                                     ;
                           51      24    AE  D0 00168          MOVL    NO_INT_DIGITS, R1                       ;
           50              51      1C    AE  C1 0016C          ADDL3   NO_FRAC_DIGITS, R1, R0                  ;
                           2C      BE        B5 00171          TSTW    @VALUE                                  ; 2626
                                         0A  18 00174          BGEQ    19$                                     ;
           05        20    AE            01  E0 00176          BBS     #1, FLAGS, 19$                          ;
                           51            01  D0 0017B          MOVL    #1, R1                                  ;
                                         02  11 0017E          BRB     20$                                     ;
                                         51  D4 00180 19$:     CLRL    R1                                      ;
                           50            51  C2 00182 20$:     SUBL2   R1, R0                                  ; 2624
                           59            50  D1 00185          CMPL    R0, SIG_DIG                             ; 2629
                                         03  15 00188          BLEQ    22$                                     ;
                           50            59  D0 0018A 21$:     MOVL    SIG_DIG, R0                             ;
```

```
                        50   AE              50   D0 0018D 22$:   MOVL    R0, KERNEL_BLOCK+20                                     ; 2623
                        09                   5A   E9 00191       BLBC    R10, 23$                                                ; 2633
                        50 02000000          8F   D0 00194       MOVL    #33554432, R0
                                             02   11 0019B       BRB     24$
                                             50   D4 0019D 23$:   CLRL    R0
                        54   AE              50   D0 0019F 24$:   MOVL    R0, KERNEL_BLOCK+24
                3C  AE  18   AE      1C  AE  C1 001A3            ADDL3   NO_FRAC_DIGITS, BAS_SCALE_FAC, KERNEL_BLOCK            ; 2637
                        0A   28      AE  D1 001AA                CMPL    DATA_TYPE, #10                                          ; 2640
                                             06   13 001AE       BEQL    25$
                        0B   28      AE  D1 001B0                CMPL    DATA_TYPE, #11                                          ; 2642
                                             10   12 001B4       BNEQ    26$
                        51   60      AE  9E 001B6 25$:           MOVAB   KERNEL_BLOCK+36, R1                                     ; 2643
                        50   2C      AE  D0 001BA                MOVL    VALUE, R0
                           00000000G         00   16 001BE       JSB     OTS$$CVT_D_T_R8
                                             71   11 001C4       BRB     34$
                        15   28      AE  D1 001C6 26$:           CMPL    DATA_TYPE, #21                                          ; 2644
                                             6D   12 001CA       BNEQ    35$
                                             54   D4 001CC       CLRL    ZERO_CTR                                                ; 2645
                    55  2C   AE              04   C1 001CE       ADDL3   #4, VALUE, R5                                           ; 2657
    60  AE          2C   BE      95      2C  BE  08 001D3        CVTPS   @VALUE, @(R5)+, @VALUE, TEMP_STRING
                        50   2C      BE  3C 001DB                MOVZWL  @VALUE, R0                                              ; 2662
                    52  2C   AE              08   C1 001DF       ADDL3   #8, VALUE, R2
                        51                   62   98 001E4       CVTBL   (R2), INT_DIGITS
                        51                   50   C0 001E7       ADDL2   R0, INT_DIGITS
                                             50   D4 001EA       CLRL    I                                                       ; 2663
                                             09   11 001EC       BRB     28$
                        30       60 AE40     91   1E 001EE 27$:  CMPB    TEMP_STRING[I], #48                                     ; 2664
                                             06   12 001F3       BNEQ    29$
                                             54   D6 001F5       INCL    ZERO_CTR
                F3              50           51   F3 001F7 28$:   AOBLEQ  INT_DIGITS, I, 27$
44  AE                  51                   54   C3 001FB 29$:   SUBL3   ZERO_CTR, INT_DIGITS, KERNEL_BLOCK+8                   ; 2666
                    40  AE      01      A4  9E 00200            MOVAB   1(R4), KERNEL_BLOCK+4                                   ; 2667
                51  2C   AE              08   C1 00205       ADDL3   #8, VALUE, R1                                           ; 2671
                        50                   61   98 0020A       CVTBL   (R1), R0
                                             03   18 0020D       BGEQ    30$
                        50                   50   CE 0020F       MNEGL   R0, R0
                59              50      44  AE  C1 00212 30$:   ADDL3   KERNEL_BLOCK+8, R0, SIG_DIG
                        2D      60      AE  91 00217            CMPB    TEMP_STRING, #45                                        ; 2672
                                             16   12 0021B       BNEQ    33$
                    48  AE                   01   CE 0021D       MNEGL   #1, KERNEL_BLOCK+12                                     ; 2675
                05      20  AE              01   E0 00221       BBS     #1, FLAGS, 31$                                          ; 2677
                        50                   01   D0 00226       MOVL    #1, R0
                                             02   11 00229       BRB     32$
                                             50   D4 0022B 31$:   CLRL    R0
                    50  AE                   50   C2 0022D 32$:   SUBL2   R0, KERNEL_BLOCK+20
                                             3F   11 00231       BRB     38$                                                     ; 2672
                    48  AE                   01   D0 00233 33$:   MOVL    #1, KERNEL_BLOCK+12                                     ; 2680
                                             39   11 00237 34$:   BRB     38$                                                     ; 2638
                        1B      28      AE  D1 00239 35$:           CMPL    DATA_TYPE, #27                                          ; 2682
                                             10   12 0023D       BNEQ    36$
                        51      60      AE  9E 0023F            MOVAB   KERNEL_BLOCK+36, R1                                     ; 2683
                        50      2C      AE  D0 00243            MOVL    VALUE, R0
                           00000000G         00   16 00247       JSB     OTS$$CVT_G_T_R8
                                             23   11 0024D       BRB     38$
                        1C      28      AE  D1 0024F 36$:           CMPL    DATA_TYPE, #28                                          ; 2684
                                             10   12 00253       BNEQ    37$
                        51      60      AE  9E 00255            MOVAB   KERNEL_BLOCK+36, R1                                     ; 2685
```

```
                        50        2C   AE   D0 00259          MOVL    VALUE, R0                                       ;
                 00000000G        00   16 0025D              JSB     OTS$$CVT_H_T_R8                                 ;
                                  0D   11 00263              BRB     38$                                             ;
                 00000000G        8F   DD 00265 37$:        PUSHL   #OTS$_FATINTERR                                 ; 2687
          00000000G      00             01   FB 0026B              CALLS   #1, LIB$STOP                            ;
                        50        18   AE   98 00272 38$:        CVTBL   BAS_SCALE_FAC, R0                       ; 2693
                  44    AE             50   C0 00276              ADDL2   R0, KERNEL_BLOCK+8                      ;
                        56        44   AE   D0 0027A              MOVL    KERNEL_BLOCK+8, R6                      ; 2698
                                       0E   18 0027E              BGEQ    40$                                     ;
                        50             56   D0 00280              MOVL    R6, R0                                  ;
                                       03   18 00283              BGEQ    39$                                     ;
                        50             50   CE 00285              MNEGL   R0, R0                                  ;
                  1C    AE             50   D1 00288 39$:        CMPL    R0, NO_FRAC_DIGITS                      ;
                                       0C   18 0028C              BGEQ    42$                                     ;
                                       56   D5 0028E 40$:        TSTL    R6                                      ; 2699
                                       03   12 00290              BNEQ    41$                                     ;
                                 1C    AE   D5 00292              TSTL    NO_FRAC_DIGITS                          ;
                                       03   13 00295 41$:        BEQL    42$                                     ;
                                    00D8   31 00297              BRW     55$                                     ;
                        46                 5A   E9 0029A 42$:        BLBC    R10, 45$                                ; 2700
             44         20    AE           0C   E0 0029D              BBS     #12, FLAGS, 46$                         ; 2709
                                 40    AE   D4 002A2              CLRL    KERNEL_BLOCK+4                          ; 2713
                  30    AE             01   D0 002A5              MOVL    #1, NO_DIGITS                           ;
                                 24    AE   D5 002A9              TSTL    NO_INT_DIGITS                           ; 2712
                                       07   14 002AC              BGTR    43$                                     ;
                                       20   12 002AE              BNEQ    44$                                     ; 2723
                                 1C    AE   D5 002B0              TSTL    NO_FRAC_DIGITS                          ;
                                       1B   12 002B3              BNEQ    44$                                     ;
                  44    AE             01   7D 002B5 43$:        MOVQ    #1, KERNEL_BLOCK+8                      ; 2724
                  50    AE             01   D0 002B9              MOVL    #1, KERNEL_BLOCK+20                     ;
                  4C    BE             30   90 002BD              MOVB    #48, @KERNEL_BLOCK+16                   ;
            51    24    AE       1C    AE   C1 002C1              ADDL3   NO_FRAC_DIGITS, NO_INT_DIGITS, R1       ;
      51          30             6E   00   2C 002C7              MOVC5   #0, (SP), #48, R1, @KERNEL_BLOCK+16     ;
                                 4C    BE        002CC                                                            ;
                                       13   11 002CE              BRB     45$                                     ; 2722
                                 44    AE   7C 002D0 44$:        CLRQ    KERNEL_BLOCK+8                          ; 2726
                  50    AE             01   D0 002D3              MOVL    #1, KERNEL_BLOCK+20                     ;
                  4C    BE             30   90 002D7              MOVB    #48, @KERNEL_BLOCK+16                   ;
1C  AE            30             6E   00   2C 002DB              MOVC5   #0, (SP), #48, NO_FRAC_DIGITS, -        ;
                                 4C    BE        002E1                    @KERNEL_BLOCK+16                         ;
                                    00B3   31 002E3 45$:        BRW     57$                                     ; 2711
                              00000000'  EF   D5 002E6 46$:        TSTL    E_D                                     ; 2732
                                       24   12 002EC              BNEQ    47$                                     ;
                 00000000'  EF       FCB4   CF   9E 002EE              MOVAB   MINUS, M_D                              ;
                 00000000'  EF       FC7F   CF   9E 002F7              MOVAB   DOT, D_D                                ;
                 00000000'  EF       FCA6   CF   9E 00300              MOVAB   BLANK, B_D                              ;
                 00000000'  EF       FCA1   CF   9E 00309              MOVAB   E, E_D                                  ;
                        36    AE     010E   8F   B0 00312 47$:        MOVW    #270, BLANKS_STR_DESC+2                 ; 2735
                        38    AE     FC96   CF   9E 00318              MOVAB   BLANKS, BLANKS_STR_DESC+4               ; 2737
                              51       24   AE   D0 0031E              MOVL    NO_INT_DIGITS, R1                       ; 2738
                        50    51       1C   AE   C1 00322              ADDL3   NO_FRAC_DIGITS, R1, R0                  ;
                        05          20   AE   E1 00327              BBC     #1, FLAGS, 48$                          ; 2739
                                    51        01   D0 0032C              MOVL    #1, R1                                  ;
                                              02   11 0032F              BRB     49$                                     ;
                                              51   D4 00331 48$:        CLRL    R1                                      ;
                                    50        51   C0 00333 49$:        ADDL2   R1, R0                                  ;
                        10          20   AE        02   E1 00336              BBC     #2, FLAGS, 51$                          ; 2740
```

```
                                51      FF    A6  9E 0033B          MOVAB   -1(R6), R1                              ; 2741
                                51            03  C6 0033F 50$:     DIVL2   #3, R1
                                52      14    BE  3C 00342          MOVZWL  @DIGIT_SEP, R2
                                51            52  C4 00346          MULL2   R2, R1
                                              02  11 00349          BRB     52$
                                              51  D4 0034B 51$:     CLRL    R1                                  ; 2740
                                50            51  C0 0034D 52$:     ADDL2   R1, R0
                          06        20  AE        06  E1 00350          BBC     #6, FLAGS, 53$                      ; 2743
                                5B      0C    BE  3C 00355          MOVZWL  @RADIX_PT, R11
                                              02  11 00359          BRB     54$
                                              5B  D4 0035B 53$:     CLRL    R11
                    34  AE                50      5B  A1 0035D 54$:     ADDW3   R11, R0, BLANKS_STR_DESC
                                                34  AE  9F 00362          PUSHAB  BLANKS_STR_DESC                 ; 2745
                                                14  AE  DD 00365          PUSHL   BAS_OUT_STR
                              00000000G     00        02  FB 00368          CALLS   #2, STR$CONCAT
                                                  0169  31 0036F          BRW     81$                             ; 2746
                                24                  5A  E9 00372 55$:     BLBC    R10, 57$                        ; 2757
                                                48  AE  D5 00375          TSTL    KERNEL_BLOCK+12                 ; 2767
                                                    1F  18 00378          BGEQ    57$
                          05        20  AE            04  E0 0037A          BBS     #4, FLAGS, 56$                  ; 2770
                          15        20  AE            03  E1 0037F          BBC     #3, FLAGS, 57$                  ; 2773
                          10        20  AE            01  E0 00384 56$:     BBS     #1, FLAGS, 57$                  ; 2777
                          0B        20  AE            0E  E0 00389          BBS     #14, FLAGS, 57$                 ; 2778
                                7E      00G   8F        9A 0038E          MOVZBL  #BAS$K_PRIUSIFOR, -(SP)         ; 2782
                              00000000G     00        01  FB 00392          CALLS   #1, BAS$$STOP
                          51        4C  AE    40  AE  C1 00399 57$:     ADDL3   KERNEL_BLOCK+4, KERNEL_BLOCK+16, R1 ; 2789
                                14                  5A  E9 0039F          BLBC    R10, 59$                        ; 2794
                          52        44  AE    1C  AE  C1 003A2          ADDL3   NO_FRAC_DIGITS, KERNEL_BLOCK+8, R2  ; 2795
                                50                  59  D0 003A8          MOVL    SIG_DIG, R0
                                52                  50  D1 003AB          CMPL    R0, R2
                                                    03  15 003AE          BLEQ    58$
                                50                  52  D0 003B0          MOVL    R2, R0
                                59                  50  D0 003B3 58$:     MOVL    R0, R9                          ; 2794
                                50        FF  A941  9E  00  3B6 59$:     MOVAB   -1(R9)[R1], TEMP_PTR            ; 2800
                                51                  50  D1 003BB          CMPL    TEMP_PTR, R1                    ; 2802
                                                    0E  1B 003BE          BLEQU   61$
                                51                  50  D1 003C0 60$:     CMPL    TEMP_PTR, R1                    ; 2805
                                                    09  1F 003C3          BLSSU   61$
                                30                  60  91 003C5          CMPB    (TEMP_PTR), #48                 ; 2807
                                                    04  12 003C8          BNEQ    61$
                                                    50  D7 003CA          DECL    TEMP_PTR                        ; 2808
                                                    F2  11 003CC          BRB     60$
                                50                  51  C2 003CE 61$:     SUBL2   R1, R0                          ; 2810
                    30  AE          01        A0  9E 003D1          MOVAB   1(R0), NO_DIGITS
                                0A      28  AE    D1 003D6 62$:     CMPL    DATA_TYPE, #10                  ; 2821
                                                  06  13 003DA          BEQL    63$
                                0B      28  AE    D1 003DC          CMPL    DATA_TYPE, #11                  ; 2823
                                                  05  12 003E0          BNEQ    64$
                                5A              26  D0 003E2 63$:     MOVL    #38, LARGEST_EXP                ; 2824
                                                2D  11 003E5          BRB     68$
                                15      28  AE    D1 003E7 64$:     CMPL    DATA_TYPE, #21                  ; 2825
                                                  13  13 003EB          BEQL    66$
                                1B      28  AE    D1 003ED          CMPL    DATA_TYPE, #27                  ; 2827
                                                  07  12 003F1          BNEQ    65$
                                5A    0134  8F    3C 003F3          MOVZWL  #308, LARGEST_EXP               ; 2828
                                                  1A  11 003F8          BRB     68$
                                1C      28  AE    D1 003FA 65$:     CMPL    DATA_TYPE, #28                  ; 2829
```

```
                               07  12 003FE        BNEQ    67$                                    ;
                5A      1344   8F  3C 00400 66$:   MOVZWL  #4932, LARGEST_EXP                     ; 2830
                               0D  11 00405        BRB     68$                                    ;
                   00000000G   8F  DD 00407 67$:   PUSHL   #OTS$_FATINTERR                        ; 2832
      00000000G 00             01  FB 0040D        CALLS   #1, LIB$STOP                           ;
1C        20    AE             05  E1 00414 68$:   BBC     #5, FLAGS, 69$                         ; 2835
                52        3C   AE  9E 00419        MOVAB   KERNEL_BLOCK, R2                       ; 2837
                55             5A  D0 0041D        MOVL    LARGEST_EXP, R5                        ; 2838
                54             6E  D0 00420        MOVL    BAS_OUT_STR_LEN, R4                    ;
                53        10   AE  D0 00423        MOVL    BAS_OUT_STR, R3                        ;
                51        20   AE  D0 00427        MOVL    FLAGS, R1                              ;
                50        30   AE  D0 0042B        MOVL    NO_DIGITS, R0                          ;
                             0000V 30 0042F        BSBW    PLAIN_F_FORM_11                        ;
                             00A6  31 00432        BRW     81$                                    ;
                0C        08   AE  D1 00435 69$:   CMPL    ARG_COUNT, #12                         ; 2858
                               0D  13 00439        BEQL    70$                                    ;
                   00000000G   8F  DD 0043B        PUSHL   #OTS$_FATINTERR                        ;
      00000000G 00             01  FB 00441        CALLS   #1, LIB$STOP                           ;
                          48   AE  D5 00448 70$:   TSTL    KERNEL_BLOCK+12                        ; 2863
                               0A  18 0044B        BGEQ    71$                                    ;
05        20    AE             01  E0 0044D        BBS     #1, FLAGS, 71$                         ;
                50             01  D0 00452        MOVL    #1, R0                                 ;
                               02  11 00455        BRB     72$                                    ;
                               50  D4 00457 71$:   CLRL    R0                                     ;
                51        44   AE  D0 00459 72$:   MOVL    KERNEL_BLOCK+8, R1                     ; 2874
                               53  D4 0045D        CLRL    R3                                     ;
                               51  D5 0045F        TSTL    R1                                     ;
                               07  15 00461        BLEQ    73$                                    ;
                               53  D6 00463        INCL    R3                                     ;
                52             51  D0 00465        MOVL    R1, R2                                 ; 2876
                               02  11 00468        BRB     74$                                    ;
                               52  D4 0046A 73$:   CLRL    R2                                     ; 2874
                50             52  C0 0046C 74$:   ADDL2   R2, R0                                 ; 2871
06        20    AE             03  E1 0046F        BBC     #3, FLAGS, 75$                         ; 2884
                59        04   BE  3C 00474        MOVZWL  @CURRENCY, R9                          ; 2886
                               02  11 00478        BRB     76$                                    ;
                               59  D4 0047A 75$:   CLRL    R9                                     ; 2884
52              50             59  C1 0047C 76$:   ADDL3   R9, R0, R2                             ; 2881
13        20    AE             02  E1 00480        BBC     #2, FLAGS, 77$                         ; 2894
                10             53  E9 00485        BLBC    R3, 77$                                ;
                50        FF   A1  9E 00488        MOVAB   -1(R1), R0                             ; 2896
                50             03  C6 0048C        DIVL2   #3, R0                                 ;
                53        14   BE  3C 0048F        MOVZWL  @DIGIT_SEP, R3                         ;
                50             53  C4 00493        MULL2   R3, R0                                 ;
                               02  11 00496        BRB     78$                                    ;
                               50  D4 00498 77$:   CLRL    R0                                     ; 2894
                50             52  C0 0049A 78$:   ADDL2   R2, INT_DIGITS_NEED                    ; 2892
          24    AE             50  D1 0049D        CMPL    INT_DIGITS_NEED, NO_INT_DIGITS         ; 2905
                               3D  14 004A1        BGTR    82$                                    ;
                               51  D5 004A3        TSTL    R1                                     ;
                               0E  18 004A5        BGEQ    80$                                    ;
                50             51  D0 004A7        MOVL    R1, R0                                 ;
                               03  18 004AA        BGEQ    79$                                    ;
                50             50  CE 004AC        MNEGL   R0, R0                                 ;
          1C    AE             50  D1 004AF 79$:   CMPL    R0, NO_FRAC_DIGITS                     ; 2906
                               2B  14 004B3        BGTR    82$                                    ;
                54        3C   AE  9E 004B5 80$:   MOVAB   KERNEL_BLOCK, R4                       ; 2910
```

```
                     59              6E  D0 004B9          MOVL    BAS_OUT_STR_LEN, R9                             ; 2911
                     57        0C    AE  7D 004BC          MOVQ    RADIX_PT, R7                                    ;
                     56        14    AE  D0 004C0          MOVL    DIGIT_SEP, R6                                   ;
                     55        04    AE  D0 004C4          MOVL    CURRENCY, R5                                    ;
                     53        20    AE  D0 004C8          MOVL    FLAGS, R3                                       ;
                     52        30    AE  D0 004CC          MOVL    NO_DIGITS, R2                                   ;
                     51        1C    AE  D0 004D0          MOVL    NO_FRAC_DIGITS, R1                              ;
                     50        24    AE  D0 004D4          MOVL    NO_INT_DIGITS, R0                               ;
                                 0000V 30 004D8          BSBW    FANCY_F_FORM_11                                 ;
                     50              01  D0 004DB 81$:     MOVL    #1, R0                                          ; 2915
                                     02  11 004DE          BRB     83$                                             ;
                                     50  D4 004E0 82$:     CLRL    R0                                              ; 2916
                     5E        0094  CE  9E 004E2 83$:     MOVAB   148(SP), SP                                     ;
                               0FFC  8F  BA 004E7          POPR    #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>            ;
                                     05  004EB          RSB                                                     ;

; Routine Size:  1260 bytes,    Routine Base:  _BAS$CODE + 0706
```

```
;  2275          2917  1 ROUTINE COMMON_G (                                    !
;  2276          2918  1          VALUE,                                       !
;  2277          2919  1          FLAGS,                                       !
;  2278          2920  1          BAS_OUT_STR_LEN,                             !
;  2279          2921  1          BAS_OUT_STR,                                 !
;  2280          2922  1          BAS_SCALE_FAC,                               !
;  2281          2923  1          NUM_DIGITS,                                  !
;  2282          2924  1          DATA_TYPE                                    !
;  2283          2925  1      ): JSB_G_FORMAT =                                !
;  2284          2926  1
;  2285          2927  1  !++
;  2286          2928  1  ! FUNCTIONAL DESCRIPTION:
;  2287          2929  1  !
;  2288          2930  1  !       This routine converts a number to either E or F format
;  2289          2931  1  !       depending on its value.  This routine supports PRINT, PRINT USING, NUM$,
;  2290          2932  1  !       and STR$.  If a number can be represented in six digits without any loss
;  2291          2933  1  !       of accuracy then it is returned in F format.  Otherwise, it is returned
;  2292          2934  1  !       in E format.  If the optional argument NUM_DIGITS is included, then the
;  2293          2935  1  !       number is returned in F format if it can be represented no less accur-
;  2294          2936  1  !       ately in NUM_DIGITS.
;  2295          2937  1  !
;  2296          2938  1  ! FORMAL PARAMETERS:
;  2297          2939  1  !
;  2298          2940  1  !       VALUE.rz.r              the value to convert
;  2299          2941  1  !       FLAGS.rlu.v             = 1, strip leading and trailing space
;  2300          2942  1  !       BAS_OUT_STR_LEN.ww.r    length of the string returned.  Useful if return
;  2301          2943  1  !                               string is static.
;  2302          2944  1  !       BAS_OUT_STR.wt.dx       return string
;  2303          2945  1  !       BAS_SCALE_FAC.rb.v      the scale factor for double precision values.
;  2304          2946  1  !       NUM_DIGITS.rl.v         number of digits in F format.
;  2305          2947  1  !       DATA_TYPE.rl.v          data type of element to format
;  2306          2948  1  !
;  2307          2949  1  ! IMPLICIT INPUTS:
;  2308          2950  1  !
;  2309          2951  1  !       NONE
;  2310          2952  1  !
;  2311          2953  1  ! IMPLICIT OUTPUTS:
;  2312          2954  1  !
;  2313          2955  1  !       NONE
;  2314          2956  1  !
;  2315          2957  1  ! COMPLETION CODES:
;  2316          2958  1  !
;  2317          2959  1  !       success         the value could be converted as required
;  2318          2960  1  !       failure         the value did not fit in the field described
;  2319          2961  1  !
;  2320          2962  1  ! SIDE EFFECTS:
;  2321          2963  1  !
;  2322          2964  1  !       NONE
;  2323          2965  1  !
;  2324          2966  1  !--
;  2325          2967  1
;  2326          2968  2      BEGIN
;  2327          2969  2
;  2328          2970  2      MAP
;  2329          2971  2          BAS_SCALE_FAC : VECTOR [1, BYTE, SIGNED],
;  2330          2972  2          BAS_OUT_STR_LEN : REF VECTOR,
;  2331          2973  2          VALUE : REF VECTOR,
```

```
;  2332     2974  2          BAS_OUT_STR : REF VECTOR [2, LONG];
;  2333     2975  2
;  2334     2976  2      LOCAL
;  2335     2977  2          RANGE,                                      ! maximum number of digits for F format
;  2336     2978  2          TEMP_STRING : VECTOR [K_TMP_STR_LEN_H, BYTE],   ! temp string to pass to the kernel
;  2337     2979  2                                                      ! conversion routine
;  2338     2980  2          TEMP_PTR : REF VECTOR [, BYTE],             ! pointer into string returned by
;  2339     2981  2                                                      ! kernel conversion routine
;  2340     2982  2          NO_DIGITS,                                  ! actual length of string returned
;  2341     2983  2                                                      ! by the kernel routine
;  2342     2984  2          STATUS,                                     ! return status
;  2343     2985  2          KERNEL_BLOCK : BLOCK [K_KERNEL_BLK_SZ, BYTE]
;  2344     2986  2                         INITIAL (REP (K_KERNEL_BLK_SZ/4) OF (0)),
;  2345     2987  2          LARGEST_EXP,
;  2346     2988  2          RESULT;
;  2347     2989  2
;  2348     2990  2      !+
;  2349     2991  2      ! Special case 0
;  2350     2992  2      !-
;  2351     2993  2
;  2352     2994  2      SELECTONE .DATA_TYPE OF
;  2353     2995  2          SET
;  2354     2996  2              [DSC$K_DTYPE_P]:
;  2355     2997  3                  BEGIN
;  2356     2998  3                  MAP
;  2357     2999  3                      VALUE : REF BLOCK [12,BYTE];
;  2358     3000  3                  RESULT = CMPP (VALUE [DSC$W_LENGTH], .VALUE [DSC$A_POINTER],
;  2359     3001  3                          %REF (15), PACKED_ZERO);
;  2360     3002  2                  END;
;  2361     3003  2              [OTHERWISE]:
;  2362     3004  2                  RESULT = .VALUE [0];
;  2363     3005  2          TES;
;  2364     3006  2
;  2365     3007  2      IF .RESULT EQL 0
;  2366     3008  2      THEN
;  2367     3009  3          BEGIN
;  2368     3010  3
;  2369     3011  3          BIND
;  2370     3012  3              BLANK = UPLIT (BYTE (' ')),
;  2371     3013  3              ZERO = UPLIT (BYTE ('0'));
;  2372     3014  3
;  2373     3015  3          LOCAL
;  2374     3016  3              ZERO_STR_DESC : BLOCK [8, BYTE];
;  2375     3017  3
;  2376     3018  3      !+
;  2377     3019  3      ! Initialize the BLANK desc in case this is the very first PRINT.
;  2378     3020  3      !-
;  2379     3021  3          B_D = BLANK;
;  2380     3022  3          ZERO_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  2381     3023  3          ZERO_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  2382     3024  3          ZERO_STR_DESC [DSC$W_LENGTH] = 1;
;  2383     3025  3          ZERO_STR_DESC [DSC$A_POINTER] = ZERO;
;  2384     3026  3          STATUS = STR$CONCAT (BAS_OUT_STR [0],
;  2385     3027  3              (IF .FLAGS AND V_STRIP_SPACES THEN NULL_DESC ELSE BLANK_DESC), ZERO_STR_DESC,
;  2386     3028  3              (IF .FLAGS AND V_STRIP_SPACES THEN NULL_DESC ELSE BLANK_DESC));
;  2387     3029  3          BAS_OUT_STR_LEN [0] = (IF .FLAGS AND V_STRIP_SPACES THEN 1 ELSE 3);
;  2388     3030  3          RETURN .STATUS;
```

```
;  2389          3031  2          END;
;  2390          3032  2
;  2391          3033  2      RANGE = .NUM_DIGITS;
;  2392          3034  2  !+
;  2393          3035  2  ! Load up the parameter block, stick the value in R0 and call the kernel
;  2394          3036  2  ! conversion routine.
;  2395          3037  2  !-
;  2396          3038  2      KERNEL_BLOCK [STRING_ADDR] = TEMP_STRING;
;  2397          3039  2      KERNEL_BLOCK [SIG_DIGITS] = .RANGE;
;  2398          3040  2      KERNEL_BLOCK [CONV_FLAGS] = KERNEL_BLOCK [RT_RND] = 0;
;  2399          3041  2      SELECTONE .DATA_TYPE OF
;  2400          3042  2      SET
;  2401          3043  2          [DSC$K_DTYPE_F]:
;  2402          3044  2              OTS$$CVT_D_T_R8 (.VALUE, KERNEL_BLOCK [KERNEL_PTR]);
;  2403          3045  2          [DSC$K_DTYPE_D]:
;  2404          3046  2              OTS$$CVT_D_T_R8 (.VALUE, KERNEL_BLOCK [KERNEL_PTR]);
;  2405          3047  2          [DSC$K_DTYPE_P]:
;  2406          3048  3              BEGIN
;  2407          3049  3              !+
;  2408          3050  3              ! No kernel conversion routine so convert to text here and
;  2409          3051  3              ! set some parameters.
;  2410          3052  3              !-
;  2411          3053  3              MAP
;  2412          3054  3                  VALUE : REF BLOCK [,, BYTE];
;  2413          3055  3              LOCAL
;  2414          3056  3                  ZERO_CTR : INITIAL (0),
;  2415          3057  3                  INT_DIGITS;
;  2416          3058  3
;  2417          3059  3              CVTPS (VALUE [DSC$W_LENGTH], .VALUE [DSC$A_POINTER],
;  2418          3060  3                     VALUE [DSC$W_LENGTH], TEMP_STRING);
;  2419          3061  3              !+
;  2420          3062  3              ! Calculate the number of integer digits, excluding leading
;  2421          3063  3              ! zeroes.
;  2422          3064  3              !-
;  2423          3065  3              INT_DIGITS = .VALUE [DSC$W_LENGTH] + .VALUE [DSC$B_SCALE];
;  2424          3066  3              INCR I FROM 1 TO .INT_DIGITS DO
;  2425          3067  3                  IF .TEMP_STRING [.I] EQL %C'0' THEN ZERO_CTR = .ZERO_CTR + 1
;  2426          3068  3                  ELSE EXITLOOP;
;  2427          3069  3              KERNEL_BLOCK [DEC_EXP] = .INT_DIGITS - .ZERO_CTR;
;  2428          3070  3              KERNEL_BLOCK [OFFSET] = 1 + .ZERO_CTR;
;  2429          3071  3              IF .TEMP_STRING [0] EQL %C'-'
;  2430          3072  3              THEN
;  2431          3073  3                  KERNEL_BLOCK [SIGN] = -1
;  2432          3074  3              ELSE
;  2433          3075  3                  KERNEL_BLOCK [SIGN] = 1;
;  2434          3076  3              !+
;  2435          3077  3              ! Range should be set to the actual number of digits in the
;  2436          3078  3              ! number, since there is no limit on the print size of decimal.
;  2437          3079  3              ! Note that this means only f format will be used by PRINT, STR$
;  2438          3080  3              ! and NUM$.
;  2439          3081  3              !-
;  2440          3082  3              RANGE = .KERNEL_BLOCK [DEC_EXP] - .VALUE [DSC$B_SCALE];
;  2441          3083  2              END;
;  2442          3084  2          [DSC$K_DTYPE_G]:
;  2443          3085  2              OTS$$CVT_G_T_R8 (.VALUE, KERNEL_BLOCK [KERNEL_PTR]);
;  2444          3086  2          [DSC$K_DTYPE_H]:
;  2445          3087  2              OTS$$CVT_H_T_R8 (.VALUE, KERNEL_BLOCK [KERNEL_PTR]);
```

```
;  2446          3088  2          [OTHERWISE]:
;  2447          3089  2              LIB$STOP (OTS$_FATINTERR);
;  2448          3090  2      TES;
;  2449          3091  2  !+
;  2450          3092  2  ! Adjust for the scale factor, if any
;  2451          3093  2  !-
;  2452          3094  2      KERNEL_BLOCK [DEC_EXP] = .KERNEL_BLOCK [DEC_EXP] + .BAS_SCALE_FAC [0];
;  2453          3095  2  !+
;  2454          3096  2  ! Find the last non-zero digit in order to know the actual length of the number
;  2455          3097  2  !-
;  2456          3098  2      TEMP_PTR = .KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET] + .RANGE - 1;
;  2457          3099  2
;  2458          3100  2      WHILE .TEMP_PTR [0] EQL %C'0' DO
;  2459          3101  2          TEMP_PTR = .TEMP_PTR - 1;
;  2460          3102  2
;  2461          3103  2      NO_DIGITS = .TEMP_PTR - (.KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET]) + 1;
;  2462          3104  2  !+
;  2463          3105  2  ! Now determine whether to do E format or F format.  Note 0 has been special
;  2464          3106  2  ! cased.  Some rough decisions can be made based on the exponent of the value
;  2465          3107  2  ! returned by the kernel conversion routine.  If the exponent is > 6 or < -6
;  2466          3108  2  ! the number should be output in E format.  If the exponent is < 6 and > 0,
;  2467          3109  2  ! then the number is output as F format.  For numbers where the exponent is
;  2468          3110  2  ! > -6 and < 0, the actual value comes into play.  If the number can be represented
;  2469          3111  2  ! no less accurately in 6 digits, then it is output in F format.  Otherwise,
;  2470          3112  2  ! the number is output in E format with 6 significant digits.
;  2471          3113  2  ! If NUM_DIGITS is specified, then substitute its value for 6 in the above
;  2472          3114  2  ! discussion.
;  2473          3115  2  !-
;  2474          3116  2
;  2475          3117  2      SELECTONE .KERNEL_BLOCK [DEC_EXP] OF
;  2476          3118  2          SET
;  2477          3119  2
;  2478          3120  2          [0 TO .RANGE] :
;  2479          3121  2  !+
;  2480          3122  2  ! Unconditionally F format
;  2481          3123  2  !-
;  2482          3124  3              BEGIN
;  2483          3125  3              SELECTONE .DATA_TYPE OF
;  2484          3126  3              SET
;  2485          3127  3                  [DSC$K_DTYPE_F]:
;  2486          3128  3                      LARGEST_EXP = K_RANGE;
;  2487          3129  3                  [DSC$K_DTYPE_D]:
;  2488          3130  3                      LARGEST_EXP = K_RANGE;
;  2489          3131  3                  [DSC$K_DTYPE_P]:
;  2490          3132  3                      LARGEST_EXP = K_P_RANGE;
;  2491          3133  3                  [DSC$K_DTYPE_G]:
;  2492          3134  3                      LARGEST_EXP = K_G_RANGE;
;  2493          3135  3                  [DSC$K_DTYPE_H]:
;  2494          3136  3                      LARGEST_EXP = K_H_RANGE;
;  2495          3137  3                  [OTHERWISE]:
;  2496          3138  3                      LIB$STOP (OTS$_FATINTERR);
;  2497          3139  3              TES;
;  2498          3140  3              STATUS = PLAIN_F_FORM_11 (.NO_DIGITS, .FLAGS, KERNEL_BLOCK,
;  2499          3141  3                   BAS_OUT_STR [0], BAS_OUT_STR_LEN [0], .LARGEST_EXP);
;  2500          3142  2              END;
;  2501          3143  2
;  2502          3144  2          [-(.RANGE - 1) TO -1] :
```

```
2 !+
2 ! May be either E or F format.  Have to check and see if any accuracy will
2 ! be lost by displaying 6 digits in F format.  If accuracy will be lost,
2 ! then use E format.
2 !-
2
3               IF ((.NO_DIGITS - .KERNEL_BLOCK [DEC_EXP]) LEQ .RANGE)
2               THEN
2 !+
2 ! F format
2 !-
3                   BEGIN
3                   SELECTONE .DATA_TYPE OF
3                   SET
3                       [DSC$K_DTYPE_F]:
3                           LARGEST_EXP = K_RANGE;
3                       [DSC$K_DTYPE_D]:
3                           LARGEST_EXP = K_RANGE;
3                       [DSC$K_DTYPE_P]:
3                           LARGEST_EXP = K_P_RANGE;
3                       [DSC$K_DTYPE_G]:
3                           LARGEST_EXP = K_G_RANGE;
3                       [DSC$K_DTYPE_H]:
3                           LARGEST_EXP = K_H_RANGE;
3                       [OTHERWISE]:
3                           LIB$STOP (OTS$_FATINTERR);
3                   TES;
3                   STATUS = PLAIN_F_FORM_11 (.NO_DIGITS, .FLAGS, KERNEL_BLOCK, BAS_OUT_STR [0],
3                       BAS_OUT_STR_LEN [0], .LARGEST_EXP)
3                   END
2               ELSE
2 !+
2 ! E format
2 !-
2                   STATUS = PLAIN_E_FORM_11 (.NO_DIGITS, .FLAGS, KERNEL_BLOCK, BAS_OUT_STR [0],
2                       BAS_OUT_STR_LEN [0], .DATA_TYPE );
2
2           [OTHERWISE] :
2 !+
2 ! Unconditionally E format
2 !-
2               STATUS = PLAIN_E_FORM_11 (.NO_DIGITS, .FLAGS, KERNEL_BLOCK,
2                   BAS_OUT_STR [0], BAS_OUT_STR_LEN [0], .DATA_TYPE)
2           ;
2           TES;
2
2       RETURN .STATUS;
1       END;                                    !End of COMMON_G


                  00BF2          .BLKB    2
       00000000#  00BF4 P.AAY:   .LONG    0[9]                    ;
             20   00C18 P.AAZ:   .ASCII   \ \                     ;
                  00C19          .BLKB    3
             30   00C1C P.ABA:   .ASCII   \0\                     ;
```

```
                                BLANK=                  P.AAZ
                                ZERO=                   P.ABA

                                        OFFC    8F   BB 00000 COMMON_G:
                                                               PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>    ; 2917
                                     5E   90    AE   9E 00004         MOVAB   -112(SP), SP
                                     5B         56   D0 00008         MOVL    R6, R11
                                     57         55   D0 0000B         MOVL    R5, R7
                                                54   DD 0000E         PUSHL   R4
                                08   AE         52   7D 00010         MOVQ    R2, BAS_OUT_STR_LEN
                                04   AE         51   D0 00014         MOVL    R1, FLAGS
                                     5A         50   D0 00018         MOVL    R0, R10
                           1C   AE   B8    AF   24   28 0001B         MOVC3   #36, P.AAY, KERNEL_BLOCK        ; 2986
                                     15         5B   D1 00021         CMPL    DATA_TYPE, #21                  ; 2996
                                                15   12 00024         BNEQ    1$
                F3B5  CF        0F   04    BA   6A   37 00026         CMPP4   (VALUE), @4(VALUE), #15, PACKED_ZERO    ; 3000
                      54        54         02   54   DC 0002E         MOVPSL  R4
                                50         01   02   EF 00030         EXTZV   #2, #2, R4, R4
                                                54   C3 00035         SUBL3   R4, #1, RESULT
                                                03   11 00039         BRB     2$                              ; 2994
                                     50         6A   D0 0003B 1$:      MOVL    (VALUE), RESULT                 ; 3004
                                                65   12 0003E 2$:      BNEQ    9$                              ; 3007
                           00000000'    EF   B8    AF   9E 00040         MOVAB   BLANK, B_D                      ; 3021
                                  14    AE 010E0001  8F   D0 00048         MOVL    #17694721, ZERO_STR_DESC        ; 3024
                                  18    AE   AC    AF   9E 00050         MOVAB   ZERO, ZERO_STR_DESC+4           ; 3025
                                  09         04    AE   E9 00055         BLBC    FLAGS, 3$                       ; 3028
                                     50 00000000'   EF   9E 00059         MOVAB   NULL_DESC, R0
                                                07   11 00060         BRB     4$
                                     50 00000000'   EF   9E 00062 3$:      MOVAB   BLANK_DESC, R0
                                                50   DD 00069 4$:      PUSHL   R0
                                           18    AE   9F 0006B         PUSHAB  ZERO_STR_DESC                   ; 3026
                                     09         0C    AE   E9 0006E         BLBC    FLAGS, 5$                       ; 3027
                                     50 00000000'   EF   9E 00072         MOVAB   NULL_DESC, R0
                                                07   11 00079         BRB     6$
                                     50 00000000'   EF   9E 0007B 5$:      MOVAB   BLANK_DESC, R0
                                                50   DD 00082 6$:      PUSHL   R0
                                           18    AE   DD 00084         PUSHL   BAS_OUT_STR                     ; 3026
                           000000000G      00         04   FB 00087         CALLS   #4, STR$CONCAT
                                  10    AE            50   D0 0008E         MOVL    R0, STATUS
                                     05         04    AE   E9 00092         BLBC    FLAGS, 7$                       ; 3029
                                     50                01   D0 00096         MOVL    #1, R0
                                                       03   11 00099         BRB     8$
                                     50                03   D0 0009B 7$:      MOVL    #3, R0
                                08   BE                50   D0 0009E 8$:      MOVL    R0, @BAS_OUT_STR_LEN
                                                     0178  31   000A2         BRW     34$                             ; 3030
                                     59                57   D0 000A5 9$:      MOVL    NUM_DIGITS, RANGE               ; 3033
                                2C   AE           40    AE   9E 000A8         MOVAB   TEMP_STRING, KERNEL_BLOCK+16    ; 3038
                                30   AE                59   D0 000AD         MOVL    RANGE, KERNEL_BLOCK+20          ; 3039
                                           1C    AE   D4 000B1         CLRL    KERNEL_BLOCK                    ; 3040
                                           34    AE   D4 000B4         CLRL    KERNEL_BLOCK+24
                                     0A                5B   D1 000B7         CMPL    DATA_TYPE, #10                  ; 3043
                                                       05   13 000BA         BEQL    10$
                                     0B                5B   D1 000BC         CMPL    DATA_TYPE, #11                  ; 3045
                                                       0F   12 000BF         BNEQ    11$
                                     51           40    AE   9E 000C1 10$:      MOVAB   KERNEL_BLOCK+36, R1     ; 3046
                                     50                5A   D0 000C5         MOVL    VALUE, R0
```

```
                                00000000G  00  16 000C8          JSB     OTS$$CVT_D_T_R8                                 ;
                                           74  11 000CE          BRB     19$                                             ;
                                    15     5B  D1 000D0 11$:     CMPL    DATA_TYPE, #21                                  ; 3047
                                           49  12 000D3          BNEQ    17$                                             ;
                                           54  D4 000D5          CLRL    ZERO_CTR                                        ; 3048
                                       40  AE  6A  04  BA         6A  08 000D7          CVTPS   (VALUE), @4(VALUE), (VALUE), TEMP_STRING        ; 3060
                                           51      6A  3C 000DE          MOVZWL  (VALUE), INT_DIGITS                     ; 3065
                                           50  08  AA  98 000E1          CVTBL   8(VALUE), R0                            ;
                                           51      50  C0 000E5          ADDL2   R0, INT_DIGITS                          ;
                                               50  D4 000E8          CLRL    I                                       ; 3066
                                               09  11 000EA          BRB     13$                                     ;
                                    30  40  AE40  91 000EC 12$:     CMPB    TEMP_STRING[I], #48                     ; 3067
                                               06  12 000F1          BNEQ    14$                                     ;
                                               54  D6 000F3          INCL    ZERO_CTR                                ;
                                    F3         50      51  F3 000F5 13$:     AOBLEQ  INT_DIGITS, I, 12$              ;
                                24  AE         51      54  C3 000F9 14$:     SUBL3   ZERO_CTR, INT_DIGITS, KERNEL_BLOCK+8    ; 3069
                                20  AE  01     A4  9E 000FE          MOVAB   1(R4), KERNEL_BLOCK+4                   ; 3070
                                2D  40  AE     91  00103          CMPB    TEMP_STRING, #45                        ; 3071
                                               06  12 00107          BNEQ    15$                                     ;
                                28  AE         01  CE 00109          MNEGL   #1, KERNEL_BLOCK+12                     ; 3073
                                               04  11 0010D          BRB     16$                                     ;
                                28  AE         01  D0 0010F 15$:     MOVL    #1, KERNEL_BLOCK+12                     ; 3075
                                59     08  AA  98 00113 16$:     CVTBL   8(VALUE), RANGE                         ; 3082
                            59  24  AE         59  C3 00117          SUBL3   RANGE, KERNEL_BLOCK+8, RANGE            ;
                                               35  11 0011C          BRB     21$                                     ; 3041
                                    1B         5B  D1 0011E 17$:     CMPL    DATA_TYPE, #27                          ; 3084
                                               0F  12 00121          BNEQ    18$                                     ;
                                51     40  AE  9E 00123          MOVAB   KERNEL_BLOCK+36, R1                     ; 3085
                                50             5A  D0 00127          MOVL    VALUE, R0                               ;
                                00000000G      00  16 0012A          JSB     OTS$$CVT_G_T_R8                         ;
                                               21  11 00130          BRB     21$                                     ;
                                    1C         5B  D1 00132 18$:     CMPL    DATA_TYPE, #28                          ; 3086
                                               0F  12 00135          BNEQ    20$                                     ;
                                51     40  AE  9E 00137          MOVAB   KERNEL_BLOCK+36, R1                     ; 3087
                                50             5A  D0 0013B          MOVL    VALUE, R0                               ;
                                00000000G      00  16 0013E          JSB     OTS$$CVT_H_T_R8                         ;
                                               0D  11 00144 19$:     BRB     21$                                     ;
                                00000000G      8F  DD 00146 20$:     PUSHL   #OTS$_FATINTERR                         ; 3089
                            00000000G  00      01  FB 0014C          CALLS   #1, LIB$STOP                            ;
                                50             6E  98 00153 21$:     CVTBL   BAS_SCALE_FAC, R0                       ; 3094
                                24  AE         50  C0 00156          ADDL2   R0, KERNEL_BLOCK+8                      ;
                            51  2C  AE     20  AE  C1 0015A          ADDL3   KERNEL_BLOCK+4, KERNEL_BLOCK+16, R1     ; 3098
                                50     FF  A941  9E 00160          MOVAB   -1(RANGE)[R1], TEMP_PTR                 ;
                                30             60  91 00165 22$:     CMPB    (TEMP_PTR), #48                         ; 3100
                                               04  12 00168          BNEQ    23$                                     ;
                                               50  D7 0016A          DECL    TEMP_PTR                                ; 3101
                                               F7  11 0016C          BRB     22$                                     ;
                                50             51  C2 0016E 23$:     SUBL2   R1, R0                                  ; 3103
                                56     01      A0  9E 00171          MOVAB   1(R0), NO_DIGITS                        ;
                                52     24      AE  D0 00175          MOVL    KERNEL_BLOCK+8, R2                      ; 3117
                                               2A  19 00179          BLSS    25$                                     ; 3120
                                59             52  D1 0017B          CMPL    R2, RANGE                               ;
                                               25  14 0017E          BGTR    25$                                     ;
                                    0A         5B  D1 00180          CMPL    DATA_TYPE, #10                          ; 3127
                                               43  13 00183          BEQL    26$                                     ;
                                    0B         5B  D1 00185          CMPL    DATA_TYPE, #11                          ; 3129
                                               3E  13 00188          BEQL    26$                                     ;
```

```
                        15              5B  D1 0018A          CMPL    DATA_TYPE, #21                          ; 3131
                                        54  13 0018D          BEQL    30$
                        1B              5B  D1 0018F          CMPL    DATA_TYPE, #27                          ; 3133
                                        43  13 00192          BEQL    28$
                                        48  11 00194          BRB     29$                                     ; 3135
                            00000000G   8F  DD 00196 24$:     PUSHL   #OTS$_FATINTERR                         ; 3138
                    00000000G   00      01  FB 0019C          CALLS   #1, LIB$STOP
                                        43  11 001A3          BRB     31$                                     ; 3140
                        50      FF      A9  9E 001A5 25$:     MOVAB   -1(R9), R0                              ; 3144
                        50              50  CE 001A9          MNEGL   R0, R0
                        50              52  D1 001AC          CMPL    R2, R0
                                        4F  19 001AF          BLSS    32$
                                        52  D5 001B1          TSTL    R2
                                        4B  18 001B3          BGEQ    32$
                50              56      52  C3 001B5          SUBL3   R2, NO_DIGITS, R0                       ; 3151
                        59              50  D1 001B9          CMPL    R0, RANGE
                                        42  14 001BC          BGTR    32$
                        0A              5B  D1 001BE          CMPL    DATA_TYPE, #10                          ; 3159
                                        05  13 001C1          BEQL    26$
                        0B              5B  D1 001C3          CMPL    DATA_TYPE, #11                          ; 3161
                                        05  12 001C6          BNEQ    27$
                        55              26  D0 001C8 26$:     MOVL    #38, LARGEST_EXP                        ; 3162
                                        1B  11 001CB          BRB     31$
                        15              5B  D1 001CD 27$:     CMPL    DATA_TYPE, #21                          ; 3163
                                        11  13 001D0          BEQL    30$
                        1B              5B  D1 001D2          CMPL    DATA_TYPE, #27                          ; 3165
                                        07  12 001D5          BNEQ    29$
                        55      0134    8F  3C 001D7 28$:     MOVZWL  #308, LARGEST_EXP                       ; 3166
                                        0A  11 001DC          BRB     31$
                        1C              5B  D1 001DE 29$:     CMPL    DATA_TYPE, #28                          ; 3167
                                        B3  12 001E1          BNEQ    24$
                        55      1344    8F  3C 001E3 30$:     MOVZWL  #4932, LARGEST_EXP                      ; 3168
                        52        1C    AE  9E 001E8 31$:     MOVAB   KERNEL_BLOCK, R2                        ; 3172
                        54        08    AE  D0 001EC          MOVL    BAS_OUT_STR_LEN, R4                     ; 3173
                        53        0C    AE  D0 001F0          MOVL    BAS_OUT_STR, R3
                        51        04    AE  D0 001F4          MOVL    FLAGS, R1
                        50              56  D0 001F8          MOVL    NO_DIGITS, R0
                                  0000V 30  11 001FB          BSBW    PLAIN_F_FORM_11
                                        19  11 001FE          BRB     33$
                        52        1C    AE  9E 00200 32$:     MOVAB   KERNEL_BLOCK, R2                        ; 3186
                        55              5B  D0 00204          MOVL    DATA_TYPE, R5                           ; 3187
                        54        08    AE  D0 00207          MOVL    BAS_OUT_STR_LEN, R4
                        53        0C    AE  D0 0020B          MOVL    BAS_OUT_STR, R3
                        51        04    AE  D0 0020F          MOVL    FLAGS, R1
                        50              56  D0 00213          MOVL    NO_DIGITS, R0
                                  0000V 30  00216          BSBW    PLAIN_E_FORM_11
                    10  AE              50  D0 00219 33$:     MOVL    R0, STATUS
                        50        10    AE  D0 0021D 34$:     MOVL    STATUS, R0                              ; 3191
                        5E        74    AE  9E 00221          MOVAB   116(SP), SP                             ; 3192
                                  0FFC  8F  BA 00225          POPR    #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                                            05 00229          RSB

; Routine Size:  554 bytes,    Routine Base:  _BAS$CODE + 0C1D

; 2551          3193  1
```

```
;  2553          3194  1 ROUTINE PLAIN_F_FORM_11 (                        ! Plain old F formatting
;  2554          3195  1         NO_DIGITS,                               !
;  2555          3196  1         FLAGS,                                   !
;  2556          3197  1         KERNEL_BLOCK,                            !
;  2557          3198  1         BAS_OUT_STR,                             !
;  2558          3199  1         BAS_OUT_STR_LEN,                         !
;  2559          3200  1         LARGEST_EXP                              !
;  2560          3201  1     ) : JSB_FORMAT_A6 =
;  2561          3202  1
;  2562          3203  1 !++
;  2563          3204  1 ! FUNCTIONAL DESCRIPTION:
;  2564          3205  1 !
;  2565          3206  1 !       Do plain old explicit point unscaled formatting.  Strip leading and
;  2566          3207  1 !       trailing spaces if necessary.
;  2567          3208  1 !
;  2568          3209  1 ! FORMAL PARAMETERS:
;  2569          3210  1 !
;  2570          3211  1 !       NO_DIGITS.rlu.v         number of actual significant digits in value to
;  2571          3212  1 !                               be formatted
;  2572          3213  1 !       FLAGS.rlu.v             no flags are applicable
;  2573          3214  1 !       KERNEL_BLOCK.mz.r       block returned by kernel conversion routine
;  2574          3215  1 !       BAS_OUT_STR_LEN.wlu.v   length of output string (handy for fixed length
;  2575          3216  1 !                               output strings
;  2576          3217  1 !       BAS_OUT_STR.wt.dx       the output string
;  2577          3218  1 !       LARGEST_EXP.rlu.v       largest possible exponent for data type
;  2578          3219  1 !
;  2579          3220  1 ! IMPLICIT INPUTS:
;  2580          3221  1 !
;  2581          3222  1 !       NONE
;  2582          3223  1 !
;  2583          3224  1 ! IMPLICIT OUTPUTS:
;  2584          3225  1 !
;  2585          3226  1 !       NONE
;  2586          3227  1 !
;  2587          3228  1 ! COMPLETION CODES:
;  2588          3229  1 !
;  2589          3230  1 !       Returns STR$_TRU
;  2590          3231  1 !
;  2591          3232  1 ! SIDE EFFECTS:
;  2592          3233  1 !
;  2593          3234  1 !       NONE
;  2594          3235  1 !
;  2595          3236  1 !--
;  2596          3237  1
;  2597          3238  2     BEGIN
;  2598          3239  2
;  2599          3240  2
;  2600          3241  2     LOCAL
;  2601          3242  2         ZEROES_DESC : BLOCK [8, BYTE],          ! if used must be known to whole module
;  2602          3243  2         ZEROES_PTR,
;  2603          3244  2         STATUS;
;  2604          3245  2
;  2605          3246  2     MAP
;  2606          3247  2         BAS_OUT_STR_LEN : REF VECTOR,
;  2607          3248  2         BAS_OUT_STR : REF VECTOR [2, LONG],
;  2608          3249  2         KERNEL_BLOCK : REF BLOCK [K_KERNEL_BLK_SZ, BYTE];
;  2609          3250  2
```

```
;  2610          3251  2     INITIALIZE_DESC;
;  2611          3252  2
;  2612          3253  2 !+
;  2613          3254  2 ! The largest possible exponent for single and double floating point is 38.
;  2614          3255  2 ! G and h floating, however, have largest exponents of 308 and 4932 respectively.
;  2615          3256  2 ! NUM1$ must be able to print out all these characters but the user who doesn't
;  2616          3257  2 ! want all these should not have to pay the performance penalty.  Therefore,
;  2617          3258  2 ! fill characters (zero, star, blank) are initialized to size 38 and then
;  2618          3259  2 ! re-initialized here only if necessary.
;  2619          3260  2 !-
;  2620          3261  2
;  2621          3262  3     IF (.LARGEST_EXP GTR K_RANGE)
;  2622          3263  2     THEN
;  2623          3264  3         BEGIN
;  2624          3265  3         LOCAL
;  2625          3266  3             ZERO : BYTE;
;  2626          3267  3
;  2627          3268  3         ZEROES_DESC [DSC$W_LENGTH] = .LARGEST_EXP;
;  2628          3269  3         ZEROES_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  2629          3270  3         ZEROES_DESC [DSC$B_CLASS] = DSC$K_CLASS_D;
;  2630          3271  3         ZEROES_DESC [DSC$A_POINTER] = 0;        ! STR$DUPL_CHAR will alloc space
;  2631          3272  3
;  2632          3273  3         ZERO = %X'30';
;  2633          3274  3         STR$DUPL_CHAR (ZEROES_DESC, LARGEST_EXP, ZERO);
;  2634          3275  3                                                 ! create string of zeroes
;  2635          3276  3         ZEROES_PTR = .ZEROES_DESC [DSC$A_POINTER];
;  2636          3277  3
;  2637          3278  3         END
;  2638          3279  2     ELSE
;  2639          3280  2         ZEROES_PTR = ZEROES;                    ! use string of 38 zeroes
;  2640          3281  2
;  2641          3282  2
;  2642          3283  3     IF (.KERNEL_BLOCK [DEC_EXP] GEQ 0)
;  2643          3284  2     THEN
;  2644          3285  2 !+
;  2645          3286  2 ! The number is greater than or equal to 1
;  2646          3287  2 !-
;  2647          3288  2
;  2648          3289  3         IF (.KERNEL_BLOCK [DEC_EXP] GEQ .NO_DIGITS)
;  2649          3290  2         THEN
;  2650          3291  3             BEGIN
;  2651          3292  3 !+
;  2652          3293  3 ! This entire number consists only of integer part.  Put the number
;  2653          3294  3 ! returned by the conversion routine and optionally some trailing
;  2654          3295  3 ! zeroes into the return strirg.
;  2655          3296  3 !-
;  2656          3297  3
;  2657          3298  3             LOCAL
;  2658          3299  3                 ZERO_STR_DESC : BLOCK [8, BYTE],        ! desc. for trailing zeroes
;  2659          3300  3                 INT_STR_DESC : BLOCK [8, BYTE]; ! desc. for integer portion of number
;  2660          3301  3
;  2661          3302  3             INT_STR_DESC [DSC$W_LENGTH] = .NO_DIGITS;
;  2662          3303  3             INT_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  2663          3304  3             INT_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  2664          3305  3             INT_STR_DESC [DSC$A_POINTER] = .KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET];
;  2665          3306  3             ZERO_STR_DESC [DSC$W_LENGTH] = .KERNEL_BLOCK [DEC_EXP] - .NO_DIGITS;
;  2666          3307  3             ZERO_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
```

```
;  2667          3308  3          ZERO_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  2668          3309  3          ZERO_STR_DESC [DSC$A_POINTER] = .ZEROES_PTR;
;  2669          3310  3          STATUS = STR$CONCAT (BAS_OUT_STR [0],
;  2670          3311  4              BEGIN
;  2671          3312  4
;  2672          3313  5              IF (.KERNEL_BLOCK [SIGN] LSS 0)
;  2673          3314  4              THEN
;  2674          3315  4                  MINUS_DESC
;  2675          3316  4              ELSE
;  2676          3317  4
;  2677          3318  4                  IF (.FLAGS AND V_STRIP_SPACES) THEN NULL_DESC ELSE BLANK_DESC
;  2678          3319  4
;  2679          3320  4              END
;  2680          3321  3          , INT_STR_DESC, ZERO_STR_DESC,
;  2681          3322  4              BEGIN
;  2682          3323  4
;  2683          3324  4              IF (.FLAGS AND V_STRIP_SPACES) THEN NULL_DESC ELSE BLANK_DESC
;  2684          3325  4
;  2685          3326  4              END
;  2686          3327  3          );
;  2687          3328  3          BAS_OUT_STR_LEN [0] =
;  2688          3329  4          BEGIN
;  2689          3330  4
;  2690          3331  5          IF (.FLAGS AND V_STRIP_SPACES)
;  2691          3332  4          THEN
;  2692          3333  4
;  2693          3334  4              IF (.KERNEL_BLOCK [SIGN] LSS 0) THEN 1 ELSE 0
;  2694          3335  4
;  2695          3336  4          ELSE
;  2696          3337  4              2
;  2697          3338  4
;  2698          3339  4              END
;  2699          3340  3              + .KERNEL_BLOCK [DEC_EXP];
;  2700          3341  3              END
;  2701          3342  2          ELSE
;  2702          3343  3              BEGIN
;  2703          3344  3          !+
;  2704          3345  3          ! This number has both integer and fractional portions.
;  2705          3346  3          !-
;  2706          3347  3
;  2707          3348  3              LOCAL
;  2708          3349  3                  INT_STR_DESC : BLOCK [8, BYTE],
;  2709          3350  3                  FRAC_STR_DESC : BLOCK [8, BYTE];
;  2710          3351  3
;  2711          3352  3              INT_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  2712          3353  3              INT_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  2713          3354  3              INT_STR_DESC [DSC$W_LENGTH] = .KERNEL_BLOCK [DEC_EXP];
;  2714          3355  3              INT_STR_DESC [DSC$A_POINTER] = .KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET];
;  2715          3356  3              FRAC_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  2716          3357  3              FRAC_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  2717          3358  3              FRAC_STR_DESC [DSC$W_LENGTH] = .NO_DIGITS - .KERNEL_BLOCK [DEC_EXP];
;  2718          3359  3              FRAC_STR_DESC [DSC$A_POINTER] = .INT_STR_DESC [DSC$A_POINTER] + .KERNEL_BLOCK [DEC_EXP];
;  2719          3360  3              STATUS = STR$CONCAT (BAS_OUT_STR [0],
;  2720          3361  4                  BEGIN
;  2721          3362  4
;  2722          3363  5                  IF (.KERNEL_BLOCK [SIGN] LSS 0)
;  2723          3364  4                  THEN
```

```
;  2724          3365  4                    MINUS_DESC
;  2725          3366  4                ELSE
;  2726          3367  4
;  2727          3368  4                    IF (.FLAGS AND V_STRIP_SPACES) THEN NULL_DESC ELSE BLANK_DESC
;  2728          3369  4
;  2729          3370  4                END
;  2730          3371  3            , INT_STR_DESC, DOT_DESC, FRAC_STR_DESC,
;  2731          3372  4                BEGIN
;  2732          3373  4
;  2733          3374  4                IF (.FLAGS AND V_STRIP_SPACES) THEN NULL_DESC ELSE BLANK_DESC
;  2734          3375  4
;  2735          3376  4                END
;  2736          3377  3            );
;  2737          3378  3            BAS_OUT_STR_LEN [0] =
;  2738          3379  4            BEGIN
;  2739          3380  4
;  2740          3381  5            IF (.FLAGS AND V_STRIP_SPACES)
;  2741          3382  4            THEN
;  2742          3383  4
;  2743          3384  4                IF (.KERNEL_BLOCK [SIGN] LSS 0) THEN 1 ELSE 0
;  2744          3385  4
;  2745          3386  4            ELSE
;  2746          3387  4                2
;  2747          3388  4
;  2748          3389  4            END
;  2749          3390  3            + .NO_DIGITS + 1;
;  2750          3391  3            END
;  2751          3392  3
;  2752          3393  2        ELSE
;  2753          3394  3            BEGIN
;  2754          3395  3        !+
;  2755          3396  3        ! This is a purely fractional number which should be put out in F format
;  2756          3397  3        !-
;  2757          3398  3
;  2758          3399  3            LOCAL
;  2759          3400  3                ZERO_STR_DESC : BLOCK [8, BYTE],    ! desc. for leading zeroes
;  2760          3401  3                FRAC_STR_DESC : BLOCK [8, BYTE];    ! desc. for fraction
;  2761          3402  3
;  2762          3403  3            ZERO_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  2763          3404  3            ZERO_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  2764          3405  3            ZERO_STR_DESC [DSC$W_LENGTH] = -.KERNEL_BLOCK [DEC_EXP];
;  2765          3406  3            ZERO_STR_DESC [DSC$A_POINTER] = .ZEROES_PTR;
;  2766          3407  3            FRAC_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  2767          3408  3            FRAC_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  2768          3409  3            FRAC_STR_DESC [DSC$W_LENGTH] = .NO_DIGITS;
;  2769          3410  3            FRAC_STR_DESC [DSC$A_POINTER] = .KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET];
;  2770          3411  3            STATUS = STR$CONCAT (BAS_OUT_STR [0],
;  2771          3412  4                BEGIN
;  2772          3413  4
;  2773          3414  5                IF (.KERNEL_BLOCK [SIGN] LSS 0)
;  2774          3415  4                THEN
;  2775          3416  4                    MINUS_DESC
;  2776          3417  4                ELSE
;  2777          3418  4
;  2778          3419  4                    IF (.FLAGS AND V_STRIP_SPACES) THEN NULL_DESC ELSE BLANK_DESC
;  2779          3420  4
;  2780          3421  4                END
```

```
;  2781          3422  3          , DOT_DESC, ZERO_STR_DESC, FRAC_STR_DESC,
;  2782          3423  4              BEGIN
;  2783          3424  4
;  2784          3425  4              IF (.FLAGS AND V_STRIP_SPACES) THEN NULL_DESC ELSE BLANK_DESC
;  2785          3426  4
;  2786          3427  4              END
;  2787          3428  3          );
;  2788          3429  3          BAS_OUT_STR_LEN [0] =
;  2789          3430  4          BEGIN
;  2790          3431  4
;  2791          3432  5          IF (.FLAGS AND V_STRIP_SPACES)
;  2792          3433  4          THEN
;  2793          3434  4
;  2794          3435  4              IF (.KERNEL_BLOCK [SIGN] LSS 0) THEN 1 ELSE 0
;  2795          3436  4
;  2796          3437  4          ELSE
;  2797          3438  4              2
;  2798          3439  4
;  2799          3440  4          END
;  2800          3441  4          ! The exponent is subtracted 'cuz it is negative
;  2801          3442  3          + .NO_DIGITS - .KERNEL_BLOCK [DEC_EXP] + 1;
;  2802          3443  2          END;
;  2803          3444  2
;  2804          3445  3      IF (.LARGEST_EXP GTR K_RANGE)                ! need to free space alloc
;  2805          3446  2      THEN                                         ! for zeroes
;  2806          3447  2          STR$FREE1_DX (ZEROES_DESC);
;  2807          3448  2
;  2808          3449  2      RETURN .STATUS;
;  2809          3450  1      END;                                         ! PLAIN_F_FORMAT


                00E47                  .BLKB   1
2E              00E48 P.ABB:           .ASCII  \.\                                                    ;
                00E49                  .BLKB   3
2A              00E4C P.ABC:           .ASCII  \*\                                                    ;
2A              00E4D                  .ASCII  \*\                                                    ;
2A              00E4E                  .ASCII  \*\                                                    ;
2A              00E4F                  .ASCII  \*\                                                    ;
2A              00E50                  .ASCII  \*\                                                    ;
2A              00E51                  .ASCII  \*\                                                    ;
2A              00E52                  .ASCII  \*\                                                    ;
2A              00E53                  .ASCII  \*\                                                    ;
2A              00E54                  .ASCII  \*\                                                    ;
2A              00E55                  .ASCII  \*\                                                    ;
2A              00E56                  .ASCII  \*\                                                    ;
2A              00E57                  .ASCII  \*\                                                    ;
2A              00E58                  .ASCII  \*\                                                    ;
2A              00E59                  .ASCII  \*\                                                    ;
2A              00E5A                  .ASCII  \*\                                                    ;
2A              00E5B                  .ASCII  \*\                                                    ;
2A              00E5C                  .ASCII  \*\                                                    ;
2A              00E5D                  .ASCII  \*\                                                    ;
2A              00E5E                  .ASCII  \*\                                                    ;
2A              00E5F                  .ASCII  \*\                                                    ;
2A              00E60                  .ASCII  \*\                                                    ;
2A              00E61                  .ASCII  \*\                                                    ;
```

```
2A  00E62           .ASCII  \*\
2A  00E63           .ASCII  \*\
2A  00E64           .ASCII  \*\
2A  00E65           .ASCII  \*\
2A  00E66           .ASCII  \*\
2A  00E67           .ASCII  \*\
2A  00E68           .ASCII  \*\
2A  00E69           .ASCII  \*\
2A  00E6A           .ASCII  \*\
2A  00E6B           .ASCII  \*\
2A  00E6C           .ASCII  \*\
2A  00E6D           .ASCII  \*\
2A  00E6E           .ASCII  \*\
2A  00E6F           .ASCII  \*\
2A  00E70           .ASCII  \*\
2A  00E71           .ASCII  \*\
    00E72           .BLKB   2
2D  00E74 P.ABD:    .ASCII  \-\
    00E75           .BLKB   3
20  00E78 P.ABE:    .ASCII  \ \
    00E79           .BLKB   3
45  00E7C P.ABF:    .ASCII  \E\
    00E7D           .BLKB   3
20  00E80 P.ABG:    .ASCII  \ \
20  00E81           .ASCII  \ \
20  00E82           .ASCII  \ \
20  00E83           .ASCII  \ \
20  00E84           .ASCII  \ \
20  00E85           .ASCII  \ \
20  00E86           .ASCII  \ \
20  00E87           .ASCII  \ \
20  00E88           .ASCII  \ \
20  00E89           .ASCII  \ \
20  00E8A           .ASCII  \ \
20  00E8B           .ASCII  \ \
20  00E8C           .ASCII  \ \
20  00E8D           .ASCII  \ \
20  00E8E           .ASCII  \ \
20  00E8F           .ASCII  \ \
20  00E90           .ASCII  \ \
20  00E91           .ASCII  \ \
20  00E92           .ASCII  \ \
20  00E93           .ASCII  \ \
20  00E94           .ASCII  \ \
20  00E95           .ASCII  \ \
20  00E96           .ASCII  \ \
20  00E97           .ASCII  \ \
20  00E98           .ASCII  \ \
20  00E99           .ASCII  \ \
20  00E9A           .ASCII  \ \
20  00E9B           .ASCII  \ \
20  00E9C           .ASCII  \ \
20  00E9D           .ASCII  \ \
20  00E9E           .ASCII  \ \
20  00E9F           .ASCII  \ \
20  00EA0           .ASCII  \ \
20  00EA1           .ASCII  \ \
```

```
20  00EA2            .ASCII  \ \
20  00EA3            .ASCII  \ \
20  00EA4            .ASCII  \ \
20  00EA5            .ASCII  \ \
    00EA6            .BLKB   2
30  00EA8 P.ABH:     .ASCII  \0\
30  00EA9            .ASCII  \0\
30  00EAA            .ASCII  \0\
30  00EAB            .ASCII  \0\
30  00EAC            .ASCII  \0\
30  00EAD            .ASCII  \0\
30  00EAE            .ASCII  \0\
30  00EAF            .ASCII  \0\
30  00EB0            .ASCII  \0\
30  00EB1            .ASCII  \0\
30  00EB2            .ASCII  \0\
30  00EB3            .ASCII  \0\
30  00EB4            .ASCII  \0\
30  00EB5            .ASCII  \0\
30  00EB6            .ASCII  \0\
30  00EB7            .ASCII  \0\
30  00EB8            .ASCII  \0\
30  00EB9            .ASCII  \0\
30  00EBA            .ASCII  \0\
30  00EBB            .ASCII  \0\
30  00EBC            .ASCII  \0\
30  00EBD            .ASCII  \0\
30  00EBE            .ASCII  \0\
30  00EBF            .ASCII  \0\
30  00EC0            .ASCII  \0\
30  00EC1            .ASCII  \0\
30  00EC2            .ASCII  \0\
30  00EC3            .ASCII  \0\
30  00EC4            .ASCII  \0\
30  00EC5            .ASCII  \0\
30  00EC6            .ASCII  \0\
30  00EC7            .ASCII  \0\
30  00EC8            .ASCII  \0\
30  00EC9            .ASCII  \0\
30  00ECA            .ASCII  \0\
30  00ECB            .ASCII  \0\
30  00ECC            .ASCII  \0\
30  00ECD            .ASCII  \0\

                DOT=                P.ABB
                STAR=               P.ABC
                MINUS=              P.ABD
                BLANK=              P.ABE
                E=                  P.ABF
                BLANKS=             P.ABG
                ZEROES=             P.ABH

                01E4    8F  BB 00000 PLAIN_F_FORM_11:
                                      PUSHR   #^M<R2,R5,R6,R7,R8>          ; 3194
        5E              20  C2 00004          SUBL2   #32, SP
04      AE              55  D0 00007          MOVL    R5, LARGEST_EXP
```

```
                58              51   D0 0000B          MOVL    R1, R8                                  ;
                56              50   D0 0000E          MOVL    R0, R6                                  ;
                    00000000'   EF   D5 00011          TSTL    E_D                                     ; 3249
                                23   12 00017          BNEQ    1$                                      ;
00000000'       EF        8A    AF   9E 00019          MOVAB   MINUS, M_D                              ;
00000000'       EF      FF55    CF   9E 00021          MOVAB   DOT, D_D                                ;
00000000'       EF      FF7C    CF   9E 0002A          MOVAB   BLANK, B_D                              ;
00000000'       EF      FF77    CF   9E 00033          MOVAB   E, E_D                                  ;
                26        04    AE   D1 0003C 1$:      CMPL    LARGEST_EXP, #38                        ; 3262
                                26   15 00040          BLEQ    2$                                      ;
            18  AE        04    AE   B0 00042          MOVW    LARGEST_EXP, ZEROES_DESC                ; 3268
            1A  AE      020E    8F   B0 00047          MOVW    #526, ZEROES_DESC+2                     ; 3269
                          1C    AE   D4 0004D          CLRL    ZEROES_DESC+4                           ; 3271
                6E                 30   90 00050          MOVB    #48, ZERO                               ; 3273
                                   5E   DD 00053          PUSHL   SP                                      ; 3274
                              08   AE   9F 00055          PUSHAB  LARGEST_EXP                             ;
                              20   AE   9F 00058          PUSHAB  ZEROES_DESC                             ;
00000000G       00                 03   FB 0005B          CALLS   #3, STR$DUPL_CHAR                       ;
                50              1C   AE   D0 00062          MOVL    ZEROES_DESC+4, ZEROES_PTR               ; 3276
                                05   11 00066          BRB     3$                                      ; 3262
                50      FF6E    CF   9E 00068 2$:      MOVAB   ZEROES, ZEROES_PTR                      ; 3280
                51        0C    A2   9E 0006D 3$:      MOVAB   12(KERNEL_BLOCK), R1                    ; 3313
                55        08    A2   D0 00071          MOVL    8(KERNEL_BLOCK), R5                     ; 3283
                                03   18 00075          BGEQ    4$                                      ;
                              011D   31 00077          BRW     23$                                     ;
                56                55   D1 0007A 4$:      CMPL    R5, NO_DIGITS                           ; 3289
                                03   18 0007D          BGEQ    5$                                      ;
                              0086   31 0007F          BRW     14$                                     ;
            08  AE                56   B0 00082 5$:      MOVW    NO_DIGITS, INT_STR_DESC                 ; 3302
            0A  AE      010E    8F   B0 00086          MOVW    #270, INT_STR_DESC+2                    ; 3303
0C  AE      10  A2        04    A2   C1 0008C          ADDL3   4(KERNEL_BLOCK), 16(KERNEL_BLOCK), -    ; 3305
                                                                INT_STR_DESC+4
10  AE              55                56   A3 00093          SUBW3   NO_DIGITS, R5, ZERO_STR_DESC            ; 3306
            12  AE      010E    8F   B0 00098          MOVW    #270, ZERO_STR_DESC+2                   ; 3308
            14  AE                50   D0 0009E          MOVL    ZEROES_PTR, ZERO_STR_DESC+4             ; 3309
                09                58   E9 000A2          BLBC    FLAGS, 6$                               ; 3324
                50 00000000'      EF   9E 000A5          MOVAB   NULL_DESC, R0                           ;
                                  07   11 000AC          BRB     7$                                      ;
                50 00000000'      EF   9E 000AE 6$:      MOVAB   BLANK_DESC, R0                          ;
                                  50   DD 000B5 7$:      PUSHL   R0                                      ;
                          14      AE   9F 000B7          PUSHAB  ZERO_STR_DESC                           ; 3310
                          10      AE   9F 000BA          PUSHAB  INT_STR_DESC                            ;
                                  52   D4 000BD          CLRL    R2                                      ; 3313
                                  61   D5 000BF          TSTL    (R1)                                    ;
                                  0B   18 000C1          BGEQ    8$                                      ;
                                  52   D6 000C3          INCL    R2                                      ;
                50 00000000'      EF   9E 000C5          MOVAB   MINUS_DESC, R0                          ;
                                  13   11 000CC          BRB     10$                                     ;
                09                58   E9 000CE 8$:      BLBC    FLAGS, 9$                               ; 3318
                50 00000000'      EF   9E 000D1          MOVAB   NULL_DESC, R0                           ;
                                  07   11 000D8          BRB     10$                                     ;
                50 00000000'      EF   9E 000DA 9$:      MOVAB   BLANK_DESC, R0                          ;
                                  50   DD 000E1 10$:     PUSHL   R0                                      ;
                                  53   DD 000E3          PUSHL   BAS_OUT_STR                             ; 3310
        00000000G 00              05   FB 000E5          CALLS   #5, STR$CONCAT                          ;
                57                50   D0 000EC          MOVL    R0, STATUS                              ;
                0C                58   E9 000EF          BLBC    FLAGS, 12$                              ; 3331
```

```
                        05            52  E9 000F2          BLBC    R2, 11$                                 ; 3334
                        50            01  D0 000F5          MOVL    #1, R0
                                      07  11 000F8          BRB     13$
                                      50  D4 000FA 11$:      CLRL    R0
                                      03  11 000FC          BRB     13$
                        50            02  D0 000FE 12$:      MOVL    #2, R0                                  ; 3331
                64                    50  C1 00101 13$:      ADDL3   R5, R0, (BAS_OUT_STR_LEN)               ; 3340
                                  011D    31 00105          BRW     32$                                     ; 3289
                    12  AE      010E    8F  B0 00108 14$:      MOVW    #270, INT_STR_DESC+2                    ; 3353
                    10  AE              55  B0 0010E          MOVW    R5, INT_STR_DESC                        ; 3354
            14  AE      10  A2      04  A2  C1 00112          ADDL3   4(KERNEL_BLOCK), 16(KERNEL_BLOCK), -    ; 3355
                                                                INT_STR_DESC+4
                    0A  AE      010E    8F  B0 00119          MOVW    #270, FRAC_STR_DESC+2                   ; 3357
            08  AE      56              55  A3 0011F          SUBW3   R5, NO_DIGITS, FRAC_STR_DESC            ; 3358
                    0C  AE          14 BE45 9E 00124          MOVAB   @INT_STR_DESC+4[R5], FRAC_STR_DESC+4    ; 3359
                        09              58  E9 0012A          BLBC    FLAGS, 15$                              ; 3374
                        50 00000000'    EF  9E 0012D          MOVAB   NULL_DESC, R0
                                        07  11 00134          BRB     16$
                        50 00000000'    EF  9E 00136 15$:      MOVAB   BLANK_DESC, R0
                                        50  DD 0013D 16$:      PUSHL   R0
                                  0C    AE  9F 0013F          PUSHAB  FRAC_STR_DESC                           ; 3360
                             00000000'  EF  9F 00142          PUSHAB  DOT_DESC
                                  1C    AE  9F 00148          PUSHAB  INT_STR_DESC
                                        52  D4 0014B          CLRL    R2                                      ; 3363
                                        61  D5 0014D          TSTL    (R1)
                                        0B  18 0014F          BGEQ    17$
                                        52  D6 00151          INCL    R2
                        50 00000000'    EF  9E 00153          MOVAB   MINUS_DESC, R0
                                        13  11 0015A          BRB     19$
                        09              58  E9 0015C 17$:      BLBC    FLAGS, 18$                              ; 3368
                        50 00000000'    EF  9E 0015F          MOVAB   NULL_DESC, R0
                                        07  11 00166          BRB     19$
                        50 00000000'    EF  9E 00168 18$:      MOVAB   BLANK_DESC, R0
                                        50  DD 0016F 19$:      PUSHL   R0
                                        53  DD 00171          PUSHL   BAS_OUT_STR                             ; 3360
                00000000G   00              06  FB 00173          CALLS   #6, STR$CONCAT
                            57              50  D0 0017A          MOVL    R0, STATUS
                            0C              58  E9 0017D          BLBC    FLAGS, 21$                          ; 3381
                            05              52  E9 00180          BLBC    R2, 20$                             ; 3384
                            50              01  D0 00183          MOVL    #1, R0
                                            07  11 00186          BRB     22$
                                            50  D4 00188 20$:      CLRL    R0
                                            03  11 0018A          BRB     22$
                            50              02  D0 0018C 21$:      MOVL    #2, R0                              ; 3381
                            64          01  A640  9E 0018F 22$:      MOVAB   1(NO_DIGITS)[R0], (BAS_OUT_STR_LEN) ; 3390
                                        008E    31 00194          BRW     32$                                 ; 3289
                        12  AE      010E    8F  B0 00197 23$:      MOVW    #270, ZERO_STR_DESC+2               ; 3404
                        10  AE              55  AE 0019D          MNEGW   R5, ZERO_STR_DESC                   ; 3405
                        14  AE              50  D0 001A1          MOVL    ZEROES_PTR, ZERO_STR_DESC+4         ; 3406
                        0A  AE      010E    8F  B0 001A5          MOVW    #270, FRAC_STR_DESC+2               ; 3408
                        08  AE              56  B0 001AB          MOVW    NO_DIGITS, FRAC_STR_DESC            ; 3409
            0C  AE      10  A2      04  A2  C1 001AF          ADDL3   4(KERNEL_BLOCK), 16(KERNEL_BLOCK), -    ; 3410
                                                                FRAC_STR_DESC+4
                        09              58  E9 001B6          BLBC    FLAGS, 24$                              ; 3425
                        50 00000000'    EF  9E 001B9          MOVAB   NULL_DESC, R0
                                        07  11 001C0          BRB     25$
                        50 00000000'    EF  9E 001C2 24$:      MOVAB   BLANK_DESC, R0
```

```
                                50  DD 001C9 25$:    PUSHL   R0                                      ;
                              0C    AE  9F 001CB     PUSHAB  FRAC_STR_DESC                           ; 3411
                              18    AE  9F 001CE     PUSHAB  ZERO_STR_DESC                           ;
                    00000000'       EF  9F 001D1     PUSHAB  DOT_DESC                                ;
                                52  D4 001D7        CLRL    R2                                      ; 3414
                                61  D5 001D9        TSTL    (R1)                                    ;
                                0B  18 001DB        BGEQ    26$                                     ;
                                52  D6 001DD        INCL    R2                                      ;
                    50 00000000'    EF  9E 001DF     MOVAB   MINUS_DESC, R0                          ;
                                13  11 001E6        BRB     28$                                     ;
                09                  58  E9 001E8 26$:    BLBC    FLAGS, 27$                          ; 3419
                    50 00000000'    EF  9E 001EB     MOVAB   NULL_DESC, R0                           ;
                                07  11 001F2        BRB     28$                                     ;
                    50 00000000'    EF  9E 001F4 27$:    MOVAB   BLANK_DESC, R0                     ;
                                50  DD 001FB 28$:    PUSHL   R0                                      ;
                                53  DD 001FD        PUSHL   BAS_OUT_STR                             ; 3411
          00000000G             00  06  FB 001FF     CALLS   #6, STR$CONCAT                          ;
                                57  D0 00206        MOVL    R0, STATUS                              ;
                                0C  58  E9 00209     BLBC    FLAGS, 30$                              ; 3432
                                05  52  E9 0020C     BLBC    R2, 29$                                 ; 3435
                                50  01  D0 0020F     MOVL    #1, R0                                  ;
                                    07  11 00212     BRB     31$                                     ;
                                50  D4 00214 29$:    CLRL    R0                                      ;
                                    03  11 00216     BRB     31$                                     ;
                    50              02  D0 00218 30$:    MOVL    #2, R0                              ; 3432
                    50              56  C0 0021B 31$:    ADDL2   NO_DIGITS, R0                       ; 3442
                    50              55  C2 0021E        SUBL2   R5, R0                              ;
                    64          01  A0  9E 00221     MOVAB   1(R0), (BAS_OUT_STR_LEN)                ;
                    26          04  AE  D1 00225 32$:    CMPL    LARGEST_EXP, #38                    ; 3445
                                0A  15 00229        BLEQ    33$                                     ;
                                18  AE  9F 0022B     PUSHAB  ZEROES_DESC                             ; 3447
          00000000G             00  01  FB 0022E     CALLS   #1, STR$FREE1_DX                        ;
                    50              57  D0 00235 33$:    MOVL    STATUS, R0                          ; 3449
                    5E              20  C0 00238        ADDL2   #32, SP                             ; 3450
                            01E4    8F  BA 0023B        POPR    #^M<R2,R5,R6,R7,R8>                 ;
                                    05  00 0023F        RSB                                         ;

; Routine Size:  576 bytes,    Routine Base:  _BAS$CODE + 0ECE
```

```
;  2811          3451  1 ROUTINE FANCY_F_FORM_11 (
;  2812          3452  1         NO_INT_DIGITS,
;  2813          3453  1         NO_FRAC_DIGITS,
;  2814          3454  1         NO_DIGITS,
;  2815          3455  1         FLAGS,
;  2816          3456  1         KERNEL_BLOCK,
;  2817          3457  1         CURRENCY,
;  2818          3458  1         DIGIT_SEP,
;  2819          3459  1         RADIX_PT,
;  2820          3460  1         BAS_OUT_STR,
;  2821          3461  1         BAS_OUT_STR_LEN,
;  2822          3462  1         LARGEST_EXP
;  2823          3463  1     ) : JSB_FORMAT_A11 NOVALUE =
;  2824          3464  1
;  2825          3465  1 !++
;  2826          3466  1 ! FUNCTIONAL DESCRIPTION:
;  2827          3467  1 !
;  2828          3468  1 !       Do all of the Print Using formatting for F format.
;  2829          3469  1 !
;  2830          3470  1 ! FORMAL PARAMETERS:
;  2831          3471  1 !
;  2832          3472  1 !       NO_INT_DIGITS.rlu.r     number of integer digits in output string
;  2833          3473  1 !       NO_FRAC_DIGITS.rlu.r    number of fraction digits in output string
;  2834          3474  1 !       NO_DIGITS.rlu.v         number of significant digits in value
;  2835          3475  1 !       FLAGS.rlu.v             no flags are applicable
;  2836          3476  1 !       KERNEL_BLOCK.mz.r       parameter block from kernel routine
;  2837          3477  1 !       CURRENCY.rt.dx          currency symbol
;  2838          3478  1 !       DIGIT_SEP.rt.dx         digit group separator
;  2839          3479  1 !       RADIX_PT.rt.dx          radix point
;  2840          3480  1 !       BAS_OUT_STR_LEN.wlu.v   length of output string (handy for fixed length
;  2841          3481  1 !                               output strings
;  2842          3482  1 !       BAS_OUT_STR.wt.dx       the output string
;  2843          3483  1 !       LARGEST_EXP.rlu.v       largest exponent possible for data type
;  2844          3484  1 !
;  2845          3485  1 ! IMPLICIT INPUTS:
;  2846          3486  1 !
;  2847          3487  1 !       NONE
;  2848          3488  1 !
;  2849          3489  1 ! IMPLICIT OUTPUTS:
;  2850          3490  1 !
;  2851          3491  1 !       NONE
;  2852          3492  1 !
;  2853          3493  1 ! COMPLETION CODES:
;  2854          3494  1 !
;  2855          3495  1 !       NONE
;  2856          3496  1 !
;  2857          3497  1 ! SIDE EFFECTS:
;  2858          3498  1 !
;  2859          3499  1 !       NONE
;  2860          3500  1 !
;  2861          3501  1 !--
;  2862          3502  1
;  2863          3503  2     BEGIN
;  2864          3504  2
;  2865          3505  2     MAP
;  2866          3506  2         CURRENCY : REF BLOCK [, BYTE],
;  2867          3507  2         DIGIT_SEP : REF BLOCK [, BYTE],
```

```
; 2868          3508  2          RADIX_PT : REF BLOCK [, BYTE],
; 2869          3509  2          BAS_OUT_STR_LEN : REF VECTOR,
; 2870          3510  2          BAS_OUT_STR : REF VECTOR [2, LONG],
; 2871          3511  2          KERNEL_BLOCK : REF BLOCK [K_KERNEL_BLK_SZ, BYTE];
; 2872          3512  2
; 2873          3513  2      LOCAL
; 2874          3514  2          ZEROES_PTR,
; 2875          3515  2          BLANKS_PTR,
; 2876          3516  2          STAR_PTR,
; 2877          3517  2          STAR_STR_DESC : BLOCK [8, BYTE],        ! desc. for asterisks
; 2878          3518  2          BLANKS_STR_DESC : BLOCK [8, BYTE],      ! desc. for leading spaces
; 2879          3519  2          ZERO_STR_DESC : BLOCK [8, BYTE],        ! desc. for trailing zeroes
; 2880          3520  2          INT_STR_DESC : BLOCK [8, BYTE],         ! desc. for integer portion of number
; 2881          3521  2          ZEROES_DESC : BLOCK [8, BYTE],          ! place to create string of zeroes
; 2882          3522  2          BLANKS_DESC : BLOCK [8, BYTE],          ! place to create string of blanks
; 2883          3523  2          STAR_DESC : BLOCK [8, BYTE],            ! place to create string of stars
; 2884          3524  2          LEADING_STR_DESC : BLOCK [8, BYTE],     ! place to create leading zeroes
; 2885          3525  2          LENGTH,                                 ! length of leading blanks or
; 2886          3526  2                                                  !  zeroes
; 2887          3527  2          CR_DESC : BLOCK [8, BYTE],              ! desc. for CR
; 2888          3528  2          DR_DESC : BLOCK [8, BYTE],              ! desc. for DR
; 2889          3529  2          CR_STR,
; 2890          3530  2          DR_STR;
; 2891          3531  2
; 2892          3532  2      INITIALIZE_DESC;
; 2893          3533  2
; 2894          3534  2      CR_STR = %ASCIZ'CR';
; 2895          3535  2      DR_STR = %ASCIZ'DR';
; 2896          3536  2
; 2897          3537  2      IF (.FLAGS AND V_CR_DR) NEQ 0 THEN
; 2898          3538  3      BEGIN
; 2899          3539  3          CR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
; 2900          3540  3          CR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
; 2901          3541  3          CR_DESC [DSC$W_LENGTH] = 2;
; 2902          3542  3          CR_DESC [DSC$A_POINTER] = CR_STR;
; 2903          3543  3          DR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
; 2904          3544  3          DR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
; 2905          3545  3          DR_DESC [DSC$W_LENGTH] = 2;
; 2906          3546  3          DR_DESC [DSC$A_POINTER] = DR_STR;
; 2907          3547  2      END;
; 2908          3548  2
; 2909          3549  2 !+
; 2910          3550  2 ! The largest possible exponent for single and double floating point is
; 2911          3551  2 ! 38.  G and H floating, however, have largest exponents of 308 and 4932
; 2912          3552  2 ! respectively.  NUM1$ must be able to print out all these characters but
; 2913          3553  2 ! the user who doesn't want all these characters should not have to pay the
; 2914          3554  2 ! performance penalty.  Therefore, fill characters (zero, star, blank) are
; 2915          3555  2 ! initialized to size 38 and then re-initialized here only if necessary.
; 2916          3556  2 !-
; 2917          3557  2
; 2918          3558  3      IF (.LARGEST_EXP GTR K_RANGE)
; 2919          3559  2      THEN
; 2920          3560  3          BEGIN
; 2921          3561  3          LOCAL
; 2922          3562  3              ZERO : BYTE,
; 2923          3563  3              STAR : BYTE,
; 2924          3564  3              BLANK : BYTE;
```

```
; 2925          3565  3
; 2926          3566  3          ZEROES_DESC [DSC$W_LENGTH] = .LARGEST_EXP;
; 2927          3567  3          ZEROES_DESC [DSC$B_CLASS] = DSC$K_CLASS_D;
; 2928          3568  3          ZEROES_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
; 2929          3569  3          ZEROES_DESC [DSC$A_POINTER] = 0;        ! STR$DUPL_CHAR will alloc space
; 2930          3570  3
; 2931          3571  3          ZERO = %X'30';
; 2932          3572  3          STR$DUPL_CHAR (ZEROES_DESC, LARGEST_EXP, ZERO);
; 2933          3573  3                                                  ! create string of zeroes
; 2934          3574  3          ZEROES_PTR = .ZEROES_DESC [DSC$A_POINTER];
; 2935          3575  3
; 2936          3576  3          BLANKS_DESC [DSC$W_LENGTH] = .LARGEST_EXP;
; 2937          3577  3          BLANKS_DESC [DSC$B_CLASS] = DSC$K_CLASS_D;
; 2938          3578  3          BLANKS_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
; 2939          3579  3          BLANKS_DESC [DSC$A_POINTER] = 0;
; 2940          3580  3
; 2941          3581  3          BLANK = %X'20';
; 2942          3582  3          STR$DUPL_CHAR (BLANKS_DESC, LARGEST_EXP, BLANK);
; 2943          3583  3                                                  ! create string of blanks
; 2944          3584  3          BLANKS_PTR = .BLANKS_DESC [DSC$A_POINTER];
; 2945          3585  3
; 2946          3586  3          STAR_DESC [DSC$W_LENGTH] = .LARGEST_EXP;
; 2947          3587  3          STAR_DESC [DSC$B_CLASS] = DSC$K_CLASS_D;
; 2948          3588  3          STAR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
; 2949          3589  3          STAR_DESC [DSC$A_POINTER] = 0;
; 2950          3590  3
; 2951          3591  3          STAR = %X'2A';
; 2952          3592  3          STR$DUPL_CHAR (STAR_DESC, LARGEST_EXP, STAR);
; 2953          3593  3                                                  ! create string of asterisks
; 2954          3594  3          STAR_PTR = .STAR_DESC [DSC$A_POINTER];
; 2955          3595  3
; 2956          3596  3          END
; 2957          3597  2      ELSE
; 2958          3598  2          ZEROES_PTR = ZEROES;                    ! use string of 38 zeroes
; 2959          3599  2          BLANKS_PTR = BLANKS;
; 2960          3600  2          STAR_PTR = STAR;
; 2961          3601  2
; 2962          3602  3      IF (.KERNEL_BLOCK [DEC_EXP] GTR 0)
; 2963          3603  2      THEN
; 2964          3604  2  !+
; 2965          3605  2  ! The number is greater than or equal to 1
; 2966          3606  2  !-
; 2967          3607  2
; 2968          3608  3          IF (.KERNEL_BLOCK [DEC_EXP] GEQ .NO_DIGITS)
; 2969          3609  2          THEN
; 2970          3610  3              BEGIN
; 2971          3611  3  !+
; 2972          3612  3  ! This entire number consists only of integer part.  Put the number
; 2973          3613  3  ! returned by the conversion routine and optionally some trailing
; 2974          3614  3  ! zeroes into the return string.
; 2975          3615  3  !-
; 2976          3616  3                  INT_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
; 2977          3617  3                  INT_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_D;
; 2978          3616  4                  IF NOT ((.FLAGS AND V_COMMA) NEQ 0 AND
; 2979          3619  4                          (.FLAGS AND V_LEADING_ZERO) NEQ 0)
; 2980          3620  3                  THEN
; 2981          3621  4                      BEGIN
```

```
;  2982          3622  4                INT_STR_DESC [DSC$A_POINTER] = .KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET];
;  2983          3623  4                INT_STR_DESC [DSC$W_LENGTH] = .NO_DIGITS;
;  2984          3624  3                END;
;  2985          3625  3            ZERO_STR_DESC [DSC$W_LENGTH] = .KERNEL_BLOCK [DEC_EXP] - .NO_DIGITS;
;  2986          3626  3            ZERO_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_D;
;  2987          3627  3            ZERO_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  2988          3628  3            ZERO_STR_DESC [DSC$A_POINTER] = .ZEROES_PTR;
;  2989          3629  3            BLANKS_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  2990          3630  3            BLANKS_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  2991          3631  3            LEADING_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  2992          3632  3            LEADING_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  2993          3633  3            LEADING_STR_DESC [DSC$A_POINTER] = .ZEROES_PTR;
;  2994          3634  3            LEADING_STR_DESC [DSC$W_LENGTH] = 0;                ! this will be changed
;  2995          3635  3                                                                ! later
;  2996          3636  3        !+
;  2997          3637  3        ! Take advantage of the fact that negative numbers have to have trailing sign if
;  2998          3638  3        ! the currency symbol is expected to float.
;  2999          3639  3        !-
;  3000          3640  3            LENGTH = .NO_INT_DIGITS - .KERNEL_BLOCK [DEC_EXP] -
;  3001          3641  4            BEGIN
;  3002          3642  4
;  3003          3643  5            IF ((.FLAGS AND V_TRAILING_SIGN) NEQ 0)
;  3004          3644  4            THEN
;  3005          3645  4                0                               ! trailing sign
;  3006          3646  4            ELSE
;  3007          3647  5                BEGIN
;  3008          3648  5
;  3009          3649  5                IF (.KERNEL_BLOCK [SIGN] LSS 0) AND
;  3010          3650  5                   (.FLAGS AND V_CR_DR) EQL 0
;  3011          3651  5                THEN 1
;  3012          3652  5                ELSE 0
;  3013          3653  5
;  3014          3654  5                END
;  3015          3655  5
;  3016          3656  4            END
;  3017          3657  3            -                                               ! negative number - leading sign
;  3018          3658  4            BEGIN
;  3019          3659  4
;  3020          3660  4            IF ((.FLAGS AND V_CURRENCY) NEQ 0) THEN .CURRENCY [DSC$W_LENGTH] ELSE 0
;  3021          3661  4
;  3022          3662  4            END
;  3023          3663  3            -                                               ! currency symbol
;  3024          3664  4            BEGIN
;  3025          3665  4
;  3026          3666  5            IF ((.FLAGS AND V_COMMA) NEQ 0)
;  3027          3667  4            THEN
;  3028          3668  4                (.KERNEL_BLOCK [DEC_EXP] - 1)/3*.DIGIT_SEP [DSC$W_LENGTH]
;  3029          3669  4            ELSE
;  3030          3670  4                0
;  3031          3671  4
;  3032          3672  3            END;                                            ! commas
;  3033          3673  3
;  3034          3674  3        !+
;  3035          3675  3        ! If leading zeroes should be printed, then replace the blanks descriptor
;  3036          3676  3        ! with a leading zeroes descriptor.
;  3037          3677  3        !-
;  3038          3678  3
```

```
;  3039          3679  3              IF (.FLAGS AND V_LEADING_ZERO) NEQ 0
;  3040          3680  3              THEN
;  3041          3681  3                  LEADING_STR_DESC [DSC$W_LENGTH] = .LENGTH
;  3042          3682  3              ELSE
;  3043          3683  3                  BLANKS_STR_DESC [DSC$W_LENGTH] = .LENGTH;
;  3044          3684  3
;  3045          3685  3              BLANKS_STR_DESC [DSC$A_POINTER] = .BLANKS_PTR;
;  3046          3686  3              STAR_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3047          3687  3              STAR_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3048          3688  3              STAR_STR_DESC [DSC$W_LENGTH] = .NO_INT_DIGITS - .KERNEL_BLOCK [DEC_EXP] -
;  3049          3689  4              BEGIN
;  3050          3690  4
;  3051          3691  5              IF ((.FLAGS AND V_TRAILING_SIGN) NEQ 0)
;  3052          3692  4              THEN
;  3053          3693  4                  0                               ! trailing sign
;  3054          3694  4              ELSE
;  3055          3695  5                  BEGIN
;  3056          3696  5
;  3057          3697  5                  IF (.KERNEL_BLOCK [SIGN] LSS 0) AND
;  3058          3698  5                     (.FLAGS AND V_CR_DR) EQL 0
;  3059          3699  5                  THEN 1
;  3060          3700  5                  ELSE 0
;  3061          3701  5
;  3062          3702  5                  END
;  3063          3703  5
;  3064          3704  4              END
;  3065          3705  3              -                                   ! negative number - leading sign
;  3066          3706  4              BEGIN
;  3067          3707  4
;  3068          3708  4              IF ((.FLAGS AND V_COMMA) NEQ 0) THEN (.KERNEL_BLOCK [DEC_EXP] - 1)/3 ELSE 0
;  3069          3709  4
;  3070          3710  3              END;                                ! commas
;  3071          3711  3              STAR_STR_DESC [DSC$A_POINTER] = .STAR_PTR;
;  3072          3712  3
;  3073          3713  4              IF ((.FLAGS AND V_COMMA) EQL 0)
;  3074          3714  3              THEN
;  3075          3715  4                  BEGIN
;  3076          3716  4  !+
;  3077          3717  4  ! There are no commas in this number
;  3078          3718  4  !-
;  3079          3719  4
;  3080          3720  4                  IF (.FLAGS AND V_CURRENCY) NEQ 0 AND
;  3081          3721  4                     (.FLAGS AND V_LEADING_ZERO) NEQ 0
;  3082          3722  4                  THEN
;  3083          3723  5                      BEGIN
;  3084          3724  5                      STR$CONCAT (BAS_OUT_STR [0], .CURRENCY,
;  3085          3725  6                          (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN NULL_DESC
;  3086          3726  8                          ELSE IF (.KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR)
;  3087          3727  5                          EQL 0) THEN MINUS_DESC ELSE NULL_DESC),
;  3088          3728  5                          LEADING_STR_DESC, INT_STR_DESC, ZERO_STR_DESC)
;  3089          3729  5                      END
;  3090          3730  4                  ELSE
;  3091          3731  5                      BEGIN
;  3092          3732  5                      STR$CONCAT (BAS_OUT_STR [0],
;  3093          3733  5                          (IF (.FLAGS AND V_STAR) NEQ 0 THEN STAR_STR_DESC ELSE NULL_DESC),
;  3094          3734  6                          (IF (.FLAGS AND V_STAR) NEQ 0
;  3095          3735  6                          THEN
```

```
;3096     3736  6                          NULL_DESC
;3097     3737  6                      ELSE
;3098     3738  6  !+
;3099     3739  6  ! Stars and leading zeroes are mutually exclusive.  If no star fill, then
;3100     3740  6  ! check if leading zeroes should be printed.
;3101     3741  6  !-
;3102     3742  7                          (IF (.FLAGS AND V_LEADING_ZERO) NEQ 0
;3103     3743  7                          THEN NULL_DESC                          ! print leading zeroes
;3104     3744  5                          ELSE BLANKS_STR_DESC),                  ! no leading zeroes
;3105     3745  5                      (IF (.FLAGS AND V_CURRENCY) NEQ 0 THEN .CURRENCY ELSE NULL_DESC),
;3106     3746  6                      (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN NULL_DESC
;3107     3747  8                      ELSE IF (.KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR)
;3108     3748  5                      EQL 0) THEN MINUS_DESC ELSE NULL_DESC),
;3109     3749  6                      (IF (.FLAGS AND V_LEADING_ZERO) NEQ 0 THEN
;3110     3750  5                      LEADING_STR_DESC ELSE NULL_DESC), INT_STR_DESC, ZERO_STR_DESC)
;3111     3751  5                  END
;3112     3752  4              END                                         ! end of no commas
;3113     3753  3          ELSE
;3114     3754  3  !+
;3115     3755  3  ! There are commas in this number and we need to do some looping on the call to
;3116     3756  3  ! concatenate.  The processing got too complex to express in a single call.
;3117     3757  3  !-
;3118     3758  4              BEGIN
;3119     3759  4
;3120     3760  4              LOCAL
;3121     3761  4                  COUNTER,
;3122     3762  4                  FIRST_GROUP,
;3123     3763  4                  TEMP_INT_DESC : BLOCK [8, BYTE];
;3124     3764  4  !+
;3125     3765  4  ! Concatenate the leading zeroes to the integer, if necessary, so commas will
;3126     3766  4  ! be inserted at the proper places.  The length and pointer fields of INT_STR_DESC
;3127     3767  4  ! were not intialized earlier so that we can fill them in here.
;3128     3768  4  !-
;3129     3769  4              IF (.FLAGS AND V_LEADING_ZERO) NEQ 0
;3130     3770  4              THEN
;3131     3771  5                  BEGIN
;3132     3772  5
;3133     3773  5                  TEMP_INT_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;3134     3774  5                  TEMP_INT_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;3135     3775  5                  TEMP_INT_DESC [DSC$W_LENGTH] = .KERNEL_BLOCK [DEC_EXP];
;3136     3776  5                  TEMP_INT_DESC [DSC$A_POINTER] = .KERNEL_BLOCK [STRING_ADDR] +
;3137     3777  5                                                  .KERNEL_BLOCK [OFFSET];
;3138     3778  5                  INT_STR_DESC [DSC$W_LENGTH] = 0;
;3139     3779  5                  INT_STR_DESC [DSC$A_POINTER] = 0;
;3140     3780  5  !+
;3141     3781  5  ! Adjust leading zeroes length if extra commas will be required.
;3142     3782  5  !-
;3143     3783  5                  LEADING_STR_DESC [DSC$W_LENGTH] = .LEADING_STR_DESC [DSC$W_LENGTH]
;3144     3784  5                      - ((.LEADING_STR_DESC [DSC$W_LENGTH] - 1)/3 * .DIGIT_SEP [DSC$W_LENGTH]);
;3145     3785  5  !*** 1-048    LEADING_STR_DESC [DSC$W_LENGTH] = .LEADING_STR_DESC [DSC$W_LENGTH]
;3146     3786  5  !*** 1-048- ((.LEADING_STR_DESC [DSC$W_LENGTH] - 1)/3 * .DIGIT_SEP [DSC$W_LENGTH]) - 1;
;3147     3787  5                  STR$CONCAT (INT_STR_DESC, LEADING_STR_DESC, TEMP_INT_DESC);
;3148     3788  4                  END;
;3149     3789  4
;3150     3790  7              FIRST_GROUP = (((IF (.FLAGS AND V_LEADING_ZERO) NEQ 0 THEN .LEADING_STR_DESC [DSC$W_LENGTH]
;3151     3791  4                  ELSE 0) + .KERNEL_BLOCK [DEC_EXP] - 1) MOD 3) + 1;
;3152     3792  4              STR$CONCAT (BAS_OUT_STR [0],
```

```
;  3153          3793  4                (IF (.FLAGS AND V_STAR) NEQ 0 THEN STAR_STR_DESC ELSE NULL_DESC),
;  3154          3794  5                (IF (.FLAGS AND V_STAR) NEQ 0 THEN NULL_DESC
;  3155          3795  6                ELSE (IF (.FLAGS AND V_LEADING_ZERO) NEQ 0 THEN NULL_DESC
;  3156          3796  4                ELSE BLANKS_STR_DESC)),
;  3157          3797  4                (IF (.FLAGS AND V_CURRENCY) NEQ 0 THEN .CURRENCY ELSE NULL_DESC),
;  3158          3798  5                (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN NULL_DESC
;  3159          3799  6                ELSE IF (.KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR) EQL 0)
;  3160          3800  4                THEN MINUS_DESC ELSE NULL_DESC),
;  3161          3801  5                (INT_STR_DESC [DSC$W_LENGTH] = MIN (.FIRST_GROUP,
;  3162          3802  4                .LEADING_STR_DESC [DSC$W_LENGTH] + .NO_DIGITS); INT_STR_DESC));
;  3163          3803  5            COUNTER = (IF (.FLAGS AND V_LEADING_ZERO) NEQ 0 THEN .LEADING_STR_DESC [DSC$W_LENGTH]
;  3164          3804  4                ELSE 0) + .NO_DIGITS - .INT_STR_DESC [DSC$W_LENGTH];
;  3165          3805  4            INT_STR_DESC [DSC$A_POINTER] = .INT_STR_DESC [DSC$A_POINTER] + .INT_STR_DESC [DSC$W_LENGTH];
;  3166          3806  4
;  3167          3807  4            WHILE .COUNTER GTR 0 DO
;  3168          3808  5                BEGIN
;  3169          3809  5                INT_STR_DESC [DSC$W_LENGTH] = MIN (.COUNTER, 3);
;  3170          3810  5                STR$CONCAT (BAS_OUT_STR [0], BAS_OUT_STR [0], .DIGIT_SEP, INT_STR_DESC);
;  3171          3811  5                INT_STR_DESC [DSC$A_POINTER] = .INT_STR_DESC [DSC$A_POINTER] + .INT_STR_DESC [
;  3172          3812  5                    DSC$W_LENGTH];
;  3173          3813  5                COUNTER = .COUNTER - .INT_STR_DESC [DSC$W_LENGTH];
;  3174          3814  4                END;                            ! WHILE loop
;  3175          3815  4
;  3176          3816  4            ZERO_STR_DESC [DSC$W_LENGTH] = (.KERNEL_BLOCK [DEC_EXP] - .NO_DIGITS) MOD 3;
;  3177          3817  4
;  3178          3818  5            IF (.ZERO_STR_DESC [DSC$W_LENGTH] NEQ 0)
;  3179          3819  4            THEN
;  3180          3820  4                STR$CONCAT (BAS_OUT_STR [0], BAS_OUT_STR [0],
;  3181          3821  4                    ZERO_STR_DESC);
;  3182          3822  4
;  3183          3823  4                ZERO_STR_DESC [DSC$W_LENGTH] = 3;
;  3184          3824  4                COUNTER = (.KERNEL_BLOCK [DEC_EXP] - .NO_DIGITS)/3;
;  3185          3825  4
;  3186          3826  4                WHILE (.COUNTER GTR 0) DO
;  3187          3827  5                    BEGIN
;  3188          3828  5                    STR$CONCAT (BAS_OUT_STR [0], BAS_OUT_STR [0], .DIGIT_SEP, ZERO_STR_DESC);
;  3189          3829  5                    COUNTER = .COUNTER - 1;
;  3190          3830  4                    END;                        ! WHILE loop
;  3191          3831  4
;  3192          3832  3            END;                                ! of comma processing
;  3193          3833  3
;  3194          3834  3        !+
;  3195          3835  3        ! Common ending for comma and non-comma.
;  3196          3836  3        !-
;  3197          3837  3
;  3198          3838  3            ZERO_STR_DESC [DSC$W_LENGTH] = .NO_FRAC_DIGITS;
;  3199          3839  3            STR$CONCAT (BAS_OUT_STR [0], BAS_OUT_STR [0],
;  3200          3840  3                (IF (.FLAGS AND V_PERIOD) NEQ 0 THEN .RADIX_PT ELSE NULL_DESC), ZERO_STR_DESC,
;  3201          3841  4                (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN
;  3202          3842  5                    IF (.KERNEL_BLOCK [SIGN] LSS 0)
;  3203          3843  3                    THEN MINUS_DESC ELSE BLANK_DESC ELSE NULL_DESC),
;  3204          3844  4                    (IF (.FLAGS AND V_CR_DR) NEQ 0
;  3205          3845  5                    THEN (IF .KERNEL_BLOCK [SIGN] LSS 0 THEN CR_DESC ELSE DR_DESC)
;  3206          3846  3                    ELSE NULL_DESC));
;  3207          3847  3
;  3208          3848  4            BAS_OUT_STR_LEN [0] = (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN 1
;  3209          3849  5            ELSE (IF .KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR) EQL 0
```

```
;  3210     3850  4              THEN 1 ELSE 0))
;  3211     3851  3              + .KERNEL_BLOCK [DEC_EXP]
;  3212     3852  4              + (IF (.FLAGS AND V_COMMA) NEQ 0
;  3213     3853  5              THEN ((.KERNEL_BLOCK [DEC_EXP] - 1)/3*.DIGIT_SEP [DSC$W_LENGTH])    ! adjust for commas
;  3214     3854  4              ELSE 0)
;  3215     3855  4              + (IF (.FLAGS AND V_PERIOD) NEQ 0 THEN .RADIX_PT [DSC$W_LENGTH] ELSE 0)
;  3216     3856  3              + ZERO_STR_DESC [DSC$W_LENGTH];
;  3217     3857  3
;  3218     3858  3          END
;  3219     3859  2          ELSE
;  3220     3860  2              BEGIN
;  3221     3861  3      !+
;  3222     3862  3      ! This number has both integer and fractional portions.
;  3223     3863  3      !-
;  3224     3864  3
;  3225     3865  3              LOCAL
;  3226     3866  3                  FRAC_STR_DESC : BLOCK [8, BYTE];
;  3227     3867  3
;  3228     3868  3              INT_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_D;
;  3229     3869  3              INT_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3230     3870  4              IF NOT ((.FLAGS AND V_COMMA) NEQ 0 AND
;  3231     3871  4                 (.FLAGS AND V_LEADING_ZERO) NEQ 0)
;  3232     3872  3              THEN
;  3233     3873  4                  BEGIN
;  3234     3874  4                  INT_STR_DESC [DSC$W_LENGTH] = .KERNEL_BLOCK [DEC_EXP];
;  3235     3875  4                  INT_STR_DESC [DSC$A_POINTER] = .KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET];
;  3236     3876  3                  END;
;  3237     3877  3              FRAC_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3238     3878  3              FRAC_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3239     3879  3              FRAC_STR_DESC [DSC$W_LENGTH] = .NO_DIGITS - .KERNEL_BLOCK [DEC_EXP];
;  3240     3880  3              FRAC_STR_DESC [DSC$A_POINTER] = .KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET]
;  3241     3881  3                                            + .KERNEL_BLOCK [DEC_EXP];
;  3242     3882  3              ZERO_STR_DESC [DSC$W_LENGTH] = .NO_FRAC_DIGITS - .FRAC_STR_DESC [DSC$W_LENGTH];
;  3243     3883  3              ZERO_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3244     3884  3              ZERO_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3245     3885  3              ZERO_STR_DESC [DSC$A_POINTER] = .ZEROES_PTR;
;  3246     3886  3              BLANKS_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3247     3887  3              BLANKS_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3248     3888  3              LEADING_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3249     3689  3              LEADING_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3250     3890  3              LEADING_STR_DESC [DSC$A_POINTER] = .ZEROES_PTR;
;  3251     3891  3              LEADING_STR_DESC [DSC$W_LENGTH] = 0;                          ! to be changed
;  3252     3892  3                                                                            ! later
;  3253     3893  3      !+
;  3254     3894  3      ! Take advantage of the fact that negative numbers have to have trailing sign if
;  3255     3895  3      ! the currency symbol is expected to float.
;  3256     3896  3      !-
;  3257     3897  3              LENGTH = .NO_INT_DIGITS - .KERNEL_BLOCK [DEC_EXP] -
;  3258     3898  4              BEGIN
;  3259     3899  4
;  3260     3900  5              IF ((.FLAGS AND V_TRAILING_SIGN) NEQ 0)
;  3261     3901  4              THEN
;  3262     3902  4                  0                                 ! trailing sign
;  3263     3903  4              ELSE
;  3264     3904  5                  BEGIN
;  3265     3905  5
;  3266     3906  5                  IF (.KERNEL_BLOCK [SIGN] LSS 0) AND (.FLAGS AND V_CR_DR) EQL 0
```

```
;  3267     3907  5                THEN 1
;  3268     3908  5                ELSE 0
;  3269     3909  5
;  3270     3910  5                END
;  3271     3911  5
;  3272     3912  4            END
;  3273     3913  3            -                                   ! negative number - leading sign
;  3274     3914  4            BEGIN
;  3275     3915  4
;  3276     3916  4            IF ((.FLAGS AND V_CURRENCY) NEQ 0) THEN .CURRENCY [DSC$W_LENGTH] ELSE 0
;  3277     3917  4
;  3278     3918  4            END
;  3279     3919  3            -                                   ! currency symbol
;  3280     3920  4            BEGIN
;  3281     3921  4
;  3282     3922  5            IF ((.FLAGS AND V_COMMA) NEQ 0)
;  3283     3923  4            THEN
;  3284     3924  4                (.KERNEL_BLOCK [DEC_EXP] - 1)/3*.DIGIT_SEP [DSC$W_LENGTH]
;  3285     3925  4            ELSE
;  3286     3926  4                0
;  3287     3927  4
;  3288     3928  3            END;                                ! commas
;  3289     3929  3
;  3290     3930  3 !+
;  3291     3931  3 ! If leading zeroes should be printed, then replace the blanks descriptor
;  3292     3932  3 ! with a leading zeroes descriptor.
;  3293     3933  3 !-
;  3294     3934  3            IF (.FLAGS AND V_LEADING_ZERO) NEQ 0
;  3295     3935  3            THEN
;  3296     3936  3                LEADING_STR_DESC [DSC$W_LENGTH] = .LENGTH
;  3297     3937  3            ELSE
;  3298     3938  3                BLANKS_STR_DESC [DSC$W_LENGTH] = .LENGTH;
;  3299     3939  3
;  3300     3940  3            BLANKS_STR_DESC [DSC$A_POINTER] = .BLANKS_PTR;
;  3301     3941  3            STAR_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3302     3942  3            STAR_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3303     3943  3            STAR_STR_DESC [DSC$W_LENGTH] = .NO_INT_DIGITS - .KERNEL_BLOCK [DEC_EXP] -
;  3304     3944  4            BEGIN
;  3305     3945  4
;  3306     3946  5            IF ((.FLAGS AND V_TRAILING_SIGN) NEQ 0)
;  3307     3947  4            THEN
;  3308     3948  4                0                               ! trailing sign
;  3309     3949  4            ELSE
;  3310     3950  5                BEGIN
;  3311     3951  5
;  3312     3952  5                IF .KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR) EQL 0
;  3313     3953  5                THEN 1
;  3314     3954  5                ELSE 0
;  3315     3955  5
;  3316     3956  5                END
;  3317     3957  5
;  3318     3958  4            END
;  3319     3959  3            -                                   ! negative number - leading sign
;  3320     3960  4            BEGIN
;  3321     3961  4
;  3322     3962  5            IF ((.FLAGS AND V_COMMA) NEQ 0)
;  3323     3963  4            THEN
```

```
;  3324          3964  4                (.KERNEL_BLOCK [DEC_EXP] - 1)/3*.DIGIT_SEP [DSC$W_LENGTH]
;  3325          3965  4            ELSE
;  3326          3966  4                0
;  3327          3967  4
;  3328          3968  3            END;                                    ! commas
;  3329          3969  3            STAR_STR_DESC [DSC$A_POINTER] = .STAR_PTR;
;  3330          3970  3
;  3331          3971  4            IF ((.FLAGS AND V_COMMA) EQL 0)
;  3332          3972  3            THEN
;  3333          3973  4                BEGIN
;  3334          3974  4 !+
;  3335          3975  4 ! There are no commas required.
;  3336          3976  4 !-
;  3337          3977  4
;  3338          3978  4                IF (.FLAGS AND V_CURRENCY) NEQ 0 AND
;  3339          3979  4                   (.FLAGS AND V_LEADING_ZERO) NEQ 0
;  3340          3980  4                THEN
;  3341          3981  5                    BEGIN
;  3342          3982  5                    STR$CONCAT (BAS_OUT_STR [0], .CURRENCY,
;  3343          3983  6                        (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN NULL_DESC
;  3344          3984  6                        ELSE IF .KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR) EQL 0
;  3345          3985  5                        THEN MINUS_DESC ELSE NULL_DESC), LEADING_STR_DESC,
;  3346          3986  5                        INT_STR_DESC)
;  3347          3987  5                    END
;  3348          3988  4                ELSE
;  3349          3989  5                    BEGIN
;  3350          3990  5                    STR$CONCAT (BAS_OUT_STR [0],
;  3351          3991  5                        (IF (.FLAGS AND V_STAR) NEQ 0 THEN STAR_STR_DESC ELSE NULL_DESC),
;  3352          3992  6                        (IF (.FLAGS AND V_STAR) NEQ 0 THEN NULL_DESC ELSE
;  3353          3993  7                        (IF (.FLAGS AND V_LEADING_ZERO) NEQ 0 THEN NULL_DESC ELSE
;  3354          3994  5                        BLANKS_STR_DESC)),
;  3355          3995  5                        (IF (.FLAGS AND V_CURRENCY) NEQ 0 THEN .CURRENCY ELSE NULL_DESC),
;  3356          3996  6                        (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN NULL_DESC
;  3357          3997  7                        ELSE (IF .KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR) EQL 0
;  3358          3998  5                            THEN MINUS_DESC ELSE NULL_DESC)),
;  3359          3999  6                        (IF (.FLAGS AND V_LEADING_ZERO) NEQ 0 THEN LEADING_STR_DESC
;  3360          4000  5                        ELSE NULL_DESC), INT_STR_DESC)
;  3361          4001  5                    END
;  3362          4002  4                END
;  3363          4003  4
;  3364          4004  3            ELSE
;  3365          4005  3 !+
;  3366          4006  3 ! This number should have commas in it.  Do all of the fancy figuring and looping.
;  3367          4007  3 !-
;  3368          4008  4                BEGIN
;  3369          4009  4
;  3370          4010  4                LOCAL
;  3371          4011  4                    COUNTER,
;  3372          4012  4                    FIRST_GROUP,
;  3373          4013  4                    TEMP_INT_DESC : BLOCK [8, BYTE];
;  3374          4014  4 !+
;  3375          4015  4 ! Concatenate the leading zeroes to the integer, if necessary, so commas will
;  3376          4016  4 ! be inserted at the proper places.  The length and pointer fields of INT_STR_DESC
;  3377          4017  4 ! were not intialized earlier so that we can fill them in here.
;  3378          4018  4 !-
;  3379          4019  4                IF (.FLAGS AND V_LEADING_ZERO) NEQ 0
;  3380          4020  4                THEN
```

```
;  3381          4021  5                    BEGIN
;  3382          4022  5
;  3383          4023  5                    TEMP_INT_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3384          4024  5                    TEMP_INT_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3385          4025  5                    TEMP_INT_DESC [DSC$W_LENGTH] = .KERNEL_BLOCK [DEC_EXP];
;  3386          4026  5                    TEMP_INT_DESC [DSC$A_POINTER] = .KERNEL_BLOCK [STRING_ADDR] +
;  3387          4027  5                                                    .KERNEL_BLOCK [OFFSET];
;  3388          4028  5                    INT_STR_DESC [DSC$W_LENGTH] = 0;
;  3389          4029  5                    INT_STR_DESC [DSC$A_POINTER] = 0;
;  3390          4030  5 !+
;  3391          4031  5 ! Adjust leading zeroes length if extra commas will be required.
;  3392          4032  5 !-
;  3393          4033  5                    LEADING_STR_DESC [DSC$W_LENGTH] = .LEADING_STR_DESC [DSC$W_LENGTH]
;  3394          4034  5                        - (.LEADING_STR_DESC [DSC$W_LENGTH] - 1)/3 * .DIGIT_SEP [DSC$W_LENGTH]);
;  3395          4035  5 !*** 1-048    LEADING_STR_DESC [DSC$W_LENGTH] = .LEADING_STR_DESC [DSC$W_LENGTH]
;  3396          4036  5 !*** 1-048- ((.LEADING_STR_DESC [DSC$W_LENGTH] - 1)/3 * .DIGIT_SEP [DSC$W_LENGTH]) - 1;
;  3397          4037  5                    STR$CONCAT (INT_STR_DESC, LEADING_STR_DESC, TEMP_INT_DESC);
;  3398          4038  4                    END;
;  3399          4039  4
;  3400          4040  7                FIRST_GROUP = (((IF (.FLAGS AND V_LEADING_ZERO) NEQ 0 THEN .LEADING_STR_DESC [DSC$W_LENGTH]
;  3401          4041  4                    ELSE 0) + .KERNEL_BLOCK [DEC_EXP] - 1) MOD 3) + 1;
;  3402          4042  4                STR$CONCAT (BAS_OUT_STR [0],
;  3403          4043  4                    (IF (.FLAGS AND V_STAR) NEQ 0 THEN STAR_STR_DESC ELSE NULL_DESC),
;  3404          4044  7                    (IF (.FLAGS AND V_STAR) NEQ 0 THEN NULL_DESC ELSE (IF (.FLAGS
;  3405          4045  4                    AND V_LEADING_ZERO) NEQ 0 THEN NULL_DESC ELSE BLANKS_STR_DESC)),
;  3406          4046  4                    (IF (.FLAGS AND V_CURRENCY) NEQ 0 THEN .CURRENCY ELSE NULL_DESC),
;  3407          4047  5                    (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN NULL_DESC ELSE
;  3408          4048  5                    IF .KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR) EQL 0
;  3409          4049  4                        THEN MINUS_DESC ELSE NULL_DESC),
;  3410          4050  5                    (INT_STR_DESC [DSC$W_LENGTH] = MIN (.FIRST_GROUP,
;  3411          4051  4                        .LEADING_STR_DESC [DSC$W_LENGTH] + .NO_DIGITS); INT_STR_DESC));
;  3412          4052  5                COUNTER = (IF (.FLAGS AND V_LEADING_ZERO) NEQ 0 THEN .LEADING_STR_DESC [DSC$W_LENGTH]
;  3413          4053  4                    ELSE 0) + .KERNEL_BLOCK [DEC_EXP] - .INT_STR_DESC [DSC$W_LENGTH];
;  3414          4054  4                INT_STR_DESC [DSC$A_POINTER] = .INT_STR_DESC [DSC$A_POINTER] + .INT_STR_DESC [DSC$W_LENGTH];
;  3415          4055  4
;  3416          4056  4                WHILE .COUNTER GTR 0 DO
;  3417          4057  5                    BEGIN
;  3418          4058  5                    INT_STR_DESC [DSC$W_LENGTH] = MIN (.COUNTER, 3);
;  3419          4059  5                    STR$CONCAT (BAS_OUT_STR [0], BAS_OUT_STR [0], .DIGIT_SEP, INT_STR_DESC);
;  3420          4060  5                    INT_STR_DESC [DSC$A_POINTER] = .INT_STR_DESC [DSC$A_POINTER] + .INT_STR_DESC [
;  3421          4061  5                        DSC$W_LENGTH];
;  3422          4062  5                    COUNTER = .COUNTER - .INT_STR_DESC [DSC$W_LENGTH];
;  3423          4063  4                    END;                            ! WHILE loop
;  3424          4064  4
;  3425          4065  4
;  3426          4066  3                END;
;  3427          4067  3
;  3428          4068  3 !+
;  3429          4069  3 ! And now for a common ending for both parts (comma and non-comma).
;  3430          4070  3 !-
;  3431          4071  3            STR$CONCAT (BAS_OUT_STR [0], BAS_OUT_STR [0], .RADIX_PT, FRAC_STR_DESC, ZERO_STR_DESC,
;  3432          4072  5                (IF ((.FLAGS AND V_TRAILING_SIGN) NEQ 0)
;  3433          4073  6                THEN (IF (.KERNEL_BLOCK [SIGN] LSS 0)
;  3434          4074  3                THEN MINUS_DESC ELSE BLANK_DESC) ELSE NULL_DESC),
;  3435          4075  5                (IF (.FLAGS AND V_CR_DR) NEQ 0 THEN (IF .KERNEL_BLOCK [SIGN] LSS 0
;  3436          4076  3                THEN CR_DESC ELSE DR_DESC) ELSE NULL_DESC));
;  3437          4077  3            BAS_OUT_STR_LEN [0] =
```

```
;  3438     4078  4              BEGIN
;  3439     4079  4
;  3440     4080  5              IF ((.FLAGS AND V_TRAILING_SIGN) NEQ 0)
;  3441     4081  4              THEN
;  3442     4082  4                  1                                   ! trailing sign
;  3443     4083  4              ELSE
;  3444     4084  5                  BEGIN
;  3445     4085  5
;  3446     4086  6                  IF (.KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR) EQL 0)
;  3447     4087  5                  THEN 1
;  3448     4088  5                  ELSE 0
;  3449     4089  5
;  3450     4090  5                  END
;  3451     4091  5
;  3452     4092  4              END
;  3453     4093  3              + .NO_INT_DIGITS + .NO_FRAC_DIGITS +
;  3454     4094  4              BEGIN
;  3455     4095  4
;  3456     4096  5              IF ((.FLAGS AND V_COMMA) NEQ 0)
;  3457     4097  4              THEN
;  3458     4098  4                  (.KERNEL_BLOCK [DEC_EXP] - 1)/3*.DIGIT_SEP [DSC$W_LENGTH]       ! adjust for commas
;  3459     4099  4              ELSE
;  3460     4100  4                  0
;  3461     4101  4
;  3462     4102  4              END
;  3463     4103  3              + .RADIX_PT [DSC$W_LENGTH];          ! decimal point
;  3464     4104  3              END
;  3465     4105  3
;  3466     4106  2      ELSE
;  3467     4107  3          BEGIN
;  3468     4108  3 !+
;  3469     4109  3 ! This is a purely fractional number which should be put out in F format
;  3470     4110  3 !-
;  3471     4111  3
;  3472     4112  3          LOCAL
;  3473     4113  3              FRAC_STR_DESC : BLOCK [8, BYTE];    ! desc. for fraction
;  3474     4114  3
;  3475     4115  3 !+
;  3476     4116  3 ! This is the number of zeroes to the right of the decimal point and to the left
;  3477     4117  3 ! of the first non-zero digit.
;  3478     4118  3 !-
;  3479     4119  3          ZERO_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3480     4120  3          ZERO_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3481     4121  3          ZERO_STR_DESC [DSC$W_LENGTH] = -.KERNEL_BLOCK [DEC_EXP];
;  3482     4122  3          ZERO_STR_DESC [DSC$A_POINTER] = .ZEROES_PTR;
;  3483     4123  3 !+
;  3484     4124  3 ! Put at most one zero to the left of the decimal point
;  3485     4125  3 !-
;  3486     4126  3          INT_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3487     4127  3          INT_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3488     4128  3          INT_STR_DESC [DSC$W_LENGTH] = MIN (1,
;  3489     4129  4              .NO_INT_DIGITS - (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN 0      ! trailing sign
;  3490     4130  5              ELSE (IF .KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR) EQL 0
;  3491     4131  3              THEN 1 ELSE 0)));
;  3492     4132  3          INT_STR_DESC [DSC$A_POINTER] = .ZEROES_PTR;
;  3493     4133  3          STAR_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3494     4134  3          STAR_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
```

```
;  3495          4135  3          STAR_STR_DESC [DSC$W_LENGTH] = .NO_INT_DIGITS - .INT_STR_DESC [DSC$W_LENGTH] -
;  3496          4136  4          BEGIN
;  3497          4137  4
;  3498          4138  5          IF ((.FLAGS AND V_TRAILING_SIGN) NEQ 0)
;  3499          4139  4          THEN
;  3500          4140  4              0
;  3501          4141  4          ELSE
;  3502          4142  5              BEGIN
;  3503          4143  5
;  3504          4144  6              IF (.KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR) EQL 0)
;  3505          4145  5              THEN 1
;  3506          4146  5              ELSE 0
;  3507          4147  5
;  3508          4148  5              END
;  3509          4149  5
;  3510          4150  3          END;
;  3511          4151  3          STAR_STR_DESC [DSC$A_POINTER] = .STAR_PTR;
;  3512          4152  3          LEADING_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3513          4153  3          LEADING_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3514          4154  3          LEADING_STR_DESC [DSC$A_POINTER] = .ZEROES_PTR;
;  3515          4155  3          LEADING_STR_DESC [DSC$W_LENGTH] = 0;                    ! set later in routine
;  3516          4156  3          BLANKS_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3517          4157  3          BLANKS_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3518          4158  5          LENGTH = .NO_INT_DIGITS - .INT_STR_DESC [DSC$W_LENGTH] - (IF (.FLAGS AND
;  3519          4159  5              V_TRAILING_SIGN) NEQ 0 THEN 0 ELSE (IF .KERNEL_BLOCK [SIGN] LSS 0
;  3520          4160  5              AND (.FLAGS AND V_CR_DR) EQL 0 THEN 1 ELSE 0)) - (IF (.FLAGS
;  3521          4161  3              AND V_CURRENCY) NEQ 0 THEN 1 ELSE 0);
;  3522          4162  3
;  3523          4163  3 !+
;  3524          4164  3 ! If leading zeroes should be printed, then replace the blanks descriptor with
;  3525          4165  3 ! a leading zeroes descriptor.
;  3526          4166  3 !-
;  3527          4167  3
;  3528          4168  3          IF (.FLAGS AND V_LEADING_ZERO) NEQ 0
;  3529          4169  3          THEN
;  3530          4170  3              LEADING_STR_DESC [DSC$W_LENGTH] = .LENGTH
;  3531          4171  3          ELSE
;  3532          4172  3              BLANKS_STR_DESC [DSC$W_LENGTH] = .LENGTH;
;  3533          4173  3          BLANKS_STR_DESC [DSC$A_POINTER] = .BLANKS_PTR;
;  3534          4174  3          FRAC_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3535          4175  3          FRAC_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3536          4176  3          FRAC_STR_DESC [DSC$W_LENGTH] = .NO_DIGITS;
;  3537          4177  3          FRAC_STR_DESC [DSC$A_POINTER] = .KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET];
;  3538          4178  3          IF (.FLAGS AND V_CURRENCY) NEQ 0 AND
;  3539          4179  3             (.FLAGS AND V_LEADING_ZERO) NEQ 0
;  3540          4180  3          THEN
;  3541          4181  4              BEGIN
;  3542          4182  4              STR$CONCAT (BAS_OUT_STR [0], .CURRENCY,
;  3543          4183  5                  (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN NULL_DESC
;  3544          4184  5                  ELSE IF .KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR) EQL 0
;  3545          4185  4                  THEN MINUS_DESC ELSE NULL_DESC), LEADING_STR_DESC,
;  3546          4186  4                  INT_STR_DESC, .RADIX_PT, ZERO_STR_DESC, FRAC_STR_DESC,
;  3547          4187  6                  (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN (IF .KERNEL_BLOCK [SIGN]
;  3548          4188  4                  LSS 0 THEN MINUS_DESC ELSE BLANK_DESC) ELSE NULL_DESC),
;  3549          4189  6                  (IF (.FLAGS AND V_CR_DR) NEQ 0 THEN (IF .KERNEL_BLOCK [SIGN] LSS 0
;  3550          4190  4                  THEN CR_DESC ELSE DR_DESC) ELSE NULL_DESC))
;  3551          4191  4              END
```

```
;  3552          4192  3          ELSE
;  3553          4193  3              STR$CONCAT (BAS_OUT_STR [0],
;  3554          4194  3                  (IF (.FLAGS AND V_STAR) NEQ 0 THEN STAR_STR_DESC ELSE NULL_DESC),
;  3555          4195  4                  (IF (.FLAGS AND V_STAR) NEQ 0 THEN NULL_DESC ELSE
;  3556          4196  5                  (IF (.FLAGS AND V_LEADING_ZERO) NEQ 0 THEN NULL_DESC ELSE
;  3557          4197  3                  BLANKS_STR_DESC)),
;  3558          4198  3                  (IF (.FLAGS AND V_CURRENCY) NEQ 0 THEN .CURRENCY ELSE NULL_DESC),
;  3559          4199  4                  (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN NULL_DESC ELSE
;  3560          4200  5                  (IF .KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR) EQL 0
;  3561          4201  3                      THEN MINUS_DESC ELSE NULL_DESC)),
;  3562          4202  4                  (IF (.FLAGS AND V_LEADING_ZERO) NEQ 0 THEN LEADING_STR_DESC ELSE
;  3563          4203  3                  NULL_DESC), INT_STR_DESC, .RADIX_PT, ZERO_STR_DESC, FRAC_STR_DESC,
;  3564          4204  4              BEGIN
;  3565          4205  4
;  3566          4206  4              LOCAL
;  3567          4207  4                  TEMP_ZERO_DESC : BLOCK [8, BYTE];
;  3568          4208  4
;  3569          4209  4              TEMP_ZERO_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3570          4210  4              TEMP_ZERO_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3571          4211  4              TEMP_ZERO_DESC [DSC$W_LENGTH] = .NO_FRAC_DIGITS - .NO_DIGITS - .ZERO_STR_DESC [DSC$W_LENGTH];
;  3572          4212  4              TEMP_ZERO_DESC [DSC$A_POINTER] = .ZEROES_PTR;
;  3573          4213  4              TEMP_ZERO_DESC
;  3574          4214  4              END
;  3575          4215  3          ,
;  3576          4216  4              BEGIN
;  3577          4217  4
;  3578          4218  5              IF ((.FLAGS AND V_TRAILING_SIGN) NEQ 0)
;  3579          4219  4              THEN
;  3580          4220  5                  BEGIN
;  3581          4221  5
;  3582          4222  6                  IF (.KERNEL_BLOCK [SIGN] LSS 0)
;  3583          4223  5                  THEN MINUS_DESC
;  3584          4224  5                  ELSE BLANK_DESC
;  3585          4225  5
;  3586          4226  5                  END
;  3587          4227  4              ELSE
;  3588          4228  4                  NULL_DESC
;  3589          4229  4
;  3590          4230  4              END
;  3591          4231  3          ,
;  3592          4232  4              BEGIN
;  3593          4233  5              IF ((.FLAGS AND V_CR_DR) NEQ 0) THEN (IF .KERNEL_BLOCK [SIGN] LSS 0
;  3594          4234  4              THEN CR_DESC ELSE DR_DESC) ELSE NULL_DESC
;  3595          4235  4              END
;  3596          4236  3          );
;  3597          4237  4          BAS_OUT_STR_LEN [0] = (IF (.FLAGS AND V_TRAILING_SIGN) NEQ 0 THEN 1      ! trailing sign
;  3598          4238  5          ELSE (IF .KERNEL_BLOCK [SIGN] LSS 0 AND (.FLAGS AND V_CR_DR) EQL 0
;  3599          4239  3          THEN 1 ELSE 0)) + .NO_INT_DIGITS + .NO_FRAC_DIGITS +
;  3600          4240  3              .RADIX_PT [DSC$W_LENGTH];
;  3601          4241  2          END;
;  3602          4242  2
;  3603          4243  3      IF (.LARGEST_EXP GTR K_RANGE)
;  3604          4244  2      THEN
;  3605          4245  3          BEGIN
;  3606          4246  3          STR$FREE1_DX (ZEROES_DESC);
;  3607          4247  3          STR$FREE1_DX (BLANKS_DESC);
;  3608          4248  3          STR$FREE1_DX (STAR_DESC);
```

```
;  3609          4249  2          END;
;  3610          4250  2
;  3611          4251  1      END;


                                        ! Routine FANCY_F_FORMAT

                    0110E                       .BLKB   2
                2E  01110 P.ABI:        .ASCII  \.\                             ;
                    01111                       .BLKB   3
                2A  01114 P.ABJ:        .ASCII  \*\                             ;
                2A  01115               .ASCII  \*\                             ;
                2A  01116               .ASCII  \*\                             ;
                2A  01117               .ASCII  \*\                             ;
                2A  01118               .ASCII  \*\                             ;
                2A  01119               .ASCII  \*\                             ;
                2A  0111A               .ASCII  \*\                             ;
                2A  0111B               .ASCII  \*\                             ;
                2A  0111C               .ASCII  \*\                             ;
                2A  0111D               .ASCII  \*\                             ;
                2A  0111E               .ASCII  \*\                             ;
                2A  0111F               .ASCII  \*\                             ;
                2A  01120               .ASCII  \*\                             ;
                2A  01121               .ASCII  \*\                             ;
                2A  01122               .ASCII  \*\                             ;
                2A  01123               .ASCII  \*\                             ;
                2A  01124               .ASCII  \*\                             ;
                2A  01125               .ASCII  \*\                             ;
                2A  01126               .ASCII  \*\                             ;
                2A  01127               .ASCII  \*\                             ;
                2A  01128               .ASCII  \*\                             ;
                2A  01129               .ASCII  \*\                             ;
                2A  0112A               .ASCII  \*\                             ;
                2A  0112B               .ASCII  \*\                             ;
                2A  0112C               .ASCII  \*\                             ;
                2A  0112D               .ASCII  \*\                             ;
                2A  0112E               .ASCII  \*\                             ;
                2A  0112F               .ASCII  \*\                             ;
                2A  01130               .ASCII  \*\                             ;
                2A  01131               .ASCII  \*\                             ;
                2A  01132               .ASCII  \*\                             ;
                2A  01133               .ASCII  \*\                             ;
                2A  01134               .ASCII  \*\                             ;
                2A  01135               .ASCII  \*\                             ;
                2A  01136               .ASCII  \*\                             ;
                2A  01137               .ASCII  \*\                             ;
                2A  01138               .ASCII  \*\                             ;
                2A  01139               .ASCII  \*\                             ;
                    0113A                       .BLKB   2
                2D  0113C P.ABK:        .ASCII  \-\                             ;
                    0113D                       .BLKB   3
                20  01140 P.ABL:        .ASCII  \ \                             ;
                    01141                       .BLKB   3
                45  01144 P.ABM:        .ASCII  \E\                             ;
                    01145                       .BLKB   3
                20  01148 P.ABN:        .ASCII  \ \                             ;
                20  01149               .ASCII  \ \                             ;
                20  0114A               .ASCII  \ \                             ;
```

```
20  0114B           .ASCII  \ \
20  0114C           .ASCII  \ \
20  0114D           .ASCII  \ \
20  0114E           .ASCII  \ \
20  0114F           .ASCII  \ \
20  01150           .ASCII  \ \
20  01151           .ASCII  \ \
20  01152           .ASCII  \ \
20  01153           .ASCII  \ \
20  01154           .ASCII  \ \
20  01155           .ASCII  \ \
20  01156           .ASCII  \ \
20  01157           .ASCII  \ \
20  01158           .ASCII  \ \
20  01159           .ASCII  \ \
20  0115A           .ASCII  \ \
20  0115B           .ASCII  \ \
20  0115C           .ASCII  \ \
20  0115D           .ASCII  \ \
20  0115E           .ASCII  \ \
20  0115F           .ASCII  \ \
20  01160           .ASCII  \ \
20  01161           .ASCII  \ \
20  01162           .ASCII  \ \
20  01163           .ASCII  \ \
20  01164           .ASCII  \ \
20  01165           .ASCII  \ \
20  01166           .ASCII  \ \
20  01167           .ASCII  \ \
20  01168           .ASCII  \ \
20  01169           .ASCII  \ \
20  0116A           .ASCII  \ \
20  0116B           .ASCII  \ \
20  0116C           .ASCII  \ \
20  0116D           .ASCII  \ \
    0116E           .BLKB   2
30  01170 P.ABO:    .ASCII  \0\
30  01171           .ASCII  \0\
30  01172           .ASCII  \0\
30  01173           .ASCII  \0\
30  01174           .ASCII  \0\
30  01175           .ASCII  \0\
30  01176           .ASCII  \0\
30  01177           .ASCII  \0\
30  01178           .ASCII  \0\
30  01179           .ASCII  \0\
30  0117A           .ASCII  \0\
30  0117B           .ASCII  \0\
30  0117C           .ASCII  \0\
30  0117D           .ASCII  \0\
30  0117E           .ASCII  \0\
30  0117F           .ASCII  \0\
30  01180           .ASCII  \0\
30  01181           .ASCII  \0\
30  01182           .ASCII  \0\
30  01183           .ASCII  \0\
30  01184           .ASCII  \0\
```

```
                          30  01185           .ASCII  \0\                              ;
                          30  01186           .ASCII  \0\                              ;
                          30  01187           .ASCII  \0\                              ;
                          30  01188           .ASCII  \0\                              ;
                          30  01189           .ASCII  \0\                              ;
                          30  0118A           .ASCII  \0\                              ;
                          30  0118B           .ASCII  \0\                              ;
                          30  0118C           .ASCII  \0\                              ;
                          30  0118D           .ASCII  \0\                              ;
                          30  0118E           .ASCII  \0\                              ;
                          30  0118F           .ASCII  \0\                              ;
                          30  01190           .ASCII  \0\                              ;
                          30  01191           .ASCII  \0\                              ;
                          30  01192           .ASCII  \0\                              ;
                          30  01193           .ASCII  \0\                              ;
                          30  01194           .ASCII  \0\                              ;
                          30  01195           .ASCII  \0\                              ;

                                  DOT=                  P.ABI
                                  STAR=                 P.ABJ
                                  MINUS=                P.ABK
                                  BLANK=                P.ABL
                                  E=                    P.ABM
                                  BLANKS=               P.ABN
                                  ZEROES=               P.ABO

                  0CA4   8F   BB 00000 FANCY_F_FORM_11:
                                            PUSHR   #^M<R2,R5,R7,R10,R11>                ; 3451
            5E    FF50   CE   9E 00004      MOVAB   -176(SP), SP                         ;
      4C    AE           5A   D0 00009      MOVL    R10, LARGEST_EXP                     ;
      0C    AE           57   D0 0000D      MOVL    R7, RADIX_PT                         ;
      10    AE           55   D0 00011      MOVL    R5, CURRENCY                         ;
      04    AE           51   D0 00015      MOVL    R1, NO_FRAC_DIGITS                   ;
      08    AE           50   D0 00019      MOVL    R0, NO_INT_DIGITS                    ;
             00000000'   EF   D5 0001D      TSTL    E_D                                  ; 3530
                         24   12 00023      BNEQ    1$                                   ;
00000000'   EF    FF7D   CF   9E 00025      MOVAB   MINUS, M_D                           ;
00000000'   EF    FF48   CF   9E 0002E      MOVAB   DOT, D_D                             ;
00000000'   EF    FF6F   CF   9E 00037      MOVAB   BLANK, B_D                           ;
00000000'   EF    FF6A   CF   9E 00040      MOVAB   E, E_D                               ;
      38    AE    5243   8F   3C 00049 1$:  MOVZWL  #21059, CR_STR                       ; 3534
      3C    AE    5244   8F   3C 0004F      MOVZWL  #21060, DR_STR                       ; 3535
                    34   AE   D4 00055      CLRL    52(SP)                               ; 3537
 1D         53           0E   E1 00058      BBC     #14, FLAGS, 2$                       ;
                    34   AE   D6 0005C      INCL    52(SP)                               ;
      68    AE 010E0002  8F   D0 0005F      MOVL    #17694722, CR_DESC                   ; 3541
      6C    AE      38   AE   9E 00067      MOVAB   CR_STR, CR_DESC+4                    ; 3542
      60    AE 010E0002  8F   D0 0006C      MOVL    #17694722, DR_DESC                   ; 3545
      64    AE      3C   AE   9E 00074      MOVAB   DR_STR, DR_DESC+4                    ; 3546
            26      4C   AE   D1 00079 2$:  CMPL    LARGEST_EXP, #38                     ; 3558
                         03   14 0007D      BGTR    3$                                   ;
                       0081   31 0007F      BRW     4$                                   ;
    0088    CE      4C   AE   B0 00082 3$:  MOVW    LARGEST_EXP, ZEROES_DESC             ; 3566
    008A    CE    020E   8F   B0 00088      MOVW    #526, ZEROES_DESC+2                  ; 3568
                  008C   CE   D4 0008F      CLRL    ZEROES_DESC+4                        ; 3569
      40    AE           30   90 00093      MOVB    #48, ZERO                            ; 3571
```

```
                        40   AE   9F 00097          PUSHAB  ZERO                                    ; 3572
                        50   AE   9F 0009A          PUSHAB  LARGEST_EXP                             ;
                  0090  CE   9F 0009D              PUSHAB  ZEROES_DESC                             ;
    00000000G     00         03   FB 000A1          CALLS   #3, STR$DUPL_CHAR                       ;
                        5B   008C CE   D0 000A8     MOVL    ZEROES_DESC+4, ZEROES_PTR               ;
                0080    CE   4C AE   B0 000AD      MOVW    LARGEST_EXP, BLANKS_DESC                ; 3574
                0082    CE   020E 8F   B0 000B3    MOVW    #526, BLANKS_DESC+2                     ; 3576
                  0084  CE   D4 000BA              CLRL    BLANKS_DESC+4                           ; 3578
                    44  AE         20 90 000BE     MOVB    #32, BLANK                              ; 3579
                        44   AE   9F 000C2          PUSHAB  BLANK                                   ; 3581
                        50   AE   9F 000C5          PUSHAB  LARGEST_EXP                             ; 3582
                  0088  CE   9F 000C8              PUSHAB  BLANKS_DESC                             ;
    00000000G     00         03   FB 000CC          CALLS   #3, STR$DUPL_CHAR                       ;
                    18  AE   0084 CE   D0 000D3     MOVL    BLANKS_DESC+4, BLANKS_PTR               ; 3584
                    78  AE   4C AE   B0 000D9      MOVW    LARGEST_EXP, STAR_DESC                  ; 3586
                    7A  AE   020E 8F   B0 000DE    MOVW    #526, STAR_DESC+2                       ; 3588
                        7C   AE   D4 000E4          CLRL    STAR_DESC+4                             ; 3589
                    48  AE         2A 90 000E7     MOVB    #42, STAR                               ; 3591
                        48   AE   9F 000EB          PUSHAB  STAR                                    ; 3592
                        50   AE   9F 000EE          PUSHAB  LARGEST_EXP                             ;
                  0080  CE   9F 000F1              PUSHAB  STAR_DESC                               ;
    00000000G     00         03   FB 000F5          CALLS   #3, STR$DUPL_CHAR                       ;
                    14  AE   7C AE   D0 000FC      MOVL    STAR_DESC+4, STAR_PTR                   ; 3594
                        05   11 00101              BRB     5$                                      ; 3558
                        5B   FED3 CF   9E 00103 4$: MOVAB   ZEROES, ZEROES_PTR                      ; 3598
                    18  AE   FEA6 CF   9E 00108 5$: MOVAB   BLANKS, BLANKS_PTR                     ; 3599
                    14  AE   FE6C CF   9E 0010E    MOVAB   STAR, STAR_PTR                          ; 3600
                        30   AE   D4 00114          CLRL    48(SP)                                  ; 3643
    03                  53       01   E1 00117     BBC     #1, FLAGS, 6$                           ;
                        30   AE   D6 0011B          INCL    48(SP)                                  ;
                        28   AE   D4 0011E 6$:      CLRL    40(SP)                                  ; 3660
    03                  53       03   E1 00121     BBC     #3, FLAGS, 7$                           ;
                        28   AE   D6 00125          INCL    40(SP)                                  ;
                        24   AE   D4 00128 7$:      CLRL    36(SP)                                  ; 3669
    03                  53       0D   E1 0012B     BBC     #13, FLAGS, 8$                          ;
                        24   AE   D6 0012F          INCL    36(SP)                                  ;
                        5A   08   A4 D0 00132 8$:   MOVL    8(KERNEL_BLOCK), R10                    ; 3602
                                 03   14 00136      BGTR    9$                                      ;
                              078E   31 00138       BRW     140$                                    ;
                        2C   AE   D4 0013B 9$:      CLRL    44(SP)                                  ; 3618
    03                  53       02   E1 0013E     BBC     #2, FLAGS, 10$                          ;
                        2C   AE   D6 00142          INCL    44(SP)                                  ;
                        50   D4   00145 10$:        CLRL    R0                                      ; 3619
    02                  53       0D   E1 00147     BBC     #13, FLAGS, 11$                         ;
                        50   D6   0014B            INCL    R0                                      ;
                        51   2C   AE   D2 0014D 11$: MCOML   44(SP), R1                             ;
                        50   51   CA 00151          BICL2   R1, R0                                  ;
                        50   50   D2 00154          MCOML   R0, R0                                  ; 3618
                0A2     CE   0E   90 00157         MOVB    #14, BLANKS_STR_DESC+2                  ; 3629
                0AA     CE   0E   90 0015C         MOVB    #14, STAR_STR_DESC+2                    ; 3686
                        20   AE   D4 00161          CLRL    32(SP)                                  ; 3713
    03                  53       02   E0 00164     BBS     #2, FLAGS, 12$                          ;
                        20   AE   D6 00168          INCL    32(SP)                                  ;
                        52   5A   D1 0016B 12$:     CMPL    R10, NO_DIGITS                          ; 3608
                                 03   18 0016E      BGEQ    13$                                     ;
                              03D5   31 00170       BRW     80$                                     ;
                0092    CE   020E 8F   B0 00173 13$: MOVW    #526, INT_STR_DESC+2                    ; 3616
```

```
                        0D      50  E9 0017A          BLBC    R0, 14$                                         ; 3618
0094    CE      10      A4  04  A4  C1 0017D          ADDL3   4(KERNEL_BLOCK), 16(KERNEL_BLOCK), -            ; 3622
                                                              INT_STR_DESC+4
        0090    CE              52  B0 00185          MOVW    NO_DIGITS, INT_STR_DESC                         ; 3623
        57      5A              52  C3 0018A 14$:     SUBL3   NO_DIGITS, R10, R7                              ; 3625
        0098    CE              57  B0 0018E          MOVW    R7, ZERO_STR_DESC
        009A    CE      020E    8F  B0 00193          MOVW    #526, ZERO_STR_DESC+2                           ; 3627
        009C    CE              5B  D0 0019A          MOVL    ZEROES_PTR, ZERO_STR_DESC+4                     ; 3628
        00A3    CE              01  90 0019F          MOVB    #1, BLANKS_STR_DESC+3                           ; 3630
          70    AE  010E0000    8F  D0 001A4          MOVL    #17694720, LEADING_STR_DESC                     ; 3634
          74    AE              5B  D0 001AC          MOVL    ZEROES_PTR, LEADING_STR_DESC+4                  ; 3633
        50      08  AE          5A  C3 001B0          SUBL3   R10, NO_INT_DIGITS, R0                          ; 3640
                    0E      30  AE  E8 001B5          BLBS    48(SP), 15$                                     ; 3643
                            0C  A4  D5 001B9          TSTL    12(KERNEL_BLOCK)                                ; 3649
                                09  18 001BC          BGEQ    15$
        05              53      0E  E0 001BE          BBS     #14, FLAGS, 15$                                 ; 3650
                        51          01  D0 001C2      MOVL    #1, R1                                          ; 3649
                                    02  11 001C5      BRB     16$
                                    51  D4 001C7 15$: CLRL    R1
                        51          50  C2 001C9 16$: SUBL2   R0, R1                                          ; 3641
                        06      28  AE  E9 001CC      BLBC    40(SP), 17$                                     ; 3660
                        55      10  BE  3C 001D0      MOVZWL  @CURRENCY, R5
                                    02  11 001D4      BRB     18$
                                    55  D4 001D6 17$: CLRL    R5
                        51          55  C0 001D8 18$: ADDL2   R5, R1                                          ; 3657
                        11      2C  AE  E9 001DB      BLBC    44(SP), 19$                                     ; 3666
                        55      FF  AA  9E 001DF      MOVAB   -1(R10), R5                                     ; 3668
                        55          03  C6 001E3      DIVL2   #3, R5
                  1C    AE          66  3C 001E6      MOVZWL  (DIGIT_SEP), 28(SP)
                        55      1C  AE  C4 001EA      MULL2   28(SP), R5
                                    02  11 001EE      BRB     20$
                                    55  D4 001F0 19$: CLRL    R5                                              ; 3666
                        51          55  C0 001F2 20$: ADDL2   R5, R1                                          ; 3664
                        51          51  CE 001F5      MNEGL   R1, LENGTH                                      ; 3663
                        06      24  AE  E9 001F8      BLBC    36(SP), 21$                                     ; 3681
          70    AE                51  B0 001FC      MOVW    LENGTH, LEADING_STR_DESC
                                    05  11 00200      BRB     22$
        00A0    CE              51  B0 00202 21$:     MOVW    LENGTH, BLANKS_STR_DESC                         ; 3683
        00A4    CE      18      AE  D0 00207 22$:     MOVL    BLANKS_PTR, BLANKS_STR_DESC+4                   ; 3685
        00AB    CE              01  90 0020D          MOVB    #1, STAR_STR_DESC+3                             ; 3687
                    0E      30  AE  E8 00212          BLBS    48(SP), 23$                                     ; 3691
                            0C  A4  D5 00216          TSTL    12(KERNEL_BLOCK)                                ; 3697
                                09  18 00219          BGEQ    23$
    05              53          0E  E0 0021B          BBS     #14, FLAGS, 23$                                 ; 3698
                    55          01  D0 0021F          MOVL    #1, R5                                          ; 3697
                                02  11 00222          BRB     24$
                                55  D4 00224 23$:     CLRL    R5
                    55          50  C2 00226 24$:     SUBL2   R0, R5                                          ; 3689
                    09      2C  AE  E9 00229          BLBC    44(SP), 25$                                     ; 3708
                    50      FF  AA  9E 0022D          MOVAB   -1(R10), R0
                    50          03  C6 00231          DIVL2   #3, R0
                                02  11 00234          BRB     26$
                                50  D4 00236 25$:     CLRL    R0
                    55          50  C0 00238 26$:     ADDL2   R0, R5                                          ; 3706
        00A8    CE              55  AE 0023B          MNEGW   R5, STAR_STR_DESC                               ; 3705
        00AC    CE      14      AE  D0 00240          MOVL    STAR_PTR, STAR_STR_DESC+4                       ; 3711
                    03      20  AE  E8 00246          BLBS    32(SP), 27$                                     ; 3713
```

```
                        00CA  31 0024A          BRW     43$
                  3D      28  AE  E9 0024D 27$:  BLBC    40(SP), 30$                 ;3720
                  39      24  AE  E9 00251      BLBC    36(SP), 30$                 ;3721
                        0098  CE  9F 00255      PUSHAB  ZERO_STR_DESC               ;3724
                        0094  CE  9F 00259      PUSHAB  INT_STR_DESC
                          78  AE  9F 0025D      PUSHAB  LEADING_STR_DESC
                  12      3C  AE  E8 00260      BLBS    60(SP), 28$                 ;3725
                          0C  A4  D5 00264      TSTL    12(KERNEL_BLOCK)            ;3726
                              0D  18 00267      BGEQ    28$
              09          53      0E  E0 00269  BBS     #14, FLAGS, 28$             ;3727
                  50 00000000'    EF  9E 0026D  MOVAB   MINUS_DESC, R0              ;3726
                              07  11 00274      BRB     29$
                  50 00000000'    EF  9E 00276 28$:     MOVAB   NULL_DESC, R0
                              50  DD 0027D 29$:     PUSHL   R0
                          20  AE  DD 0027F      PUSHL   CURRENCY                    ;3724
                              58  DD 00282      PUSHL   BAS_OUT_STR
          00000000G       00      06  FB 00284  CALLS   #6, STR$CONCAT
                        0207  31 0028B          BRW     64$                         ;3723
                        0098  CE  9F 0028E 30$:  PUSHAB  ZERO_STR_DESC               ;3732
                        0094  CE  9F 00292      PUSHAB  INT_STR_DESC
                  06      2C  AE  E9 00296      BLBC    44(SP), 31$                 ;3749
                  50      78  AE  9E 0029A      MOVAB   LEADING_STR_DESC, R0
                              07  11 0029E      BRB     32$
                  50 00000000'    EF  9E 002A0 31$:     MOVAB   NULL_DESC, R0
                              50  DD 002A7 32$:     PUSHL   R0
                  12      3C  AE  E8 002A9      BLBS    60(SP), 33$
                          0C  A4  D5 002AD      TSTL    12(KERNEL_BLOCK)            ;3747
                              0D  18 002B0      BGEQ    33$
              09          53      0E  E0 002B2  BBS     #14, FLAGS, 33$             ;3748
                  50 00000000'    EF  9E 002B6  MOVAB   MINUS_DESC, R0              ;3747
                              07  11 002BD      BRB     34$
                  50 00000000'    EF  9E 002BF 33$:     MOVAB   NULL_DESC, R0
                              50  DD 002C6 34$:     PUSHL   R0
                  05      38  AE  E9 002C8      BLBC    56(SP), 35$                 ;3746
                          20  AE  DD 002CC      PUSHL   CURRENCY                    ;3745
                              09  11 002CF      BRB     36$
                  55 00000000'    EF  9E 002D1 35$:     MOVAB   NULL_DESC, R5
                              55  DD 002D8      PUSHL   R5
                              51  D4 002DA 36$:     CLRL    R1                      ;3734
              04          53      04  E1 002DC  BBC     #4, FLAGS, 37$
                              51  D6 002E0      INCL    R1
                              04  11 002E2      BRB     38$
                  09      38  AE  E9 002E4 37$:     BLBC    56(SP), 39$             ;3742
                  50 00000000'    EF  9E 002E8 38$:     MOVAB   NULL_DESC, R0
                              05  11 002EF      BRB     40$
                  50      00B4    CE  9E 002F1 39$:     MOVAB   BLANKS_STR_DESC, R0
                              50  DD 002F6 40$:     PUSHL   R0
                  07              51  E9 002F8      BLBC    R1, 41$                 ;3734
                  50      00C0    CE  9E 002FB      MOVAB   STAR_STR_DESC, R0       ;3733
                              07  11 00300      BRB     42$
                  50 00000000'    EF  9E 00302 41$:     MOVAB   NULL_DESC, R0
                              50  DD 00309 42$:     PUSHL   R0
                              58  DD 0030B      PUSHL   BAS_OUT_STR                 ;3732
          00000000G       00      08  FB 0030D  CALLS   #8, STR$CONCAT
                        017E  31 00314          BRW     64$                         ;3715
                  47      24  AE  E9 00317 43$:  BLBC    36(SP), 44$                 ;3769
                5A    AE      010E    8F  B0 0031B      MOVW    #270, TEMP_INT_DESC+2   ;3774
```

```
                    58   AE                 5A  B0 00321          MOVW    R10, TEMP_INT_DESC                      ; 3775
                5C  AE   10   A4      04    A4  C1 00325          ADDL3   4(KERNEL_BLOCK), 16(KERNEL_BLOCK), -    ; 3777
                                                                          TEMP_INT_DESC+4
                                    0090  CE  B4 0032C          CLRW    INT_STR_DESC                            ; 3778
                                    0094  CE  D4 00330          CLRL    INT_STR_DESC+4                          ; 3779
                         50         70    AE  3C 00334          MOVZWL  LEADING_STR_DESC, R0                    ; 3784
                                              50  D7 00338          DECL    R0
                         50                   03  C6 0033A          DIVL2   #3, R0
                         51                   66  3C 0033D          MOVZWL  (DIGIT_SEP), R1
                         50                   51  C4 00340          MULL2   R1, R0
                                    70    AE  50  A2 00343          SUBW2   R0, LEADING_STR_DESC
                                    58    AE  9F 00347          PUSHAB  TEMP_INT_DESC                           ; 3787
                                    74    AE  9F 0034A          PUSHAB  LEADING_STR_DESC
                                    0098  CE  9F 0034D          PUSHAB  INT_STR_DESC
                00000000G           00        03  FB 00351          CALLS   #3, STR$CONCAT
                         06         24    AE  E9 00358          BLBC    36(SP), 44$                             ; 3790
                         50         70    AE  3C 0035C          MOVZWL  LEADING_STR_DESC, R0
                                              02  11 00360          BRB     45$
                                              50  D4 00362 44$:     CLRL    R0
                         50                   5A  C0 00364 45$:     ADDL2   R10, R0
             7E FFFFFFFF 8F                   01  7A 00367          EMUL    #1, R0, #-1, -(SP)                      ; 3791
             50          50                   03  7B 00370          EDIV    #3, (SP)+, R0, R0
                                              50  D6 00375          INCL    FIRST_GROUP
                         51         70    AE  3C 00377          MOVZWL  LEADING_STR_DESC, R1                    ; 3802
                         51                   52  C0 0037B          ADDL2   NO_DIGITS, R1
                         51                   50  D1 0037E          CMPL    R0, R1
                                              03  15 00381          BLEQ    46$
                         50                   51  D0 00383          MOVL    R1, R0
                    0090 CE                   50  B0 00386 46$:     MOVW    R0, INT_STR_DESC                        ; 3801
                              0090  CE  9F 0038B          PUSHAB  INT_STR_DESC
                         12     34    AE  E8 0038F          BLBS    52(SP), 47$
                                0C    A4  D5 00393          TSTL    12(KERNEL_BLOCK)                        ; 3799
                                      0D  18 00396          BGEQ    47$
             09          53           0E  E0 00398          BBS     #14, FLAGS, 47$
                         50 00000000' EF  9E 0039C          MOVAB   MINUS_DESC, R0
                                      07  11 003A3          BRB     48$
                         50 00000000' EF  9E 003A5 47$:     MOVAB   NULL_DESC, R0
                                      50  DD 003AC 48$:     PUSHL   R0
                         05     30    AE  E9 003AE          BLBC    48(SP), 49$                             ; 3798
                                18    AE  DD 003B2          PUSHL   CURRENCY                                ; 3797
                                      09  11 003B5          BRB     50$
                         55 00000000' EF  9E 003B7 49$:     MOVAB   NULL_DESC, R5
                                      55  DD 003BE          PUSHL   R5
                                      51  D4 003C0 50$:     CLRL    R1                                      ; 3794
             04          53           04  E1 003C2          BBC     #4, FLAGS, 51$
                                      51  D6 003C6          INCL    R1
                                      04  11 003C8          BRB     52$
                         09     30    AE  E9 003CA 51$:     BLBC    48(SP), 53$                             ; 3795
                         50 00000000' EF  9E 003CE 52$:     MOVAB   NULL_DESC, R0
                                      05  11 003D5          BRB     54$
                         50     00AC  CE  9E 003D7 53$:     MOVAB   BLANKS_STR_DESC, R0
                                      50  DD 003DC 54$:     PUSHL   R0
                         07           51  E9 003DE          BLBC    R1, 55$                                 ; 3794
                         50     00B8  CE  9E 003E1          MOVAB   STAR_STR_DESC, R0                       ; 3793
                                      07  11 003E6          BRB     56$
                         50 00000000' EF  9E 003E8 55$:     MOVAB   NULL_DESC, R0
                                      50  DD 003EF 56$:     PUSHL   R0
```

```
                                                58  DD 003F1          PUSHL   BAS_OUT_STR                                  ; 3792
                    00000000G   00              06  FB 003F3          CALLS   #6, STR$CONCAT
                                06        24    AE  E9 003FA          BLBC    36(SP), 57$                                 ; 3803
                                50        70    AE  3C 003FE          MOVZWL  LEADING_STR_DESC, R0
                                                02  11 00402          BRB     58$
                                                50  D4 00404 57$:     CLRL    R0
                                50              52  C0 00406 58$:     ADDL2   NO_DIGITS, R0                               ; 3804
                                52        0090  CE  3C 00409          MOVZWL  INT_STR_DESC, COUNTER
                    52          50              52  C3 0040E          SUBL3   COUNTER, R0, COUNTER
                                50        0090  CE  3C 00412          MOVZWL  INT_STR_DESC, R0
                          0094  CE              50  C0 00417          ADDL2   R0, INT_STR_DESC+4
                                                52  D5 0041C 59$:     TSTL    COUNTER                                     ; 3807
                                                35  15 0041E          BLEQ    61$
                                50              52  D0 00420          MOVL    COUNTER, R0                                 ; 3809
                                03              50  D1 00423          CMPL    R0, #3
                                                03  15 00426          BLEQ    60$
                                50              03  D0 00428          MOVL    #3, R0
                          0090  CE              50  B0 0042B 60$:     MOVW    R0, INT_STR_DESC
                                          0090  CE  9F 00430          PUSHAB  INT_STR_DESC                                ; 3810
                                                56  DD 00434          PUSHL   DIGIT_SEP
                                                58  DD 00436          PUSHL   BAS_OUT_STR
                                                58  DD 00438          PUSHL   BAS_OUT_STR
                    00000000G   00              04  FB 0043A          CALLS   #4, STR$CONCAT
                                50        0090  CE  3C 00441          MOVZWL  INT_STR_DESC, R0                            ; 3811
                          0094  CE              50  C0 00446          ADDL2   R0, INT_STR_DESC+4
                                50        0090  CE  3C 0044B          MOVZWL  INT_STR_DESC, R0                            ; 3813
                                52              50  C2 00450          SUBL2   R0, COUNTER
                                                C7  11 00453          BRB     59$                                         ; 3807
          7E                    00              57  01  7A 00455 61$:     EMUL    #1, R7, #0, -(SP)                       ; 3816
          50                    50              8E  03  7B 0045A          EDIV    #3, (SP)+, R0, R0
                          0098  CE              50  B0 0045F          MOVW    R0, ZERO_STR_DESC
                                                0F  13 00464          BEQL    62$                                         ; 3818
                                          0098  CE  9F 00466          PUSHAB  ZERO_STR_DESC                               ; 3820
                                                58  DD 0046A          PUSHL   BAS_OUT_STR
                                                58  DD 0046C          PUSHL   BAS_OUT_STR
                    00000000G   00              03  FB 0046E          CALLS   #3, STR$CONCAT
                          0098  CE              03  B0 00475 62$:     MOVW    #3, ZERO_STR_DESC                           ; 3823
                    52          57              03  C7 0047A          DIVL3   #3, R7, COUNTER                             ; 3824
                                                15  15 0047E 63$:     BLEQ    64$                                         ; 3826
                                          0098  CE  9F 00480          PUSHAB  ZERO_STR_DESC                               ; 3828
                                                56  DD 00484          PUSHL   DIGIT_SEP
                                                58  DD 00486          PUSHL   BAS_OUT_STR
                                                58  DD 00488          PUSHL   BAS_OUT_STR
                    00000000G   00              04  FB 0048A          CALLS   #4, STR$CONCAT
                                                52  D7 00491          DECL    COUNTER                                     ; 3829
                                                E9  11 00493          BRB     63$                                         ; 3826
                          0098  CE        04    AE  B0 00495 64$:     MOVW    NO_FRAC_DIGITS, ZERO_STR_DESC               ; 3838
                                11        34    AE  E9 0049B          BLBC    52(SP), 66$                                 ; 3844
                                          0C    A4  D5 0049F          TSTL    12(KERNEL_BLOCK)                            ; 3845
                                                06  18 004A2          BGEQ    65$
                                50        68    AE  9E 004A4          MOVAB   CR_DESC, R0
                                                0D  11 004A8          BRB     67$
                                50        60    AE  9E 004AA 65$:     MOVAB   DR_DESC, R0
                                                07  11 004AE          BRB     67$
                                50 00000000'    EF  9E 004B0 66$:     MOVAB   NULL_DESC, R0                               ; 3844
                                                50  DD 004B7 67$:     PUSHL   R0
                                17        34    AE  E9 004B9          BLBC    52(SP), 69$                                 ;
```

```
                        0C   A4  D5 004BD          TSTL    12(KERNEL_BLOCK)                        ; 3842
                             09  18 004C0          BGEQ    68$                                     ;
                50 00000000'  EF  9E 004C2          MOVAB   MINUS_DESC, R0                          ;
                             10  11 004C9          BRB     70$                                     ;
                50 00000000'  EF  9E 004CB 68$:     MOVAB   BLANK_DESC, R0                          ;
                             07  11 004D2          BRB     70$                                     ;
                50 00000000'  EF  9E 004D4 69$:     MOVAB   NULL_DESC, R0                           ; 3841
                             50  DD 004DB 70$:     PUSHL   R0                                      ;
                        00A0  CE  9F 004DD          PUSHAB  ZERO_STR_DESC                           ; 3839
                             52  D4 004E1          CLRL    R2                                      ; 3840
              07             53  06  E1 004E3          BBC     #6, FLAGS, 71$                      ;
                             52  D6 004E7          INCL    R2                                      ;
                             18  AE  DD 004E9          PUSHL   RADIX_PT                            ;
                             09  11 004EC          BRB     72$                                     ;
                57 00000000'  EF  9E 004EE 71$:     MOVAB   NULL_DESC, R7                           ;
                             57  DD 004F5          PUSHL   R7                                      ;
                             58  DD 004F7 72$:     PUSHL   BAS_OUT_STR                             ; 3839
                             58  DD 004F9          PUSHL   BAS_OUT_STR                             ;
          00000000G          00  06  FB 004FB          CALLS   #6, STR$CONCAT                      ;
                             09  30  AE  E8 00502          BLBS    48(SP), 73$                     ; 3848
                             0C  A4  D5 00506          TSTL    12(KERNEL_BLOCK)                    ; 3849
                             09  18 00509          BGEQ    74$                                     ;
              05             53  0E  E0 0050B          BBS     #14, FLAGS, 74$                     ;
                             50  01  D0 0050F 73$:     MOVL    #1, R0                              ;
                             02  11 00512          BRB     75$                                     ;
                             50  D4 00514 74$:     CLRL    R0                                      ;
                             50  5A  C0 00516 75$:     ADDL2   R10, R0                             ; 3851
                             0F  2C  AE  E9 00519          BLBC    44(SP), 76$                     ; 3852
                             55  FF  AA  9E 0051D          MOVAB   -1(R10), R5                     ; 3853
                             55  03  C6 00521          DIVL2   #3, R5                              ;
                             51  66  3C 00524          MOVZWL  (DIGIT_SEP), R1                     ;
                             55  51  C4 00527          MULL2   R1, R5                              ;
                             02  11 0052A          BRB     77$                                     ;
                             55  D4 0052C 76$:     CLRL    R5                                      ; 3852
                             50  55  C0 0052E 77$:     ADDL2   R5, R0                              ;
                             06  52  E9 00531          BLBC    R2, 78$                             ; 3855
                             57  0C  BE  3C 00534          MOVZWL  @RADIX_PT, R7                   ;
                             02  11 00538          BRB     79$                                     ;
                             57  D4 0053A 78$:     CLRL    R7                                      ;
                             50  57  C0 0053C 79$:     ADDL2   R7, R0                              ;
                             69  0098  CE40  9E 0053F          MOVAB   ZERO_STR_DESC[R0], (BAS_OUT_STR_LEN)  ; 3856
                                 05F6  31 00545          BRW     188$                                ; 3608
                        0092  CE  020E  8F  B0 00548 80$:     MOVW    #526, INT_STR_DESC+2            ; 3869
                             0D  50  E9 0054F          BLBC    R0, 81$                             ; 3870
                        0090  CE  5A  B0 00552          MOVW    R10, INT_STR_DESC                   ; 3874
          0094  CE      10  A4  04  A4  C1 00557          ADDL3   4(KERNEL_BLOCK), 16(KERNEL_BLOCK), -  ; 3875
                                                          INT_STR_DESC+4                          ;
                        5A  AE  010E  8F  B0 0055F 81$:     MOVW    #270, FRAC_STR_DESC+2           ; 3878
             58  AE             52  5A  A3 00565          SUBW3   R10, NO_DIGITS, FRAC_STR_DESC   ; 3879
             1C  AE      10  A4  04  A4  C1 0056A          ADDL3   4(KERNEL_BLOCK), 16(KERNEL_BLOCK), 28(SP)  ; 3880
                        5C  AE  1C  BE4A  9E 00571          MOVAB   @28(SP)[R10], FRAC_STR_DESC+4   ; 3881
          0098  CE      04  AE  58  AE  A3 00577          SUBW3   FRAC_STR_DESC, NO_FRAC_DIGITS, -  ; 3882
                                                          ZERO_STR_DESC                           ;
                        009A  CE  010E  8F  B0 0057F          MOVW    #270, ZERO_STR_DESC+2           ; 3884
                        009C  CE  5B  D0 00586          MOVL    ZEROES_PTR, ZERO_STR_DESC+4     ; 3885
                        00A3  CE  01  90 0058B          MOVB    #1, BLANKS_STR_DESC+3           ; 3887
                          70  AE  010E0000  8F  D0 00590          MOVL    #17694720, LEADING_STR_DESC  ; 3891
```

```
                74   AE          5B  D0 00598          MOVL    ZEROES_PTR, LEADING_STR_DESC+4      ; 3890
          50    08   AE          5A  C3 0059C          SUBL3   R10, NO_INT_DIGITS, R0              ; 3897
                0E              30  AE  E8 005A1      BLBS    48(SP), 82$                         ; 3900
                              0C    A4  D5 005A5      TSTL    12(KERNEL_BLOCK)                    ; 3906
                                    09  18 005A8      BGEQ    82$                                 ;
          05        53              0E  E0 005AA      BBS     #14, FLAGS, 82$                     ;
                    57              01  D0 005AE      MOVL    #1, R7                              ;
                                    02  11 005B1      BRB     83$                                 ;
                                    57  D4 005B3 82$:  CLRL    R7                                  ;
                    57              50  C2 005B5 83$:  SUBL2   R0, R7                             ; 3898
                    06          28  AE  E9 005B8      BLBC    40(SP), 84$                         ; 3916
                    55          10  BE  3C 005BC      MOVZWL  @CURRENCY, R5                       ;
                                    02  11 005C0      BRB     85$                                 ;
                                    55  D4 005C2 84$:  CLRL    R5                                  ;
                    57              55  C0 005C4 85$:  ADDL2   R5, R7                             ; 3913
                    0F          2C  AE  E9 005C7      BLBC    44(SP), 86$                         ; 3922
                    55          FF  AA  9E 005CB      MOVAB   -1(R10), R5                         ; 3924
                    55              03  C6 005CF      DIVL2   #3, R5                              ;
                    6E              66  3C 005D2      MOVZWL  (DIGIT_SEP), (SP)                   ;
                    55              6E  C4 005D5      MULL2   (SP), R5                            ;
                                    02  11 005D8      BRB     87$                                 ;
                                    55  D4 005DA 86$:  CLRL    R5                                  ; 3922
                    57              55  C0 005DC 87$:  ADDL2   R5, R7                             ; 3920
                    51              57  CE 005DF      MNEGL   R7, LENGTH                          ; 3919
                    06          24  AE  E9 005E2      BLBC    36(SP), 88$                         ; 3936
                      70        AE      51  B0 005E6  MOVW    LENGTH, LEADING_STR_DESC            ;
                                        05  11 005EA  BRB     89$                                 ;
                  00A0  CE          51  B0 005EC 88$:  MOVW    LENGTH, BLANKS_STR_DESC             ; 3938
                  00A4  CE      18  AE  D0 005F1 89$:  MOVL    BLANKS_PTR, BLANKS_STR_DESC+4       ; 3940
                  00AB  CE          01  90 005F7      MOVB    #1, STAR_STR_DESC+3                 ; 3942
                                0E  30  AE  E8 005FC  BLBS    48(SP), 90$                         ; 3946
                                  0C    A4  D5 00600  TSTL    12(KERNEL_BLOCK)                    ; 3952
                                        09  18 00603  BGEQ    90$                                 ;
          05        53              0E  E0 00605      BBS     #14, FLAGS, 90$                     ;
                    51              01  D0 00609      MOVL    #1, R1                              ;
                                    02  11 0060C      BRB     91$                                 ;
                                    51  D4 0060E 90$:  CLRL    R1                                  ;
                    51              50  C2 00610 91$:  SUBL2   R0, R1                             ; 3944
                    0F          2C  AE  E9 00613      BLBC    44(SP), 92$                         ; 3962
                    50          FF  AA  9E 00617      MOVAB   -1(R10), R0                         ; 3964
                    50              03  C6 0061B      DIVL2   #3, R0                              ;
                    55              66  3C 0061E      MOVZWL  (DIGIT_SEP), R5                     ;
                    50              55  C4 00621      MULL2   R5, R0                              ;
                                    02  11 00624      BRB     93$                                 ;
                                    50  D4 00626 92$:  CLRL    R0                                  ; 3962
                    51              50  C0 00628 93$:  ADDL2   R0, R1                             ; 3960
              00A8  CE              51  AE 0062B      MNEGW   R1, STAR_STR_DESC                   ; 3959
              00AC  CE      14  AE  D0 00630          MOVL    STAR_PTR, STAR_STR_DESC+4           ; 3969
                    03      20  AE  E8 00636          BLBS    32(SP), 94$                         ; 3971
                            00C2  31 0063A            BRW     110$                                ;
                    39      28  AE  E9 0063D 94$:     BLBC    40(SP), 97$                         ; 3978
                    35      24  AE  E9 00641          BLBC    36(SP), 97$                         ; 3979
                          0090  CE  9F 00645          PUSHAB  INT_STR_DESC                        ; 3982
                            74  AE  9F 00649          PUSHAB  LEADING_STR_DESC                    ;
                    12      38  AE  E8 0064C          BLBS    56(SP), 95$                         ; 3983
                              0C    A4  D5 00650      TSTL    12(KERNEL_BLOCK)                    ; 3984
                                    0D  18 00653      BGEQ    95$                                 ;
```

```
                09              53                      0E  E0 00655            BBS     #14, FLAGS, 95$                         ;
                                50 00000000'            EF  9E 00659            MOVAB   MINUS_DESC, R0                          ;
                                                        07  11 00660            BRB     96$                                     ;
                                50 00000000'            EF  9E 00662 95$:       MOVAB   NULL_DESC, R0                           ;
                                                        50  DD 00669 96$:       PUSHL   R0                                      ;
                                        1C              AE  DD 0066B            PUSHL   CURRENCY                                ; 3982
                                                        58  DD 0066E            PUSHL   BAS_OUT_STR                             ;
                    00000000G   00                      05  FB 00670            CALLS   #5, STR$CONCAT                          ;
                                                    01C1    31 00677            BRW     128$                                    ; 3981
                                    0090                CE  9F 0067A 97$:       PUSHAB  INT_STR_DESC                            ; 3990
                                06      28              AE  E9 0067E            BLBC    40(SP), 98$                             ; 3999
                                50      74              AE  9E 00682            MOVAB   LEADING_STR_DESC, R0                    ;
                                                        07  11 00686            BRB     99$                                     ;
                                50 00000000'            EF  9E 00688 98$:       MOVAB   NULL_DESC, R0                           ;
                                                        50  DD 0068F 99$:       PUSHL   R0                                      ;
                                12      38              AE  E8 00691            BLBS    56(SP), 100$                            ;
                                        0C              A4  D5 00695            TSTL    12(KERNEL_BLOCK)                        ; 3997
                                                        0D  18 00698            BGEQ    100$                                    ;
                09              53                      0E  E0 0069A            BBS     #14, FLAGS, 100$                        ;
                                50 00000000'            EF  9E 0069E            MOVAB   MINUS_DESC, R0                          ;
                                                        07  11 006A5            BRB     101$                                    ;
                                50 00000000'            EF  9E 006A7 100$:      MOVAB   NULL_DESC, R0                           ;
                                                        50  DD 006AE 101$:      PUSHL   R0                                      ;
                                05      34              AE  E9 006B0            BLBC    52(SP), 102$                            ; 3996
                                        1C              AE  DD 006B4            PUSHL   CURRENCY                                ; 3995
                                                        09  11 006B7            BRB     103$                                    ;
                                55 00000000'            EF  9E 006B9 102$:      MOVAB   NULL_DESC, R5                           ;
                                                        55  DD 006C0            PUSHL   R5                                      ;
                                                        51  D4 006C2 103$:      CLRL    R1                                      ; 3992
                04              53                      04  E1 006C4            BBC     #4, FLAGS, 104$                         ;
                                                        51  D6 006C8            INCL    R1                                      ;
                                                        04  11 006CA            BRB     105$                                    ;
                                09      34              AE  E9 006CC 104$:      BLBC    52(SP), 106$                            ; 3993
                                50 00000000'            EF  9E 006D0 105$:      MOVAB   NULL_DESC, R0                           ;
                                                        05  11 006D7            BRB     107$                                    ;
                                50      00B0            CE  9E 006D9 106$:      MOVAB   BLANKS_STR_DESC, R0                     ;
                                                        50  DD 006DE 107$:      PUSHL   R0                                      ;
                                07                      51  E9 006E0            BLBC    R1, 108$                                ; 3992
                                50      00BC            CE  9E 006E3            MOVAB   STAR_STR_DESC, R0                       ; 3991
                                                        07  11 006E8            BRB     109$                                    ;
                                50 00000000'            EF  9E 006EA 108$:      MOVAB   NULL_DESC, R0                           ;
                                                        50  DD 006F1 109$:      PUSHL   R0                                      ;
                                                        58  DD 006F3            PUSHL   BAS_OUT_STR                             ; 3990
                    00000000G   00                      07  FB 006F5            CALLS   #7, STR$CONCAT                          ;
                                                    013C    31 006FC            BRW     128$                                    ; 3973
                                45      24              AE  E9 006FF 110$:      BLBC    36(SP), 111$                            ; 4019
                        52      AE      010E            8F  B0 00703            MOVW    #270, TEMP_INT_DESC+2                   ; 4024
                        50      AE                      5A  B0 00709            MOVW    R10, TEMP_INT_DESC                      ; 4025
                        54      AE      1C              AE  D0 0070D            MOVL    28(SP), TEMP_INT_DESC+4                 ; 4026
                                        0090            CE  B4 00712            CLRW    INT_STR_DESC                            ; 4028
                                        0094            CE  D4 00716            CLRL    INT_STR_DESC+4                          ; 4029
                                50      70              AE  3C 0071A            MOVZWL  LEADING_STR_DESC, R0                    ; 4034
                                                        50  D7 0071E            DECL    R0                                      ;
                                50                      03  C6 00720            DIVL2   #3, R0                                  ;
                                51                      66  3C 00723            MOVZWL  (DIGIT_SEP), R1                         ;
                                50                      51  C4 00726            MULL2   R1, R0                                  ;
                        70      AE                      50  A2 00729            SUBW2   R0, LEADING_STR_DESC                    ;
```

```
                                                50    AE  9F 0072D          PUSHAB  TEMP_INT_DESC                    ; 4037
                                                74    AE  9F 00730          PUSHAB  LEADING_STR_DESC                 ;
                                              0098    CE  9F 00733          PUSHAB  INT_STR_DESC                     ;
                                 00000000G    00        03  FB 00737          CALLS   #3, STR$CONCAT               ;
                                              06        24  AE  E9 0073E          BLBC    36(SP), 111$             ; 4040
                                              50        70  AE  3C 00742          MOVZWL  LEADING_STR_DESC, R0     ;
                                                            02  11 00746          BRB     112$                     ;
                                                            50  D4 00748 111$:     CLRL    R0                       ;
                                              50                50  5A  C0 0074A 112$:     ADDL2   R10, R0          ;
                    7E FFFFFFFF       8F      50                01  7A 0074D          EMUL    #1, R0, #-1, -(SP)       ; 4041
                    50                50      8E                03  7B 00756          EDIV    #3, (SP)+, R0, R0        ;
                                                                50  D6 0075B          INCL    FIRST_GROUP              ;
                                              51        70  AE  3C 0075D          MOVZWL  LEADING_STR_DESC, R1     ; 4051
                                              51                52  C0 00761          ADDL2   NO_DIGITS, R1            ;
                                              51                50  D1 00764          CMPL    R0, R1                   ;
                                                                03  15 00767          BLEQ    113$                     ;
                                              50                51  D0 00769          MOVL    R1, R0                   ;
                                     0090     CE                50  B0 0076C 113$:     MOVW    R0, INT_STR_DESC         ; 4050
                                              0090    CE  9F 00771          PUSHAB  INT_STR_DESC                     ;
                                              12        34  AE  E8 00775          BLBS    52(SP), 114$             ;
                                                        0C  A4  D5 00779          TSTL    12(KERNEL_BLOCK)         ; 4048
                                                            0D  18 0077C          BGEQ    114$                     ;
                                 09                     53      0E  E0 0077E          BBS     #14, FLAGS, 114$         ;
                                              50 00000000'      EF  9E 00782          MOVAB   MINUS_DESC, R0           ;
                                                            07  11 00789          BRB     115$                     ;
                                              50 00000000'      EF  9E 0078B 114$:     MOVAB   NULL_DESC, R0            ;
                                                            50  DD 00792 115$:     PUSHL   R0                       ;
                                              05        30  AE  E9 00794          BLBC    48(SP), 116$             ; 4047
                                                        18  AE  DD 00798          PUSHL   CURRENCY                 ; 4046
                                                            09  11 0079B          BRB     117$                     ;
                                              55 00000000'      EF  9E 0079D 116$:     MOVAB   NULL_DESC, R5            ;
                                                            55  DD 007A4          PUSHL   R5                       ;
                                                            51  D4 007A6 117$:     CLRL    R1                       ; 4044
                    04                        53              04  E1 007A8          BBC     #4, FLAGS, 118$          ;
                                                            51  D6 007AC          INCL    R1                       ;
                                                            04  11 007AE          BRB     119$                     ;
                                              09        30  AE  E9 007B0 118$:     BLBC    48(SP), 120$             ; 4045
                                              50 00000000'      EF  9E 007B4 119$:     MOVAB   NULL_DESC, R0            ; 4044
                                                            05  11 007BB          BRB     121$                     ;
                                              50        00AC    CE  9E 007BD 120$:     MOVAB   BLANKS_STR_DESC, R0      ;
                                                            50  DD 007C2 121$:     PUSHL   R0                       ;
                                              07                51  E9 007C4          BLBC    R1, 122$                 ;
                                              50        00B8    CE  9E 007C7          MOVAB   STAR_STR_DESC, R0        ; 4043
                                                            07  11 007CC          BRB     123$                     ;
                                              50 00000000'      EF  9E 007CE 122$:     MOVAB   NULL_DESC, R0            ;
                                                            50  DD 007D5 123$:     PUSHL   R0                       ;
                                                            58  DD 007D7          PUSHL   BAS_OUT_STR              ; 4042
                                 00000000G    00                06  FB 007D9          CALLS   #6, STR$CONCAT           ;
                                              06        24  AE  E9 007E0          BLBC    36(SP), 124$             ; 4052
                                              50        70  AE  3C 007E4          MOVZWL  LEADING_STR_DESC, R0     ;
                                                            02  11 007E8          BRB     125$                     ;
                                                            50  D4 007EA 124$:     CLRL    R0                       ;
                                              50                50  5A  C0 007EC 125$:     ADDL2   R10, R0          ; 4053
                                              52        0090    CE  3C 007EF          MOVZWL  INT_STR_DESC, COUNTER    ;
                    52                        50                52  C3 007F4          SUBL3   COUNTER, R0, COUNTER     ;
                                              50        0090    CE  3C 007F8          MOVZWL  INT_STR_DESC, R0         ; 4054
                                     0094     CE                50  C0 007FD          ADDL2   R0, INT_STR_DESC+4       ;
```

```
                        57  D5 00802 126$:  TSTL    COUNTER                         ; 4056
                        35  15 00804        BLEQ    128$
                    50  52  D0 00806        MOVL    COUNTER, R0                     ; 4058
                    03   0  D1 00809        CMPL    R0, #3
                        03  15 0080C        BLEQ    127$
                    50      03  D0 0080E    MOVL    #3, R0
                0090    CE      50  B0 00811 127$:  MOVW    R0, INT_STR_DESC
                        0090    CE  9F 00816        PUSHAB  INT_STR_DESC            ; 4059
                                56  DD 0081A        PUSHL   DIGIT_SEP
                                58  DD 0081C        PUSHL   BAS_OUT_STR
                                58  DD 0081E        PUSHL   BAS_OUT_STR
            00000000G   00          04  FB 00820    CALLS   #4, STR$CONCAT
                    50  0090    CE  3C 00827        MOVZWL  INT_STR_DESC, R0        ; 4060
                0094    CE      50  C0 0082C        ADDL2   R0, INT_STR_DESC+4
                    50  0090    CE  3C 00831        MOVZWL  INT_STR_DESC, R0        ; 4062
                    52          50  C2 00836        SUBL2   R0, COUNTER
                                C7  11 00839        BRB     126$                    ; 4056
                    11      34  AE  E9 0083B 128$:  BLBC    52(SP), 130$            ; 4075
                            0C  A4  D5 0083F        TSTL    12(KERNEL_BLOCK)
                                06  18 00842        BGEQ    129$
                    50      68  AE  9E 00844        MOVAB   CR_DESC, R0
                                0D  11 00848        BRB     131$
                    50      60  AE  9E 0084A 129$:  MOVAB   DR_DESC, R0
                                07  11 0084E        BRB     131$
                    50 00000000' EF  9E 00850 130$:  MOVAB   NULL_DESC, R0
                                50  DD 00857 131$:  PUSHL   R0
                    17      34  AE  E9 00859        BLBC    52(SP), 133$
                            0C  A4  D5 0085D        TSTL    12(KERNEL_BLOCK)        ; 4073
                                09  18 00860        BGEQ    132$
                    50 00000000' EF  9E 00862        MOVAB   MINUS_DESC, R0
                                10  11 00869        BRB     134$
                    50 00000000' EF  9E 0086B 132$:  MOVAB   BLANK_DESC, R0
                                07  11 00872        BRB     134$
                    50 00000000' EF  9E 00874 133$:  MOVAB   NULL_DESC, R0          ; 4072
                                50  DD 0087B 134$:  PUSHL   R0
                        00A0    CE  9F 0087D        PUSHAB  ZERO_STR_DESC           ; 4071
                          64    AE  9F 00881        PUSHAB  FRAC_STR_DESC
                          1C    AE  DD 00884        PUSHL   RADIX_PT
                                58  DD 00887        PUSHL   BAS_OUT_STR
                                58  DD 00889        PUSHL   BAS_OUT_STR
                00000000G   00      07  FB 0088B    CALLS   #7, STR$CONCAT
                    09      30  AE  E8 00892        BLBS    48(SP), 135$            ; 4080
                            0C  A4  D5 00896        TSTL    12(KERNEL_BLOCK)        ; 4086
                                09  18 00899        BGEQ    136$
            05          53      0E  E0 0089B        BBS     #14, FLAGS, 136$
                    50          01  D0 0089F 135$:  MOVL    #1, R0
                                02  11 008A2        BRB     137$
                                50  D4 008A4 136$:  CLRL    R0
                    50      08  AE  C0 008A6 137$:  ADDL2   NO_INT_DIGITS, R0       ; 4093
                    50      04  AE  C0 008AA        ADDL2   NO_FRAC_DIGITS, R0
                    0F      2C  AE  E9 008AE        BLBC    44(SP), 138$            ; 4096
                    55      FF  AA  9E 008B2        MOVAB   -1(R10), R5             ; 4098
                    55          03  C6 008B6        DIVL2   #3, R5
                    51          66  3C 008B9        MOVZWL  (DIGIT_SEP), R1
                    55          51  C4 008BC        MULL2   R1, R5
                                02  11 008BF        BRB     139$
                                55  D4 008C1 138$:  CLRL    R5                      ; 4096
```

```
                                50              55    C0 008C3 139$:  ADDL2   R5, R0                                 ; 4093
                                            026D    31 008C6        BRW     187$                                  ; 4103
                          009A  CE        010E  8F    B0 008C9 140$:  MOVW    #270, ZERO_STR_DESC+2                 ; 4120
                          0098  CE              5A    AE 008D0        MNEGW   R10, ZERO_STR_DESC                    ; 4121
                          009C  CE              5B    D0 008D5        MOVL    ZEROES_PTR, ZERO_STR_DESC+4           ; 4122
                          0092  CE        010E  8F    B0 008DA        MOVW    #270, INT_STR_DESC+2                  ; 4127
                                0E        30    AE    E8 008E1        BLBS    48(SP), 141$                          ; 4129
                                  0C      A4    D5    C0 008E5        TSTL    12(KERNEL_BLOCK)                      ; 4130
                                          09    18    008E8           BGEQ    141$                                  ;
                    05                    0E    E0    008EA           BBS     #14, FLAGS, 141$                      ;
                                50        01    D0    008EE           MOVL    #1, R0                                ;
                                          02    11    008F1           BRB     142$                                  ;
                                          50    D4    008F3 141$:     CLRL    R0                                    ;
                    50          08  AE    50    C3    008F5 142$:     SUBL3   R0, NO_INT_DIGITS, R0                 ; 4129
                                01        50    D1    008FA           CMPL    R0, #1                                ; 4128
                                          03    15    008FD           BLEQ    143$                                  ;
                                50        01    D0    008FF           MOVL    #1, R0                                ;
                          0090  CE              50    B0 00902 143$:  MOVW    R0, INT_STR_DESC                      ;
                          0094  CE              5B    D0 00907        MOVL    ZEROES_PTR, INT_STR_DESC+4            ; 4132
                          00AA  CE        010E  8F    B0 0090C        MOVW    #270, STAR_STR_DESC+2                 ; 4134
                                50        0090  CF    3C 00913        MOVZWL  INT_STR_DESC, R0                      ; 4135
                    50          08  AE    50    C3    00918           SUBL3   R0, NO_INT_DIGITS, R0                 ;
                                0E        30    AE    E8 0091D        BLBS    48(SP), 144$                          ; 4138
                                  0C      A4    D5    00921           TSTL    12(KERNEL_BLOCK)                      ; 4144
                                          09    18    00924           BGEQ    144$                                  ;
                    05                    0E    E0    00926           BBS     #14, FLAGS, 144$                      ;
                                51        01    D0    0092A           MOVL    #1, R1                                ;
                                          02    11    0092D           BRB     145$                                  ;
                                          51    D4    0092F 144$:     CLRL    R1                                    ;
          00A8  CE              50              51    A3 00931 145$:  SUBW3   R1, R0, STAR_STR_DESC                 ; 4136
                          00AC  CE              14    AE    00937     MOVL    STAR_PTR, STAR_STR_DESC+4             ; 4151
                            70  AE        010E0000  8F    D0 0093D        MOVL    #17694720, LEADING_STR_DESC           ; 4155
                            74  AE              5B    D0 00945        MOVL    ZEROES_PTR, LEADING_STR_DESC+4        ; 4154
                          00A2  CE        010E  8F    B0 00949        MOVW    #270, BLANKS_STR_DESC+2               ; 4157
                                04        30    AE    E9 00950        BLBC    48(SP), 146$                          ; 4159
                                          55    D4    00954           CLRL    R5                                    ; 4158
                                          13    11    00956           BRB     149$                                  ;
                                  0C      A4    D5    00958 146$:     TSTL    12(KERNEL_BLOCK)                      ; 4159
                                          09    18    0095B           BGEQ    147$                                  ;
                    05                    0E    E0    0095D           BBS     #14, FLAGS, 147$                      ; 4160
                                51        01    D0    00961           MOVL    #1, R1                                ; 4159
                                          02    11    00964           BRB     148$                                  ;
                                          51    D4    00966 147$:     CLRL    R1                                    ;
                                55              51    D0 00968 148$:  MOVL    R1, R5                                ;
                                55              50    C2 0096B 149$:  SUBL2   R0, R5                                ; 4158
                                05        28    AE    E9 0096E        BLBC    40(SP), 150$                          ; 4161
                                50        01    D0    00972           MOVL    #1, R0                                ; 4160
                                          02    11    00975           BRB     151$                                  ;
                                          50    D4    00977 150$:     CLRL    R0                                    ;
                                55              50    C0 00979 151$:  ADDL2   R0, R5                                ;
                                51              55    CE 0097C        MNEGL   R5, LENGTH                            ;
                                06        24    AE    E9 0097F        BLBC    36(SP), 152$                          ; 4170
                            70  AE              51    B0 00983        MOVW    LENGTH, LEADING_STR_DESC              ;
                                          05    11    00987           BRB     153$                                  ;
                          00A0  CE              51    B0 00989 152$:  MOVW    LENGTH, BLANKS_STR_DESC               ; 4172
                          00A4  CE              18    AE    0098E 153$: MOVL    BLANKS_PTR, BLANKS_STR_DESC+4         ; 4173
                            5A  AE        010E  8F    B0 00994        MOVW    #270, FRAC_STR_DESC+2                 ; 4175
```

```
                    58   AE          52  B0 0099A          MOVW    NO_DIGITS, FRAC_STR_DESC                        ; 4176
                5C  AE   10  A4  04  A4  C1 0099E          ADDL3   4(KERNEL_BLOCK), 16(KERNEL_BLOCK), -            ; 4177
                                                                   FRAC_STR_DESC+4
                         03      28  AE  E8 009A5          BLBS    40(SP), 155$                                    ; 4186
                                 0086  31 009A9 154$:       BRW     164$
                         F9      24  AE  E9 009AC 155$:     BLBC    36(SP), 154$                                    ; 4179
                         11      34  AE  E9 009B0          BLBC    52(SP), 157$                                    ; 4189
                                 0C  A4  D5 009B4          TSTL    12(KERNEL_BLOCK)
                                     06  18 009B7          BGEQ    156$
                         50      68  AE  9E 009B9          MOVAB   CR_DESC, R0
                                     0D  11 009BD          BRB     158$
                         50      60  AE  9E 009BF 156$:     MOVAB   DR_DESC, R0
                                     07  11 009C3          BRB     158$
                         50 00000000'  EF  9E 009C5 157$:     MOVAB   NULL_DESC, R0
                                     50  DD 009CC 158$:     PUSHL   R0
                         17      34  AE  E9 009CE          BLBC    52(SP), 160$
                                 0C  A4  D5 009D2          TSTL    12(KERNEL_BLOCK)                                ; 4188
                                     09  18 009D5          BGEQ    159$
                         50 00000000'  EF  9E 009D7          MOVAB   MINUS_DESC, R0                                  ; 4187
                                     10  11 009DE          BRB     161$
                         50 00000000'  EF  9E 009E0 159$:     MOVAB   BLANK_DESC, R0
                                     07  11 009E7          BRB     161$
                         50 00000000'  EF  9E 009E9 160$:     MOVAB   NULL_DESC, R0
                                     50  DD 009F0 161$:     PUSHL   R0
                                 60      AE  9F 009F2          PUSHAB  FRAC_STR_DESC                               ; 4182
                              00A4       CE  9F 009F5          PUSHAB  ZERO_STR_DESC
                                 1C      AE  DD 009F9          PUSHL   RADIX_PT                                    ; 4186
                              00A4       CE  9F 009FC          PUSHAB  INT_STR_DESC                                ; 4182
                              0088       CE  9F 00A00          PUSHAB  LEADING_STR_DESC
                         12      4C  AE  E8 00A04          BLBS    76(SP), 162$                                    ; 4183
                                 0C  A4  D5 00A08          TSTL    12(KERNEL_BLOCK)                                ; 4184
                                     0D  18 00A0B          BGEQ    162$
                09               53      0E  E0 00A0D          BBS     #14, FLAGS, 162$
                         50 00000000'  EF  9E 00A11          MOVAB   MINUS_DESC, R0
                                     07  11 00A18          BRB     163$
                         50 00000000'  EF  9E 00A1A 162$:     MOVAB   NULL_DESC, R0
                                     50  DD 00A21 163$:     PUSHL   R0
                                 30      AE  DD 00A23          PUSHL   CURRENCY                                    ; 4182
                                     58  DD 00A26          PUSHL   BAS_OUT_STR
                    00000000G    00          0A  FB 00A28          CALLS   #10, STR$CONCAT
                                     00E8  31 00A2F          BRW     183$                                        ; 4181
                         11      34  AE  E9 00A32 164$:     BLBC    52(SP), 166$                                ; 4233
                                 0C  A4  D5 00A36          TSTL    12(KERNEL_BLOCK)
                                     06  18 00A39          BGEQ    165$
                         50      68  AE  9E 00A3B          MOVAB   CR_DESC, R0
                                     0D  11 00A3F          BRB     167$
                         50      60  AE  9F 00A41 165$:     MOVAB   DR_DESC, R0
                                     07  11 00A45          BRB     167$
                         50 00000000'  EF  9E 00A47 166$:     MOVAB   NULL_DESC, R0
                                     50  DD 00A4E 167$:     PUSHL   R0
                         17      34  AE  E9 00A50          BLBC    52(SP), 169$                                ; 4232
                                 0C  A4  D5 00A54          TSTL    12(KERNEL_BLOCK)                            ; 4222
                                     09  18 00A57          BGEQ    168$
                         50 00000000'  EF  9E 00A59          MOVAB   MINUS_DESC, R0
                                     10  11 00A60          BRB     170$
                         50 00000000'  EF  9E 00A62 168$:     MOVAB   BLANK_DESC, R0
                                     07  11 00A69          BRB     170$                                        ; 4220
```

```
                        50 00000000'   EF  9E 0CA6B 169$:  MOVAB   NULL_DESC, R0                          ; 4218
                                       50  DD 00A72 170$:  PUSHL   R0                                    ;
                     5A AE      010E    8F  B0 00A74        MOVW    #270, TEMP_ZERO_DESC+2               ; 4210
                  51 0C AE              52  C3 00A7A        SUBL3   NO_DIGITS, NO_FRAC_DIGITS, R1        ; 4211
               58 AE     51      00A0    CE  A3 00A7F        SUBW3   ZERO_STR_DESC, R1, TEMP_ZERO_DESC    ;
                     5C AE              5B  D0 00A86        MOVL    ZEROES_PTR, TEMP_ZERO_DESC+4         ; 4212
                           58 AE        9F  00A8A           PUSHAB  TEMP_ZERO_DESC                       ; 4204
                           64 AE        9F  00A8D           PUSHAB  FRAC_STR_DESC                        ; 4193
                         00A8 CE        9F  00A90           PUSHAB  ZERO_STR_DESC                        ;
                           20 AE        DD  00A94           PUSHL   RADIX_PT                             ; 4203
                         00A8 CE        9F  00A97           PUSHAB  INT_STR_DESC                         ; 4193
                     07    40 AE        E9  00A9B           BLBC    64(SP), 171$                         ; 4202
                     50  008C CE        9E  00A9F           MOVAB   LEADING_STR_DESC, R0                 ;
                                       07  11 00AA4        BRB     172$                                 ;
                        50 00000000'   EF  9E 00AA6 171$:  MOVAB   NULL_DESC, R0                        ;
                                       50  DD 00AAD 172$:  PUSHL   R0                                    ;
                     12    50 AE        E8  00AAF           BLBS    80(SP), 173$                         ;
                           0C A4        D5  00AB3           TSTL    12(KERNEL_BLOCK)                     ; 4200
                                       0D  18 00AB6        BGEQ    173$                                 ;
                  09       53          0E  E0 00AB8        BBS     #14, FLAGS, 173$                     ;
                        50 00000000'   EF  9E 00ABC        MOVAB   MINUS_DESC, R0                       ;
                                       07  11 00AC3        BRB     174$                                 ;
                        50 00000000'   EF  9E 00AC5 173$:  MOVAB   NULL_DESC, R0                        ;
                                       50  DD 00ACC 174$:  PUSHL   R0                                    ;
                     05    4C AE        E9  00ACE           BLBC    76(SP), 175$                         ; 4199
                           34 AE        DD  00AD2           PUSHL   CURRENCY                             ; 4198
                                       09  11 00AD5        BRB     176$                                 ;
                        55 00000000'   EF  9E 00AD7 175$:  MOVAB   NULL_DESC, R5                        ;
                                       55  DD 00ADE        PUSHL   R5                                   ;
                                       51  D4 00AE0 176$:  CLRL    R1                                   ; 4195
               04       53          04  E1 00AE2        BBC     #4, FLAGS, 177$                      ;
                                       51  D6 00AE6        INCL    R1                                   ;
                                       04  11 00AE8        BRB     178$                                 ;
                     09    4C AE        E9  00AEA 177$:  BLBC    76(SP), 179$                         ; 4196
                        50 00000000'   EF  9E 00AEE 178$:  MOVAB   NULL_DESC, R0                        ;
                                       05  11 00AF5        BRB     180$                                 ;
                     50  00C8 CE        9E  00AF7 179$:  MOVAB   BLANKS_STR_DESC, R0                   ;
                                       50  DD 00AFC 180$:  PUSHL   R0                                    ;
                     07                51  E9 00AFE        BLBC    R1, 181$                             ; 4195
                     50  00D4 CE        9E  00B01           MOVAB   STAR_STR_DESC, R0                    ; 4194
                                       07  11 00B06        BRB     182$                                 ;
                        50 00000000'   EF  9E 00B08 181$:  MOVAB   NULL_DESC, R0                        ;
                                       50  DD 00B0F 182$:  PUSHL   R0                                    ;
                                       58  DD 00B11        PUSHL   BAS_OUT_STR                          ; 4193
            00000000G   00             0D  FB 00B13        CALLS   #13, STR$CONCAT                      ;
                     09    30 AE        E8  00B1A 183$:  BLBS    48(SP), 184$                         ; 4237
                           0C A4        D5  00B1E           TSTL    12(KERNEL_BLOCK)                     ; 4238
                                       09  18 00B21        BGEQ    185$                                 ;
               05       53          0E  E0 00B23        BBS     #14, FLAGS, 185$                     ;
                        50          01  D0 00B27 184$:  MOVL    #1, R0                               ;
                                       02  11 00B2A        BRB     186$                                 ;
                                       50  D4 00B2C 185$:  CLRL    R0                                   ;
                     50    08 AE        C0  00B2E 186$:  ADDL2   NO_INT_DIGITS, R0                    ; 4239
                     50    04 AE        C0  00B32           ADDL2   NO_FRAC_DIGITS, R0                   ;
                     51    0C BE        3C  00B36 187$:  MOVZWL  @RADIX_PT, R1                        ; 4240
               69    50                51  C1 00B3A        ADDL3   R1, R0, (BAS_OUT_STR_LEN)            ;
                     26    4C AE        D1  00B3E 188$:  CMPL    LARGEST_EXP, #38                     ; 4243
```

```
                                    20  15 00B42          BLEQ    189$                                    ;
                              0088  CE  9F 00B44          PUSHAB  ZEROES_DESC                             ; 4246
                00000000G     00    01  FB 00B48          CALLS   #1, STR$FREE1_DX                        ;
                              0080  CE  9F 00B4F          PUSHAB  BLANKS_DESC                             ; 4247
                00000000G     00    01  FB 00B53          CALLS   #1, STR$FREE1_DX                        ;
                                78  AE  9F 00B5A          PUSHAB  STAR_DESC                               ; 4248
                00000000G     00    01  FB 00B5D          CALLS   #1, STR$FREE1_DX                        ;
                                5E  00B0  CE  9E 00B64 189$:  MOVAB   176(SP), SP                        ; 4251
                                    0CA4  8F  BA 00B69          POPR    #^M<R2,R5,R7,R10,R11>               ;
                                              05 00B6D          RSB                                         ;

; Routine Size:  2926 bytes,    Routine Base:  _BAS$CODE + 1196
```

```
;3613     4252  1 ROUTINE FANCY_E_FORM_11 (
;3614     4253  1           NO_INT_DIGITS,
;3615     4254  1           NO_FRAC_DIGITS,
;3616     4255  1           NO_DIGITS,
;3617     4256  1           FLAGS,
;3618     4257  1           KERNEL_BLOCK,
;3619     4258  1           CURRENCY,
;3620     4259  1           DIGIT_SEP,
;3621     4260  1           RADIX_PT,
;3622     4261  1           BAS_OUT_STR,
;3623     4262  1           BAS_OUT_STR_LEN,
;3624     4263  1           DATA_TYPE
;3625     4264  1       ) : JSB_FORMAT_A11 NOVALUE =
;3626     4265  1
;3627     4266  1 !++
;3628     4267  1 ! FUNCTIONAL DESCRIPTION:
;3629     4268  1 !
;3630     4269  1 !       Do the formatting for Basic Print Using E format for h floating.
;3631     4270  1 !
;3632     4271  1 ! FORMAL PARAMETERS:
;3633     4272  1 !
;3634     4273  1 !       NO_INT_DIGITS.rlu.r     number of integer digits in output string
;3635     4274  1 !       NO_FRAC_DIGITS.rlu.r    number of fraction digits in output string
;3636     4275  1 !       NO_DIGITS.rlu.v         number of significant digits
;3637     4276  1 !       FLAGS.rlu.v             no flags are applicable
;3638     4277  1 !       KERNEL_BLOCK.mz.r       parameter block for kernel routine
;3639     4278  1 !       CURRENCY.rt.dx          currency symbol - currently unused
;3640     4279  1 !       DIGIT_SEP.rt.dx         digit group separator - currently unused
;3641     4280  1 !       RADIX_PT.rt.dx          radix point
;3642     4281  1 !       BAS_OUT_STR_LEN.wlu.v   length of output string (handy for fixed length
;3643     4282  1 !                               output strings
;3644     4283  1 !       BAS_OUT_STR.wt.dx       the output string
;3645     4284  1 !       DATA_TYPE.rlu.v         data type of the element to format
;3646     4285  1 !
;3647     4286  1 ! IMPLICIT INPUTS:
;3648     4287  1 !
;3649     4288  1 !       NONE
;3650     4289  1 !
;3651     4290  1 ! IMPLICIT OUTPUTS:
;3652     4291  1 !
;3653     4292  1 !       NONE
;3654     4293  1 !
;3655     4294  1 ! COMPLETION CODES:
;3656     4295  1 !
;3657     4296  1 !       NONE
;3658     4297  1 !
;3659     4298  1 ! SIDE EFFECTS:
;3660     4299  1 !
;3661     4300  1 !       NONE
;3662     4301  1 !
;3663     4302  1 !--
;3664     4303  1
;3665     4304  2     BEGIN
;3666     4305  2
;3667     4306  2     LOCAL
;3668     4307  2         BLANK_STR_DESC : BLOCK [8, BYTE],       ! leading blanks for zero
;3669     4308  2         INT_STR_DESC : BLOCK [8, BYTE],         ! integer portion of number
```

```
;  3670          4309  2          ZERO_STR_DESC : BLOCK [8, BYTE],        ! trailing zeroes
;  3671          4310  2          FRAC_STR_DESC : BLOCK [8, BYTE],        ! fractional portion of number
;  3672          4311  2          EXP_STR : VECTOR [5, BYTE],             ! string for exponent
;  3673          4312  2          EXP_STR_DESC : BLOCK [8, BYTE],         ! Exponent
;  3674          4313  2          LEADING_STR_DESC : BLOCK [8, BYTE];     ! leading zeroes
;  3675          4314  2
;  3676          4315  2      MAP
;  3677          4316  2          RADIX_PT : REF BLOCK [, BYTE],
;  3678          4317  2          BAS_OUT_STR_LEN : REF VECTOR,
;  3679          4318  2          BAS_OUT_STR : REF VECTOR [2, LONG],
;  3680          4319  2          KERNEL_BLOCK : REF BLOCK [K_KERNEL_BLK_SZ, BYTE];
;  3681          4320  2
;  3682          4321  2      BIND
;  3683          4322  2          ZERO = UPLIT (BYTE ('0'));
;  3684          4323  2
;  3685          4324  2      INITIALIZE_DESC;
;  3686          4325  2
;  3687          4326  2      IF CH$EQL (1, (.KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET]), 1, ZERO)
;  3688          4327  2      THEN
;  3689          4328  2  !+
;  3690          4329  2  ! Set up for zero.  A few parameters need to be tweaked and then the general al-
;  3691          4330  2  ! gorithm should work fine.
;  3692          4331  2  !-
;  3693          4332  3          BEGIN
;  3694          4333  3          BLANK_STR_DESC [DSC$W_LENGTH] = MAX (.NO_INT_DIGITS - 1, 0);
;  3695          4334  3          ZERO_STR_DESC [DSC$W_LENGTH] = 0;
;  3696          4335  3          SELECTONE .DATA_TYPE OF
;  3697          4336  3          SET
;  3698          4337  3              [DSC$K_DTYPE_G]:
;  3699          4338  4                  BEGIN
;  3700          4339  4                  EXP_STR = %ASCII'+000';
;  3701          4340  4                  EXP_STR_DESC [DSC$A_POINTER] = EXP_STR;
;  3702          4341  3                  END;
;  3703          4342  3              [DSC$K_DTYPE_H]:
;  3704          4343  4                  BEGIN
;  3705          4344  4                  EXP_STR = UPLIT BYTE (%ASCII'+0000');
;  3706          4345  4                  EXP_STR_DESC [DSC$A_POINTER] = .EXP_STR;        ! extra level of indirection here
;  3707          4346  3                  END;
;  3708          4347  3              [OTHERWISE]:
;  3709          4348  4                  BEGIN
;  3710          4349  4                  EXP_STR = %ASCII'+00';
;  3711          4350  4                  EXP_STR_DESC [DSC$A_POINTER] = EXP_STR;
;  3712          4351  3                  END;
;  3713          4352  3          TES;
;  3714          4353  3          IF (.FLAGS AND V_LEADING_ZERO) NEQ 0
;  3715          4354  3          THEN LEADING_STR_DESC [DSC$W_LENGTH] = .NO_INT_DIGITS;
;  3716          4355  3          END
;  3717          4356  2      ELSE
;  3718          4357  3          BEGIN
;  3719          4358  3          LEADING_STR_DESC [DSC$W_LENGTH] = 0;
;  3720          4359  3          BLANK_STR_DESC [DSC$W_LENGTH] = 0;
;  3721          4360  3          ZERO_STR_DESC [DSC$W_LENGTH] = MAX (.NO_INT_DIGITS - .NO_DIGITS -
;  3722          4361  4              BEGIN
;  3723          4362  4
;  3724          4363  4              IF (.KERNEL_BLOCK [SIGN] LSS 0) THEN 1 ELSE 0
;  3725          4364  4
;  3726          4365  4              END
```

```
;  3727          4366  3          , 0);
;  3728          4367  3          KERNEL_BLOCK [DEC_EXP] = .KERNEL_BLOCK [DEC_EXP] - .NO_INT_DIGITS +
;  3729          4368  4          BEGIN
;  3730          4369  4
;  3731          4370  4          IF (.KERNEL_BLOCK [SIGN] LSS 0) THEN 1 ELSE 0
;  3732          4371  4
;  3733          4372  3          END;                                    ! Add 1 for minus sign
;  3734          4373  3          CVTLP (KERNEL_BLOCK [DEC_EXP], %REF (6), KERNEL_BLOCK [DEC_EXP]);
;  3735          4374  3          SELECTONE .DATA_TYPE OF
;  3736          4375  3          SET
;  3737          4376  3              [DSC$K_DTYPE_G]:
;  3738          4377  4                  BEGIN
;  3739          4378  4                  CVTPS (%REF (6), KERNEL_BLOCK [DEC_EXP], %REF (3), EXP_STR);
;  3740          4379  4                  EXP_STR_DESC [DSC$A_POINTER] = EXP_STR;
;  3741          4380  3                  END;
;  3742          4381  3              [DSC$K_DTYPE_H]:
;  3743          4382  4                  BEGIN
;  3744          4383  4                  CVTPS (%REF (6), KERNEL_BLOCK [DEC_EXP], %REF (4), EXP_STR);
;  3745          4384  4                  EXP_STR_DESC [DSC$A_POINTER] = EXP_STR;
;  3746          4385  3                  END;
;  3747          4386  3              [OTHERWISE]:
;  3748          4387  4                  BEGIN
;  3749          4388  4                  CVTPS (%REF (6), KERNEL_BLOCK [DEC_EXP], %REF (2), EXP_STR);
;  3750          4389  4                  EXP_STR_DESC [DSC$A_POINTER] = EXP_STR;
;  3751          4390  3                  END;
;  3752          4391  3          TES;
;  3753          4392  2          END;
;  3754          4393  2
;  3755          4394  2      LEADING_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3756          4395  2      LEADING_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3757          4396  2      LEADING_STR_DESC [DSC$A_POINTER] = ZEROES;
;  3758          4397  2      BLANK_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3759          4398  2      BLANK_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3760          4399  2      BLANK_STR_DESC [DSC$A_POINTER] = BLANKS;
;  3761          4400  2      EXP_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3762          4401  2      EXP_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3763          4402  2      SELECTONE .DATA_TYPE OF
;  3764          4403  2          SET
;  3765          4404  2              [DSC$K_DTYPE_G]:
;  3766          4405  2                  EXP_STR_DESC [DSC$W_LENGTH] = 4;
;  3767          4406  2              [DSC$K_DTYPE_H]:
;  3768          4407  2                  EXP_STR_DESC [DSC$W_LENGTH] = 5;
;  3769          4408  2              [OTHERWISE]:
;  3770          4409  2                  EXP_STR_DESC [DSC$W_LENGTH] = 3;
;  3771          4410  2          TES;
;  3772          4411  2      INT_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3773          4412  2      INT_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3774          4413  2      INT_STR_DESC [DSC$W_LENGTH] = MIN (.NO_INT_DIGITS -
;  3775          4414  3          BEGIN
;  3776          4415  3
;  3777          4416  3          IF (.KERNEL_BLOCK [SIGN] LSS 0) THEN 1 ELSE 0
;  3778          4417  3
;  3779          4418  3          END
;  3780          4419  2        .NO_DIGITS);
;  3781          4420  2      INT_STR_DESC [DSC$A_POINTER] = .KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET];
;  3782          4421  2      ZERO_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3783          4422  2      ZERO_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
```

```
;  3784          4423  2     ZERO_STR_DESC [DSC$A_POINTER] = ZEROES;
;  3785          4424  2     STR$CONCAT (BAS_OUT_STR [0], BAS_OUT_STR [0],
;  3786          4425  3         BEGIN
;  3787          4426  3
;  3788          4427  3         IF (.KERNEL_BLOCK [SIGN] LSS 0) THEN MINUS_DESC ELSE NULL_DESC
;  3789          4428  3
;  3790          4429  3         END
;  3791          4430  3     , (IF (.FLAGS AND V_LEADING_ZERO) NEQ 0 THEN LEADING_STR_DESC ELSE
;  3792          4431  2     BLANK_STR_DESC), INT_STR_DESC, ZERO_STR_DESC);
;  3793          4432  2     FRAC_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3794          4433  2     FRAC_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3795          4434  2     FRAC_STR_DESC [DSC$W_LENGTH] = MAX (.NO_DIGITS - .NO_INT_DIGITS +
;  3796          4435  3         BEGIN
;  3797          4436  3
;  3798          4437  3         IF (.KERNEL_BLOCK [SIGN] LSS 0) THEN 1 ELSE 0
;  3799          4438  3
;  3800          4439  3         END
;  3801          4440  2     , 0);
;  3802          4441  2     FRAC_STR_DESC [DSC$A_POINTER] = .KERNEL_BLOCK [STRING_ADDR] +       !
;  3803          4442  2     .KERNEL_BLOCK [OFFSET] + .INT_STR_DESC [DSC$W_LENGTH];
;  3804          4443  2     ZERO_STR_DESC [DSC$W_LENGTH] = MAX (.NO_FRAC_DIGITS - .FRAC_STR_DESC [DSC$W_LENGTH], 0);
;  3805          4444  2     STR$CONCAT (BAS_OUT_STR [0], BAS_OUT_STR [0],
;  3806          4445  3         BEGIN
;  3807          4446  3
;  3808          4447  3         IF ((.FLAGS AND V_PERIOD) NEQ 0) THEN .RADIX_PT ELSE NULL_DESC
;  3809          4448  3
;  3810          4449  3         END
;  3811          4450  2     , FRAC_STR_DESC, ZERO_STR_DESC, E_DESC, EXP_STR_DESC);
;  3812          4451  2     BAS_OUT_STR_LEN [0] = .NO_INT_DIGITS + .NO_FRAC_DIGITS +
;  3813          4452  3     BEGIN
;  3814          4453  3
;  3815          4454  3     IF (.FLAGS AND V_PERIOD) THEN .RADIX_PT [DSC$W_LENGTH] ELSE 0
;  3816          4455  3
;  3817          4456  3     END
;  3818          4457  3     + (SELECTONE .DATA_TYPE OF
;  3819          4458  3         SET
;  3820          4459  3             [DSC$K_DTYPE_G]:
;  3821          4460  3                 5;
;  3822          4461  3             [DSC$K_DTYPE_H]:
;  3823          4462  3                 6;
;  3824          4463  3             [OTHERWISE]:
;  3825          4464  3                 4;
;  3826          4465  2         TES);                                   ! add for exp length
;  3827          4466  1     END;                                        ! Routine FANCY_E_FORM_11


                                                30  01D04 P.ABP:   .ASCII  \0\                                     ;
                                                    01D05           .BLKB   3
                                                2E  01D08 P.ABQ:   .ASCII  \.\                                     ;
                                                    01D09           .BLKB   3
                                                2A  01D0C P.ABR:   .ASCII  \*\                                     ;
                                                2A  01D0D          .ASCII  \*\                                     ;
                                                2A  01D0E          .ASCII  \*\                                     ;
                                                2A  01D0F          .ASCII  \*\                                     ;
                                                2A  01D10          .ASCII  \*\                                     ;
                                                2A  01D11          .ASCII  \*\                                     ;
```

```
2A  01D12            .ASCII  \*\
2A  01D13            .ASCII  \*\
2A  01D14            .ASCII  \*\
2A  01D15            .ASCII  \*\
2A  01D16            .ASCII  \*\
2A  01D17            .ASCII  \*\
2A  01D18            .ASCII  \*\
2A  01D19            .ASCII  \*\
2A  01D1A            .ASCII  \*\
2A  01D1B            .ASCII  \*\
2A  01D1C            .ASCII  \*\
2A  01D1D            .ASCII  \*\
2A  01D1E            .ASCII  \*\
2A  01D1F            .ASCII  \*\
2A  01D20            .ASCII  \*\
2A  01D21            .ASCII  \*\
2A  01D22            .ASCII  \*\
2A  01D23            .ASCII  \*\
2A  01D24            .ASCII  \*\
2A  01D25            .ASCII  \*\
2A  01D26            .ASCII  \*\
2A  01D27            .ASCII  \*\
2A  01D28            .ASCII  \*\
2A  01D29            .ASCII  \*\
2A  01D2A            .ASCII  \*\
2A  01D2B            .ASCII  \*\
2A  01D2C            .ASCII  \*\
2A  01D2D            .ASCII  \*\
2A  01D2E            .ASCII  \*\
2A  01D2F            .ASCII  \*\
2A  01D30            .ASCII  \*\
2A  01D31            .ASCII  \*\
    01D32            .BLKB   2
2D  01D34 P.ABS:     .ASCII  \-\
    01D35            .BLKB   3
20  01D38 P.ABT:     .ASCII  \ \
    01D39            .BLKB   3
45  01D3C P.ABU:     .ASCII  \E\
    01D3D            .BLKB   3
20  01D40 P.ABV:     .ASCII  \ \
20  01D41            .ASCII  \ \
20  01D42            .ASCII  \ \
20  01D43            .ASCII  \ \
20  01D44            .ASCII  \ \
20  01D45            .ASCII  \ \
20  01D46            .ASCII  \ \
20  01D47            .ASCII  \ \
20  01D48            .ASCII  \ \
20  01D49            .ASCII  \ \
20  01D4A            .ASCII  \ \
20  01D4B            .ASCII  \ \
20  01D4C            .ASCII  \ \
20  01D4D            .ASCII  \ \
20  01D4E            .ASCII  \ \
20  01D4F            .ASCII  \ \
20  01D50            .ASCII  \ \
20  01D51            .ASCII  \ \
```

```
20  01D52            .ASCII  \ \
20  01D53            .ASCII  \ \
20  01D54            .ASCII  \ \
20  01D55            .ASCII  \ \
20  01D56            .ASCII  \ \
20  01D57            .ASCII  \ \
20  01D58            .ASCII  \ \
20  01D59            .ASCII  \ \
20  01D5A            .ASCII  \ \
20  01D5B            .ASCII  \ \
20  01D5C            .ASCII  \ \
20  01D5D            .ASCII  \ \
20  01D5E            .ASCII  \ \
20  01D5F            .ASCII  \ \
20  01D60            .ASCII  \ \
20  01D61            .ASCII  \ \
20  01D62            .ASCII  \ \
20  01D63            .ASCII  \ \
20  01D64            .ASCII  \ \
20  01D65            .ASCII  \ \
    01D66            .BLKB   2
30  01D68 P.ABW:     .ASCII  \0\
30  01D69            .ASCII  \0\
30  01D6A            .ASCII  \0\
30  01D6B            .ASCII  \0\
30  01D6C            .ASCII  \0\
30  01D6D            .ASCII  \0\
30  01D6E            .ASCII  \0\
30  01D6F            .ASCII  \0\
30  01D70            .ASCII  \0\
30  01D71            .ASCII  \0\
30  01D72            .ASCII  \0\
30  01D73            .ASCII  \0\
30  01D74            .ASCII  \0\
30  01D75            .ASCII  \0\
30  01D76            .ASCII  \0\
30  01D77            .ASCII  \0\
30  01D78            .ASCII  \0\
30  01D79            .ASCII  \0\
30  01D7A            .ASCII  \0\
30  01D7B            .ASCII  \0\
30  01D7C            .ASCII  \0\
30  01D7D            .ASCII  \0\
30  01D7E            .ASCII  \0\
30  01D7F            .ASCII  \0\
30  01D80            .ASCII  \0\
30  01D81            .ASCII  \0\
30  01D82            .ASCII  \0\
30  01D83            .ASCII  \0\
30  01D84            .ASCII  \0\
30  01D85            .ASCII  \0\
30  01D86            .ASCII  \0\
30  01D87            .ASCII  \0\
30  01D88            .ASCII  \0\
30  01D89            .ASCII  \0\
30  01D8A            .ASCII  \0\
30  01D8B            .ASCII  \0\
```

```
                              30  01D8C          .ASCII  \0\                                  ;
                              30  01D8D          .ASCII  \0\                                  ;
              30  30  30  30  2B  01D8E P.ABX:   .ASCII  \+0000\                              ;

                                        ZERO=              P.ABP
                                        DOT=               P.ABQ
                                        STAR=              P.ABR
                                        MINUS=             P.ABS
                                        BLANK=             P.ABT
                                        E=                 P.ABU
                                        BLANKS=            P.ABV
                                        ZEROES=            P.ABW

                       08BC   8F   BB 00000 FANCY_E_FORM_11:
                                                   PUSHR   #^M<R2,R3,R4,R5,R7,R11>            ; 4252
                 5E      C0   AE   9E 00004        MOVAB   -64(SP), SP                        ;
                              57   DD 00008        PUSHL   R7                                 ;
          04     AE           53   D0 0000A        MOVL    R3, FLAGS                          ;
          08     AE           52   D0 0000E        MOVL    R2, NO_DIGITS                      ;
                              51   DD 00012        PUSHL   R1                                 ;
                 5B           50   D0 00014        MOVL    R0, R11                            ;
                  00000000'   EF   D5 00017        TSTL    E_D                                ; 4322
                              24   12 0001D        BNEQ    1$                                 ;
   00000000'     EF    FF7E   CF   9E 0001F        MOVAB   MINUS, M_D                         ;
   00000000'     EF    FF49   CF   9E 00028        MOVAB   DOT, D_D                           ;
   00000000'     EF    FF70   CF   9E 00031        MOVAB   BLANK, B_D                         ;
   00000000'     EF    FF6B   CF   9E 0003A        MOVAB   E, E_D                             ;
50        10     A4      04   A4   C1 00043 1$:    ADDL3   4(KERNEL_BLOCK), 16(KERNEL_BLOCK), R0 ; 4326
        FF23     CF           60   91 00049        CMPB    (R0), ZERO                         ;
                              47   12 0004E        BNEQ    7$                                 ;
                 50      FF   AB   9E 00050        MOVAB   -1(R11), R0                        ; 4333
                              02   18 00054        BGEQ    2$                                 ;
                              50   D4 00056        CLRL    R0                                 ;
          40     AE           50   B0 00058 2$:    MOVW    R0, BLANK_STR_DESC                 ;
                         30   AE   B4 0005C        CLRW    ZERO_STR_DESC                      ; 4334
                 1B           5A   D1 0005F        CMPL    DATA_TYPE, #27                     ; 4337
                              0A   12 00062        BNEQ    3$                                 ;
          20     AE 3030302B  8F   D0 00064        MOVL    #808464427, EXP_STR                ; 4339
                              19   11 0006C        BRB     5$                                 ; 4340
                 1C           5A   D1 0006E 3$:    CMPL    DATA_TYPE, #28                     ; 4342
                              0C   12 00071        BNEQ    4$                                 ;
          20     AE      85   AF   9E 00073        MOVAB   P.ABX, EXP_STR                     ; 4344
          1C     AE      20   AE   D0 00078        MOVL    EXP_STR, EXP_STR_DESC+4            ; 4345
                              0D   11 0007D        BRB     6$                                 ; 4335
          20     AE 0030302B  8F   D0 0007F 4$:    MOVL    #3158059, EXP_STR                  ; 4349
          1C     AE      20   AE   9E 00087 5$:    MOVAB   EXP_STR, EXP_STR_DESC+4            ; 4350
6B        08     AE           0D   E1 0008C 6$:    BBC     #13, FLAGS, 16$                    ; 4353
          10     AE           5B   B0 00091        MOVW    NO_INT_DIGITS, LEADING_STR_DESC    ; 4354
                              65   11 00095        BRB     16$                                ; 4326
                         10   AE   B4 00097 7$:    CLRW    LEADING_STR_DESC                   ; 4358
                         40   AE   B4 0009A        CLRW    BLANK_STR_DESC                     ; 4359
50               5B      0C   AE   C3 0009D        SUBL3   NO_DIGITS, NO_INT_DIGITS, R0       ; 4360
                              52   D4 000A2        CLRL    R2                                 ; 4363
                         0C   A4   D5 000A4        TSTL    12(KERNEL_BLOCK)                   ;
                              07   18 000A7        BGEQ    8$                                 ;
                              52   D6 000A9        INCL    R2                                 ;
```

```
                                51              01  D0 000AB          MOVL    #1, R1                                  ;
                                                02  11 000AE          BRB     9$                                      ;
                                                51  D4 000B0 8$:      CLRL    R1                                      ;
                                50              51  C2 000B2 9$:      SUBL2   R1, R0                                  ; 4361
                                                02  18 000B5          BGEQ    10$                                     ; 4360
                                                50  D4 000B7          CLRL    R0                                      ;
                            30  AE              50  B0 000B9 10$:     MOVW    R0, ZERO_STR_DESC                       ;
                                55      08      A4  9E 000BD          MOVAB   8(KERNEL_BLOCK), R5                     ; 4367
                        51                      65  C3 000C1          SUBL3   NO_INT_DIGITS, (R5), R1                 ;
                                                05  52 E9 000C5          BLBC    R2, 11$                                 ; 4370
                                                50  01 D0 000C8          MOVL    #1, R0                                  ;
                                                    02 11 000CB          BRB     12$                                     ;
                                                    50 D4 000CD 11$:     CLRL    R0                                      ;
                        65              51          50 C1 000CF 12$:     ADDL3   R0, R1, (R5)                            ; 4368
                        65              06          65 F9 000D3          CVTLP   (R5), #6, (R5)                          ; 4373
                                        1B          5A D1 000D7          CMPL    DATA_TYPE, #27                          ; 4376
                                                    08 12 000DA          BNEQ    13$                                     ;
    20  AE              03                  65      06 08 000DC          CVTPS   #6, (R5), #3, EXP_STR                   ; 4378
                                                    13 11 000E2          BRB     15$                                     ; 4379
                                        1C          5A D1 000E4 13$:     CMPL    DATA_TYPE, #28                          ; 4381
                                                    08 12 000E7          BNEQ    14$                                     ;
    20  AE              04                  65      06 08 000E9          CVTPS   #6, (R5), #4, EXP_STR                   ; 4383
                                                    06 11 000EF          BRB     15$                                     ; 4384
    20  AE              02                  65      06 08 000F1 14$:     CVTPS   #6, (R5), #2, EXP_STR                   ; 4388
                        1C  AE              20  AE  9E 000F7 15$:     MOVAB   EXP_STR, EXP_STR_DESC+4                 ; 4389
                        12  AE          010E    8F  B0 000FC 16$:     MOVW    #270, LEADING_STR_DESC+2                ; 4395
                        14  AE          FECF    CF  9E 00102          MOVAB   ZEROES, LEADING_STR_DESC+4              ; 4396
                        42  AE          010E    8F  B0 00108          MOVW    #270, BLANK_STR_DESC+2                  ; 4398
                        44  AE          FE9B    CF  9E 0010E          MOVAB   BLANKS, BLANK_STR_DESC+4                ; 4399
                        1A  AE          010E    8F  B0 00114          MOVW    #270, EXP_STR_DESC+2                    ; 4401
                                        1B          5A D1 0011A          CMPL    DATA_TYPE, #27                          ; 4404
                                                    06 12 0011D          BNEQ    17$                                     ;
                        18  AE                      04 B0 0011F          MOVW    #4, EXP_STR_DESC                        ; 4405
                                                    0F 11 00123          BRB     19$                                     ;
                                        1C          5A D1 00125 17$:     CMPL    DATA_TYPE, #28                          ; 4406
                                                    06 12 00128          BNEQ    18$                                     ;
                        18  AE                      05 B0 0012A          MOVW    #5, EXP_STR_DESC                        ; 4407
                                                    04 11 0012E          BRB     19$                                     ;
                        18  AE                      03 B0 00130 18$:     MOVW    #3, EXP_STR_DESC                        ; 4409
                        3A  AE          010E    8F  B0 00134 19$:     MOVW    #270, INT_STR_DE    2                   ; 4412
                                                53  D4 0013A          CLRL    R3                                      ; 4416
                                    0C          A4  D5 0013C          TSTL    12(KERNEL_BLOCK)                        ;
                                                07  18 0013F          BGEQ    20$                                     ;
                                                53  D6 00141          INCL    R3                                      ;
                                        50          01 D0 00143          MOVL    #1, R0                                  ;
                                                    02 11 00146          BRB     21$                                     ;
                                                    50 D4 00148 20$:     CLRL    R0                                      ;
                50                      5B          50 C3 0014A 21$:     SUBL3   R0, NO_INT_DIGITS, R0                   ; 4414
                        0C  AE                      50 D1 0014E          CMPL    R0, NO_DIGITS                           ; 4419
                                                    04 15 00152          BLEQ    22$                                     ;
                                        50      0C  AE D0 00154          MOVL    NO_DIGITS, R0                           ;
                        38  AE                      50 B0 00158 22$:     MOVW    R0, INT_STR_DESC                        ; 4413
                54      10  A4              04      A4 C1 0015C          ADDL3   4(KERNEL_BLOCK), 16(KERNEL_BLOCK), R4   ; 4420
                        3C  AE                      54 D0 00162          MOVL    R4, INT_STR_DESC+4                      ;
                        32  AE          010E    8F  B0 00166          MOVW    #270, ZERO_STR_DESC+2                   ; 4422
                        34  AE          FE65    CF  9E 0016C          MOVAB   ZEROES, ZERO_STR_DESC+4                 ; 4423
                                            30  AE  9F 00172          PUSHAB  ZERO_STR_DESC                           ; 4424
```

```
                        3C   AE  9F 00175          PUSHAB  INT_STR_DESC
                06      10   AE     0D  E1 00178  BBC     #13, FLAGS, 23$                 ;4430
                             50  18  AE  9E 0017D  MOVAB   LEADING_STR_DESC, R0
                                      04  11 00181  BRB     24$
                             50  48  AE  9E 00183  23$:    MOVAB   BLANK_STR_DESC, R0
                                      50  DD 00187  24$:    PUSHL   R0
                             09           53  E9 00189  BLBC    R3, 25$
                             50 00000000'  EF  9E 0018C  MOVAB   MINUS_DESC, R0          ;4427
                                      07  11 00193  BRB     26$
                             50 00000000'  EF  9E 00195  25$:    MOVAB   NULL_DESC, R0
                                      50  DD 0019C  26$:    PUSHL   R0
                                      58  DD 0019E  PUSHL   BAS_OUT_STR                 ;4424
                                      58  DD 001A0  PUSHL   BAS_OUT_STR
                00000000G     00         06  FB 001A2  CALLS   #6, STR$CONCAT
                      2A   AE     010E   8F  B0 001A9  MOVW    #270, FRAC_STR_DESC+2     ;4433
                52    0C   AE           5B  C3 001AF  SUBL3   NO_INT_DIGITS, NO_DIGITS, R2 ;4434
                             05           53  E9 001B4  BLBC    R3, 27$                   ;4437
                             50           01  D0 001B7  MOVL    #1, R0
                                      02  11 001BA  BRB     28$
                                      50  D4 001BC  27$:    CLRL    R0
                             52           50  C0 001BE  28$:    ADDL2   R0, R2            ;4435
                                      02  18 001C1  BGEQ    29$                       ;4434
                                      52  D4 001C3  CLRL    R2
                      28   AE           52  B0 001C5  29$:    MOVW    R2, FRAC_STR_DESC
                             50  38   AE  3C 001C9  MOVZWL  INT_STR_DESC, R0           ;4442
                2C   AE           54      50  C1 001CD  ADDL3   R0, R4, FRAC_STR_DESC+4
                             51  28   AE  3C 001D2  MOVZWL  FRAC_STR_DESC, R1          ;4443
                51           6E           51  C3 001D6  SUBL3   R1, NO_FRAC_DIGITS, R1
                                      02  18 001DA  BGEQ    30$
                                      51  D4 001DC  CLRL    R1
                      30   AE           51  B0 001DE  30$:    MOVW    R1, ZERO_STR_DESC
                           18   AE      9F 001E2  PUSHAB  EXP_STR_DESC                ;4444
                00000000'           EF  9F 001E5  PUSHAB  E_DESC
                           38   AE      9F 001EB  PUSHAB  ZERO_STR_DESC
                           34   AE      9F 001EE  PUSHAB  FRAC_STR_DESC
                05    18   AE           06  E1 001F1  BBC     #6, FLAGS, 31$            ;4447
                           14   AE      DD 001F6  PUSHL   RADIX_PT
                                      09  11 001F9  BRB     32$
                             57 00000000'  EF  9E 001FB  31$:    MOVAB   NULL_DESC, R7
                                      57  DD 00202  PUSHL   R7
                                      58  DD 00204  32$:    PUSHL   BAS_OUT_STR       ;4444
                                      58  DD 00206  PUSHL   BAS_OUT_STR
                00000000G     00         07  FB 00208  CALLS   #7, STR$CONCAT
                50           5B           6E  C1 0020F  ADDL3   NO_FRAC_DIGITS, NO_INT_DIGITS, R0 ;4451
                                      57  D4 00213  CLRL    R7                        ;4454
                51           50           57  C1 00215  ADDL3   R7, R0, R1             ;4451
                             1B           5A  D1 00219  CMPL    DATA_TYPE, #27          ;4459
                                      05  12 0021C  BNEQ    33$
                             50           05  D0 0021E  MOVL    #5, R0
                                      0D  11 00221  BRB     35$
                             1C           5A  D1 00223  33$:    CMPL    DATA_TYPE, #28  ;4461
                                      05  12 00226  BNEQ    34$
                             50           06  D0 00228  MOVL    #6, R0
                                      03  11 0022B  BRB     35$
                             50           04  D0 0022D  34$:    MOVL    #4, R0          ;4463
                69           51           50  C1 00230  35$:    ADDL3   R0, R1, (BAS_OUT_STR_LEN) ;4457
                             5E   48   AE  9E 00234  MOVAB   72(SP), SP                ;4466
```

```
                08BC    8F  BA 00238           POPR    #^M<R2,R3,R4,R5,R7,R11>                 ;
                            05 0023C           RSB                                             ;

; Routine Size:  573 bytes,    Routine Base:  _BAS$CODE + 1D93
```

```
;  3829          4467  1 ROUTINE PLAIN_E_FORM_11 (                                    !
;  3830          4468  1         NO_DIGITS,                                           !
;  3831          4469  1         FLAGS,                                               !
;  3832          4470  1         KERNEL_BLOCK,                                        !
;  3833          4471  1         BAS_OUT_STR,                                         !
;  3834          4472  1         BAS_OUT_STR_LEN,                                     !
;  3835          4473  1         DATA_TYPE                                            !
;  3836          4474  1     ) : JSB_FORMAT_A6 =
;  3837          4475  1
;  3838          4476  1 !++
;  3839          4477  1 ! FUNCTIONAL DESCRIPTION:
;  3840          4478  1 !
;  3841          4479  1 !       Do plain old E formatting where the value returned is in the range from
;  3842          4480  1 !       0 to 1.
;  3843          4481  1 !
;  3844          4482  1 ! FORMAL PARAMETERS:
;  3845          4483  1 !
;  3846          4484  1 !       NO_DIGITS.rlu.v         number of significant digits in value
;  3847          4485  1 !       FLAGS.rlu.v             no flags are applicable
;  3848          4486  1 !       KERNEL_BLOCK.mz.r       parameter block for kernel conversion routine
;  3849          4487  1 !       BAS_OUT_STR_LEN.wlu.v   length of output string (handy for fixed length
;  3850          4488  1 !                               output strings
;  3851          4489  1 !       BAS_OUT_STR.wt.dx       the output string
;  3852          4490  1 !       DATA_TYPE.rl.v          data type of the element to format
;  3853          4491  1 !
;  3854          4492  1 ! IMPLICIT INPUTS:
;  3855          4493  1 !
;  3856          4494  1 !       NONE
;  3857          4495  1 !
;  3858          4496  1 ! IMPLICIT OUTPUTS:
;  3859          4497  1 !
;  3860          4498  1 !       NONE
;  3861          4499  1 !
;  3862          4500  1 ! COMPLETION CODES:
;  3863          4501  1 !
;  3864          4502  1 !       Returns STR$_TRU
;  3865          4503  1 !
;  3866          4504  1 ! SIDE EFFECTS:
;  3867          4505  1 !
;  3868          4506  1 !       NONE
;  3869          4507  1 !
;  3870          4508  1 !--
;  3871          4509  1
;  3872          4510  2     BEGIN
;  3873          4511  2
;  3874          4512  2     MAP
;  3875          4513  2         BAS_OUT_STR_LEN : REF VECTOR,
;  3876          4514  2         BAS_OUT_STR : REF VECTOR [2, LONG],
;  3877          4515  2         KERNEL_BLOCK : REF BLOCK [K_KERNEL_BLK_SZ, BYTE];
;  3878          4516  2
;  3879          4517  2     LOCAL
;  3880          4518  2         EXP_STR : VECTOR [5, BYTE],                  ! string for exponent
;  3881          4519  2         EXP_STR_DESC : BLOCK [8, BYTE],              ! Exponent
;  3882          4520  2         FRAC_STR_DESC : BLOCK [8, BYTE],             ! Fraction
;  3883          4521  2         STATUS;                                      ! Status returned
;  3884          4522  2
;  3885          4523  2     EXTERNAL
```

```
;  3886          4524  2          OTS$$A_CUR_LUB;
;  3887          4525  2
;  3888          4526  2      GLOBAL REGISTER
;  3889          4527  2          CCB = K_CCB_REG : REF BLOCK [, BYTE];
;  3890          4528  2
;  3891          4529  2      INITIALIZE_DESC;
;  3892          4530  2
;  3893          4531  2  !+
;  3894          4532  2  ! NOTE: E formatting varies depending on whether we are in ANSI mode or
;  3895          4533  2  !       not.  the flag indicating this should be set by the caller; in
;  3896          4534  2  !       ANSI mode, the significand is between 1 and 10 whereas in normal
;  3897          4535  2  !       mode it is between 0 and 1.  In ANSI mode therefore, the output
;  3898          4536  2  !       from the OTS conversion routines must be adjusted somewhat.
;  3899          4537  2  !-
;  3900          4538  2      CCB = .OTS$$A_CUR_LUB;
;  3901          4539  2
;  3902          4540  2      EXP_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3903          4541  2      EXP_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3904          4542  2      SELECTONE .DATA_TYPE OF
;  3905          4543  2      SET
;  3906          4544  2          [DSC$K_DTYPE_G]:
;  3907          4545  2              EXP_STR_DESC [DSC$W_LENGTH] = 4;
;  3908          4546  2          [DSC$K_DTYPE_H]:
;  3909          4547  2              EXP_STR_DESC [DSC$W_LENGTH] = 5;
;  3910          4548  2          [OTHERWISE]:
;  3911          4549  2              EXP_STR_DESC [DSC$W_LENGTH] = 3;
;  3912          4550  2      TES;
;  3913          4551  2      EXP_STR_DESC [DSC$A_POINTER] = EXP_STR;
;  3914          4552  2      FRAC_STR_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3915          4553  2      FRAC_STR_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3916          4554  2      IF .DATA_TYPE NEQ DSC$K_DTYPE_H
;  3917          4555  2      THEN
;  3918          4556  2          FRAC_STR_DESC [DSC$W_LENGTH] = .NO_DIGITS
;  3919          4557  2      ELSE
;  3920          4558  2          FRAC_STR_DESC [DSC$W_LENGTH] = MIN(.NO_DIGITS, 5);
;  3921          4559  2      FRAC_STR_DESC [DSC$A_POINTER] = .KERNEL_BLOCK [STRING_ADDR] + .KERNEL_BLOCK [OFFSET];
;  3922          4560  2
;  3923          4561  2      !+
;  3924          4562  2      ! kludge for ANSI.  decrement exponent.  significand is adjusted below.
;  3925          4563  2      !-
;  3926          4564  2      IF .CCB NEQ 0
;  3927          4565  2      THEN
;  3928          4566  2          IF .CCB [LUB$V_ANSI] EQL 1
;  3929          4567  2          THEN KERNEL_BLOCK [DEC_EXP] = .KERNEL_BLOCK [DEC_EXP] - 1;
;  3930          4568  2
;  3931          4569  2      CVTLP (KERNEL_BLOCK [DEC_EXP], %REF (6), KERNEL_BLOCK [DEC_EXP]);
;  3932          4570  2      SELECTONE .DATA_TYPE OF
;  3933          4571  2      SET
;  3934          4572  2          [DSC$K_DTYPE_G]:
;  3935          4573  2              CVTPS (%REF (6), KERNEL_BLOCK [DEC_EXP], %REF (3), EXP_STR);
;  3936          4574  2          [DSC$K_DTYPE_H]:
;  3937          4575  2              CVTPS (%REF (6), KERNEL_BLOCK [DEC_EXP], %REF (4), EXP_STR);
;  3938          4576  2          [OTHERWISE]:
;  3939          4577  2              CVTPS (%REF (6), KERNEL_BLOCK [DEC_EXP], %REF (2), EXP_STR);
;  3940          4578  2      TES;
;  3941          4579  2
;  3942          4580  3      IF ( IF .CCB NEQ 0  THEN .CCB [LUB$V_ANSI] EQL 1
```

```
;  3943          4581  3                          ELSE 0 )
;  3944          4582  2      THEN
;  3945          4583  3          BEGIN
;  3946          4584  3          !+
;  3947          4585  3          ! ANSI.  concatenate:  blank !! integer !! decimal pt. !! fraction !! E !! exponent
;  3948          4586  3          !                      sign
;  3949          4587  3          !
;  3950          4588  3          ! this is kludged together by taking the most sigificant digit of the
;  3951          4589  3          ! fraction and making it the integer digit of the significand, and
;  3952          4590  3          ! by decrementing the value of the exponent.
;  3953          4591  3          !-
;  3954          4592  3
;  3955          4593  3          !+
;  3956          4594  3          ! set up a local to be the integer portion of the significand
;  3957          4595  3          !-
;  3958          4596  3          LOCAL   TEMP_DESC:      BLOCK [8,BYTE];
;  3959          4597  3          TEMP_DESC [DSC$A_POINTER] = .FRAC_STR_DESC [DSC$A_POINTER];
;  3960          4598  3          TEMP_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  3961          4599  3          TEMP_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  3962          4600  3          TEMP_DESC [DSC$W_LENGTH] = 1;
;  3963          4601  3
;  3964          4602  3          !+
;  3965          4603  3          ! adjust the fractional portion (significand).
;  3966          4604  3          !-
;  3967          4605  3          FRAC_STR_DESC [DSC$A_POINTER] = .FRAC_STR_DESC [DSC$A_POINTER] + 1;
;  3968          4606  3          FRAC_STR_DESC [DSC$W_LENGTH] = .FRAC_STR_DESC [DSC$W_LENGTH] - 1;
;  3969          4607  3
;  3970          4608  3          !+
;  3971          4609  3          ! adjust the exponent (done above)
;  3972          4610  3          !-
;  3973          4611  3
;  3974          4612  3          !+
;  3975          4613  3          ! assemble the number by calling STR$CONCAT
;  3976          4614  3          !-
;  3977          4615  3          STATUS = STR$CONCAT (BAS_OUT_STR [0],
;  3978          4616  4              BEGIN
;  3979          4617  5              IF (.KERNEL_BLOCK [SIGN] LSS 0)
;  3980          4618  4              THEN
;  3981          4619  4                  MINUS_DESC
;  3982          4620  4              ELSE
;  3983          4621  4
;  3984          4622  4                  IF (.FLAGS AND V_STRIP_SPACES) THEN NULL_DESC ELSE BLANK_DESC
;  3985          4623  4
;  3986          4624  4              END
;  3987          4625  3          ,TEMP_DESC, DOT_DESC, FRAC_STR_DESC, E_DESC, EXP_STR_DESC,
;  3988          4626  4              BEGIN
;  3989          4627  4
;  3990          4628  4              IF (.FLAGS AND V_STRIP_SPACES) THEN NULL_DESC ELSE BLANK_DESC
;  3991          4629  4
;  3992          4630  4              END
;  3993          4631  3          );
;  3994          4632  3
;  3995          4633  3          END
;  3996          4634  2      ELSE
;  3997          4635  3          BEGIN
;  3998          4636  3          !+
;  3999          4637  3          ! not ANSI.  concatenate:  blank !! decimal pt. !! fraction !! E !! exponent
```

```
;                                       sign
;-
        STATUS = STR$CONCAT (BAS_OUT_STR [0],
            BEGIN

            IF (.KERNEL_BLOCK [SIGN] LSS 0)
            THEN
                MINUS_DESC
            ELSE

                IF (.FLAGS AND V_STRIP_SPACES) THEN NULL_DESC ELSE BLANK_DESC

            END
        , DOT_DESC, FRAC_STR_DESC, E_DESC, EXP_STR_DESC,
            BEGIN

            IF (.FLAGS AND V_STRIP_SPACES) THEN NULL_DESC ELSE BLANK_DESC

            END
        );
        END;

    BAS_OUT_STR_LEN [0] =
    BEGIN

    IF (.FLAGS AND V_STRIP_SPACES)
    THEN

        IF (.KERNEL_BLOCK [SIGN] LSS 0) THEN 1 ELSE 0

    ELSE
        2

    END
    + (IF .DATA_TYPE EQL DSC$K_DTYPE_H THEN MIN (.NO_DIGITS,5) ELSE .NO_DIGITS)
    + (SELECTONE .DATA_TYPE OF
        SET
            [DSC$K_DTYPE_G]:
                6;
            [DSC$K_DTYPE_H]:
                7;
            [OTHERWISE]:
                5;
        TES);                                   ! add exponent length
    RETURN .STATUS;
    END;                                        ! Routine PLAIN_E_FORM_11


                                    2E  01FD0 P.ABY:  .ASCII  \.\                 ;
                                        01FD1         .BLKB   3
                                    2A  01FD4 P.ABZ:  .ASCII  \*\                 ;
                                    2A  01FD5         .ASCII  \*\                 ;
                                    2A  01FD6         .ASCII  \*\                 ;
                                    2A  01FD7         .ASCII  \*\                 ;
                                    2A  01FD8         .ASCII  \*\                 ;
                                    2A  01FD9         .ASCII  \*\                 ;
```

```
2A  01FDA            .ASCII  \*\
2A  01FDB            .ASCII  \*\
2A  01FDC            .ASCII  \*\
2A  01FDD            .ASCII  \*\
2A  01FDE            .ASCII  \*\
2A  01FDF            .ASCII  \*\
2A  01FE0            .ASCII  \*\
2A  01FE1            .ASCII  \*\
2A  01FE2            .ASCII  \*\
2A  01FE3            .ASCII  \*\
2A  01FE4            .ASCII  \*\
2A  01FE5            .ASCII  \*\
2A  01FE6            .ASCII  \*\
2A  01FE7            .ASCII  \*\
2A  01FE8            .ASCII  \*\
2A  01FE9            .ASCII  \*\
2A  01FEA            .ASCII  \*\
2A  01FEB            .ASCII  \*\
2A  01FEC            .ASCII  \*\
2A  01FED            .ASCII  \*\
2A  01FEE            .ASCII  \*\
2A  01FEF            .ASCII  \*\
2A  01FF0            .ASCII  \*\
2A  01FF1            .ASCII  \*\
2A  01FF2            .ASCII  \*\
2A  01FF3            .ASCII  \*\
2A  01FF4            .ASCII  \*\
2A  01FF5            .ASCII  \*\
2A  01FF6            .ASCII  \*\
2A  01FF7            .ASCII  \*\
2A  01FF8            .ASCII  \*\
2A  01FF9            .ASCII  \*\
    01FFA            .BLKB   2
2D  01FFC P.ACA:     .ASCII  \-\
    01FFD            .BLKB   3
20  02000 P.ACB:     .ASCII  \ \
    02001            .BLKB   3
45  02004 P.ACC:     .ASCII  \E\
    02005            .BLKB   3
20  02008 P.ACD:     .ASCII  \ \
20  02009            .ASCII  \ \
20  0200A            .ASCII  \ \
20  0200B            .ASCII  \ \
20  0200C            .ASCII  \ \
20  0200D            .ASCII  \ \
20  0200E            .ASCII  \ \
20  0200F            .ASCII  \ \
20  02010            .ASCII  \ \
20  02011            .ASCII  \ \
20  02012            .ASCII  \ \
20  02013            .ASCII  \ \
20  02014            .ASCII  \ \
20  02015            .ASCII  \ \
20  02016            .ASCII  \ \
20  02017            .ASCII  \ \
20  02018            .ASCII  \ \
20  02019            .ASCII  \ \
```

```
20  0201A            .ASCII  \ \
20  0201B            .ASCII  \ \
20  0201C            .ASCII  \ \
20  0201D            .ASCII  \ \
20  0201E            .ASCII  \ \
20  0201F            .ASCII  \ \
20  02020            .ASCII  \ \
20  02021            .ASCII  \ \
20  02022            .ASCII  \ \
20  02023            .ASCII  \ \
20  02024            .ASCII  \ \
20  02025            .ASCII  \ \
20  02026            .ASCII  \ \
20  02027            .ASCII  \ \
20  02028            .ASCII  \ \
20  02029            .ASCII  \ \
20  0202A            .ASCII  \ \
20  0202B            .ASCII  \ \
20  0202C            .ASCII  \ \
20  0202D            .ASCII  \ \
    0202E            .BLKB   2
30  02030 P.ACE:     .ASCII  \0\
30  02031            .ASCII  \0\
30  02032            .ASCII  \0\
30  02033            .ASCII  \0\
30  02034            .ASCII  \0\
30  02035            .ASCII  \0\
30  02036            .ASCII  \0\
30  02037            .ASCII  \0\
30  02038            .ASCII  \0\
30  02039            .ASCII  \0\
30  0203A            .ASCII  \0\
30  0203B            .ASCII  \0\
30  0203C            .ASCII  \0\
30  0203D            .ASCII  \0\
30  0203E            .ASCII  \0\
30  0203F            .ASCII  \0\
30  02040            .ASCII  \0\
30  02041            .ASCII  \0\
30  02042            .ASCII  \0\
30  02043            .ASCII  \0\
30  02044            .ASCII  \0\
30  02045            .ASCII  \0\
30  02046            .ASCII  \0\
30  02047            .ASCII  \0\
30  02048            .ASCII  \0\
30  02049            .ASCII  \0\
30  0204A            .ASCII  \0\
30  0204B            .ASCII  \0\
30  0204C            .ASCII  \0\
30  0204D            .ASCII  \0\
30  0204E            .ASCII  \0\
30  0204F            .ASCII  \0\
30  02050            .ASCII  \0\
30  02051            .ASCII  \0\
30  02052            .ASCII  \0\
30  02053            .ASCII  \0\
```

```
                30  02054          .ASCII  \0\                                                     ;
                30  02055          .ASCII  \0\                                                     ;

                                    DOT=                P.ABY
                                    STAR=               P.ABZ
                                    MINUS=              P.ACA
                                    BLANK=              P.ACB
                                    E=                  P.ACC
                                    BLANKS=             P.ACD
                                    ZEROES=             P.ACE
                                           .EXTRN   OTS$$A_CUR_LUB

                    0FCC    8F  BB 00000 PLAIN_E_FORM_11:
                                          PUSHR   #^M<R2,R3,R6,R7,R8,R9,R10,R11>                  ; 4467
                5E                20  C2 00004           SUBL2   #32, SP                         ;
                59                53  D0 00007           MOVL    R3, R9                          ;
                56                52  D0 0000A           MOVL    R2, R6                          ;
                                  51  DD 0000D           PUSHL   R1                              ;
                5A                50  D0 0000F           MOVL    R0, R10                         ;
                    00000000'     EF  D5 00012           TSTL    E_D                             ; 4527
                                  23  12 00018           BNEQ    1$                              ;
00000000'       EF        89      AF  9E 0001A           MOVAB   MINUS, M_D                      ;
00000000'       EF      FF54      CF  9E 00022           MOVAB   DOT, D_D                        ;
00000000'       EF      FF7B      CF  9E 0002B           MOVAB   BLANK, B_D                      ;
00000000'       EF      FF76      CF  9E 00034           MOVAB   E, E_D                          ;
                5B 00000000G      00  D0 0003D 1$:       MOVL    OTS$$A_CUR_LUB, CCB             ; 4538
          16    AE        010E    8F  B0 00044           MOVW    #270, EXP_STR_DESC+2            ; 4541
                1B                55  D1 0004A           CMPL    DATA_TYPE, #27                  ; 4544
                                  06  12 0004D           BNEQ    2$                              ;
          14    AE                04  B0 0004F           MOVW    #4, EXP_STR_DESC                ; 4545
                                  0F  11 00053           BRB     4$                              ;
                1C                55  D1 00055 2$:       CMPL    DATA_TYPE, #28                  ; 4546
                                  06  12 00058           BNEQ    3$                              ;
                      14    AE    05  B0 0005A           MOVW    #5, EXP_STR_DESC                ; 4547
                                  04  11 0005E           BRB     4$                              ;
                      14    AE    03  B0 00060 3$:       MOVW    #3, EXP_STR_DESC                ; 4549
                      18    AE  1C    AE  9E 00064 4$:       MOVAB   EXP_STR, EXP_STR_DESC+4     ; 4551
                      0E    AE  010E  8F  B0 00069           MOVW    #270, FRAC_STR_DESC+2       ; 4553
                1C                55  D1 0006F           CMPL    DATA_TYPE, #28                  ; 4554
                                  06  13 00072           BEQL    5$                              ;
                      0C    AE    5A  B0 00074           MOVW    NO_DIGITS, FRAC_STR_DESC        ; 4556
                                  0F  11 00078           BRB     7$                              ;
                50                5A  D0 0007A 5$:       MOVL    NO_DIGITS, R0                   ; 4558
                05                50  D1 0007D           CMPL    R0, #5                          ;
                                  03  15 00080           BLEQ    6$                              ;
                50                05  D0 00082           MOVL    #5, R0                          ;
                      0C    AE    50  B0 00085 6$:       MOVW    R0, FRAC_STR_DESC               ;
10    AE        10    A6    04  A6  C1 00089 7$:       ADDL3   4(KERNEL_BLOCK), 16(KERNEL_BLOCK), -  ; 4559
                                                                FRAC_STR_DESC+4                 ;
                                  58  D4 00090           CLRL    R8                              ; 4564
                                  5B  D5 00092           TSTL    CCB                             ;
                                  0A  13 00094           BEQL    8$                              ;
                                  58  D6 00096           INCL    R8                              ;
          03    A1        AB      04  E1 00098           BBC     #4, -95(CCB), 8$                ; 4566
                                08    A6  D7 0009D           DECL    8(KERNEL_BLOCK)             ; 4567
                57              08  A6  9E 000A0 8$:       MOVAB   8(KERNEL_BLOCK), R7         ; 4569
          67    06              67  F9 000A4           CVTLP   (R7), #6, (R7)                  ;
```

```
                                 1B              55  D1 000A8          CMPL    DATA_TYPE, #27                          ; 4572
                                                 08  12 000AB          BNEQ    9$
      1C    AE          03       67              06  08 000AD          CVTPS   #6, (R7), #3, EXP_STR                   ; 4573
                                                 13  11 000B3          BRB     11$
                                 1C              55  D1 000B5 9$:      CMPL    DATA_TYPE, #28                          ; 4574
                                                 08  12 000B8          BNEQ    10$
      1C    AE          04       67              06  08 000BA          CVTPS   #6, (R7), #4, EXP_STR                   ; 4575
                                                 06  11 000C0          BRB     11$
      1C    AE          02       67              06  08 000C2 10$:     CVTPS   #6, (R7), #2, EXP_STR                   ; 4577
                                 51        0C    A6  9E 000C8 11$:     MOVAB   12(KERNEL_BLOCK), R1                    ; 4617
                                 74              58  E9 000CC          BLBC    R8, 17$                                 ; 4615
                    6F    A1     AB              04  E1 000CF          BBC     #4, -95(CCB), 17$                       ; 4580
                          08     AE        10    AE  D0 000D4          MOVL    FRAC_STR_DESC+4, TEMP_DESC+4            ; 4597
                          04     AE  010E0001    8F  D0 000D9          MOVL    #17694721, TEMP_DESC                    ; 4600
                                           10    AE  D6 000E1          INCL    FRAC_STR_DESC+4                         ; 4605
                                           0C    AE  B7 000E4          DECW    FRAC_STR_DESC                           ; 4606
                                 09              6E  E9 000E7          BLBC    FLAG$, 12$                              ; 4628
                                 50 00000000'    EF  9E 000EA          MOVAB   NULL_DESC, R0
                                                 07  11 000F1          BRB     13$
                                 50 00000000'    EF  9E 000F3 12$:     MOVAB   BLANK_DESC, R0
                                                 50  DD 000FA 13$:     PUSHL   R0
                                           18    AE  9F 000FC          PUSHAB  EXP_STR_DESC                            ; 4615
                                      00000000'  EF  9F 000FF          PUSHAB  E_DESC
                                           18    AE  9F 00105          PUSHAB  FRAC_STR_DESC
                                      00000000'  EF  9F 00108          PUSHAB  DOT_DESC
                                           18    AE  9F 0010E          PUSHAB  TEMP_DESC
                                                 52  D4 00111          CLRL    R2                                      ; 4617
                                                 61  D5 00113          TSTL    (R1)
                                                 0B  18 00115          BGEQ    14$
                                                 52  D6 00117          INCL    R2
                                 50 00000000'    EF  9E 00119          MOVAB   MINUS_DESC, R0
                                                 14  11 00120          BRB     16$
                                 09        18    AE  E9 00122 14$:     BLBC    FLAG$, 15$                              ; 4622
                                 50 00000000'    EF  9E 00126          MOVAB   NULL_DESC, R0
                                                 07  11 0012D          BRB     16$
                                 50 00000000'    EF  9E 0012F 15$:     MOVAB   BLANK_DESC, R0
                                                 50  DD 00136 16$:     PUSHL   R0
                                                 59  DD 00138          PUSHL   BAS_OUT_STR                             ; 4615
00000000G                        00              08  FB 0013A          CALLS   #8, STR$CONCAT
                                                 57  11 00141          BRB     23$
                                 09              6E  E9 00143 17$:     BLBC    FLAG$, 18$                              ; 4654
                                 50 00000000'    EF  9E 00146          MOVAB   NULL_DESC, R0
                                                 07  11 0014D          BRB     19$
                                 50 00000000'    EF  9E 0014F 18$:     MOVAB   BLANK_DESC, R0
                                                 50  DD 00156 19$:     PUSHL   R0
                                           18    AE  9F 00158          PUSHAB  EXP_STR_DESC                            ; 4640
                                      00000000'  EF  9F 0015B          PUSHAB  E_DESC
                                           18    AE  9F 00161          PUSHAB  FRAC_STR_DESC
                                      00000000'  EF  9F 00164          PUSHAB  DOT_DESC
                                                 52  D4 0016A          CLRL    R2                                      ; 4643
                                                 61  D5 0016C          TSTL    (R1)
                                                 0B  18 0016E          BGEQ    20$
                                                 52  D6 00170          INCL    R2
                                 50 00000000'    EF  9E 00172          MOVAB   MINUS_DESC, R0
                                                 14  11 00179          BRB     22$
                                 09        14    AE  E9 0017B 20$:     BLBC    FLAG$, 21$                              ; 4648
                                 50 00000000'    EF  9E 0017F          MOVAB   NULL_DESC, R0
```

```
                                07  11 00186          BRB     22$                                     ;
                     50 00000000'   EF  9E 00188 21$:  MOVAB   BLANK_DESC, R0                         ;
                                    50  DD 0018F 22$:  PUSHL   R0                                     ;
                                    59  DD 00191        PUSHL   BAS_OUT_STR                           ; 4640
                00000000G     00    07  FB 00193        CALLS   #7, STR$CONCAT                        ;
                              53    50  D0 0019A 23$:  MOVL    R0, STATUS                            ;
                              0C    6E  E9 0019D        BLBC    FLAGS, 25$                            ; 4663
                              05    52  E9 001A0        BLBC    R2, 24$                               ; 4666
                              51    01  D0 001A3        MOVL    #1, R1                                ;
                                    07  11 001A6        BRB     26$                                   ;
                                    51  D4 001A8 24$:  CLRL    R1                                    ;
                                    03  11 001AA        BRB     26$                                   ;
                              51    02  D0 001AC 25$:  MOVL    #2, R1                                ; 4663
                              1C    55  D1 001AF 26$:  CMPL    DATA_TYPE, #28                        ; 4672
                                    0D  12 001B2        BNEQ    27$                                   ;
                              50    5A  D0 001B4        MOVL    NO_DIGITS, R0                         ;
                              05    50  D1 001B7        CMPL    R0, #5                                ;
                                    08  15 001BA        BLEQ    28$                                   ;
                              50    05  D0 001BC        MOVL    #5, R0                                ;
                                    03  11 001BF        BRB     28$                                   ;
                              50    5A  D0 001C1 27$:  MOVL    NO_DIGITS, R0                         ;
                              51    50  C0 001C4 28$:  ADDL2   R0, R1                                ;
                              1B    55  D1 001C7        CMPL    DATA_TYPE, #27                        ; 4675
                                    05  12 001CA        BNEQ    29$                                   ;
                              50    06  D0 001CC        MOVL    #6, R0                                ;
                                    0D  11 001CF        BRB     31$                                   ;
                              1C    55  D1 001D1 29$:  CMPL    DATA_TYPE, #28                        ; 4677
                                    05  12 001D4        BNEQ    30$                                   ;
                              50    07  D0 001D6        MOVL    #7, R0                                ;
                                    03  11 001D9        BRB     31$                                   ;
                              50    05  D0 001DB 30$:  MOVL    #5, R0                                ; 4679
                        64    51    50  C1 001DE 31$:  ADDL3   R0, R1, (BAS_OUT_STR_LEN)             ; 4673
                              50    53  D0 001E2        MOVL    STATUS, R0                            ; 4682
                              5E    24  C0 001E5        ADDL2   #36, SP                               ; 4683
                                OFCC  8F  BA 001E8        POPR    #^M<R2,R3,R6,R7,R8,R9,R10,R11>        ;
                                          05 001EC        RSB                                           ;

; Routine Size:  493 bytes,    Routine Base:  _BAS$CODE + 2056

; 4046          4684  1 END                                        !End of module BAS$CVT_OUT
; 4047          4685  1
; 4048          4686  0 ELUDOM
```

```
;                                PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _BAS$CODE                     8771  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  _BAS$DATA                       40  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
;                              Library Statistics
;
;                                        --------  Symbols  --------      Pages       Processing
;       File                             Total     Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1      9776        13         0        581         00:01.1
```

```
;                                    COMMAND QUALIFIERS

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

; Size:         7687 code + 1124 data bytes
; Run Time:       02:18.0
; Elapsed Time:   05:15.0
; Lines/CPU Min:   2037
; Lexemes/CPU-Min: 18741
; Memory Used:  757 pages
; Compilation Complete
```

BASCLOSE
LIS

BASCONCAT
LIS

BASCTRLO
LIS

BASCHANGE
LIS

BASCTRLC
LIS

BASCHAIN
LIS

BASCOPYFD
LIS

BASCHR
LIS

BASCMPAPP
LIS

BASCUTOUT
LIS

BASCCPOS
LIS

BASDET
LIS

BASDISPAT
LIS

BASCVTTP
LIS

BASDELETE
LIS

BASDATETI
LIS

BASECHO
LIS

BASCVTRP
LIS