```
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  UU      UU  DDDDDDDD    FFFFFFFFFF  WW      WW  LL
BBBBBBBB       AAAAAA       SSSSSSSS  UU      UU  DDDDDDDD    FFFFFFFFFF  WW      WW  LL
BB      BB    AA    AA    SS          UU      UU  DD      DD  FF          WW      WW  LL
BB      BB    AA    AA    SS          UU      UU  DD      DD  FF          WW      WW  LL
BB      BB    AA    AA    SS          UU      UU  DD      DD  FF          WW      WW  LL
BB      BB    AA    AA    SS          UU      UU  DD      DD  FF          WW      WW  LL
BBBBBBBB      AA    AA      SSSSSS    UU      UU  DD      DD  FFFFFFFF    WW      WW  LL
BBBBBBBB      AA    AA      SSSSSS    UU      UU  DD      DD  FFFFFFFF    WW      WW  LL
BB      BB    AAAAAAAAAA          SS  UU      UU  DD      DD  FF          WW  WW  WW  LL
BB      BB    AAAAAAAAAA          SS  UU      UU  DD      DD  FF          WW  WW  WW  LL
BB      BB    AA    AA            SS  UU      UU  DD      DD  FF          WWWW  WWWW  LL              ....
BB      BB    AA    AA            SS  UU      UU  DD      DD  FF          WWWW  WWWW  LL              ....
BBBBBBBB      AA    AA    SSSSSSSS    UUUUUUUUUU  DDDDDDDD    FF          WW      WW  LLLLLLLLLL      ....
BBBBBBBB      AA    AA    SSSSSSSS    UUUUUUUUUU  DDDDDDDD    FF          WW      WW  LLLLLLLLLL      ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE BAS$$UDF_WL (                          ! BASIC Write List Directed UDF
;     2          0002  0                 IDENT = '1-077'               ! File: BASUDFWL.B32 Edit:MDL1077
;     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 ! FACILITY:  BASIC Support Library - not user callable
;    31          0031  1 !
;    32          0032  1 ! ABSTRACT:
;    33          0033  1 !
;    34          0034  1 ! ENVIRONMENT:  User access mode; reentrant AST level or not.
;    35          0035  1 !
;    36          0036  1 ! AUTHOR:       Donald G. Petersen;     CREATION DATE: 17-Mar-78
;    37          0037  1 !
;    38          0038  1 ! MODIFIED BY:
;    39          0039  1 !
;    40          0040  1 ! 0-14  - Store error in ISB$B_ERR_NO, don't signal it! JMT 14-Jan-78
;    41          0041  1 !       Donald G. Petersen, 17-Mar-78: Version 1-01
;    42          0042  1 ! 1-01  -  original
;    43          0043  1 ! 1-02  - Change to JSB linkage.  DGP 14-Nov-78
;    44          0044  1 ! 1-004 - Update copyright notice and add device names to REQUIRE
;    45          0045  1 !           files.  JBS 29-NOV-78
;    46          0046  1 ! 1-005 - Change LUB$B_PRINT_SIZ to LUB$B_R_MARGIN.  DGP 05-Dec-78
;    47          0047  1 ! 1-006 - Change REQUIRE file names from FOR... to OTS...  JBS 07-DEC-78
;    48          0048  1 ! 1-007 - Fix some failures to return values (new BLISS compiler) and
;    49          0049  1 !           run through PRETTY to improve appearance.  JBS 20-DEC-78
;    50          0050  1 ! 1-008 - Add a longword entry point to WL1.  DGP 27-Dec-78
;    51          0051  1 ! 1-009 - Change one argument in call to BAS$CNV_OUT_G. DGP 03-Jan-78
;    52          0052  1 ! 1-010 - Change references to ISB$A_BUF_BEG, BUF_END, and BUF_PTR to LUB.
;    53          0053  1 !           DGP 05-Jan-79
;    54          0054  1 ! 1-011 - Change some constants to symbolics.  DGP 10-Jan-79
;    55          0055  1 ! 1-012 - Convert to CR format for data files.  DGP 11-Jan-79
;    56          0056  1 ! 1-013 - Fix a bug in Print string.  DGP 16-Jan-79
;    57          0057  1 ! 1-014 - Add a guard bit for AST reentrancy.  DGP 20-Jan-79
```

```
;    58          0058  1 ! 1-015 - Correct a typo in edit 014.  JBS 20-JAN-1979
;    59          0059  1 ! 1-016 - Change DO_WRITE to a Global routine BAS$$DO_WRITE so that it can
;    60          0060  1 !         be called by another module which does I/O cleanup at the end of a
;    61          0061  1 !         program.  DGP 22-Jan-79
;    62          0062  1 ! 1-017 - Use 32-bit addresses for externals.  JBS 27-JAN-1979
;    63          0063  1 ! 1-018 - Change buffer overflow handling in BAS$$UDF_WL1.  DGP 22-Feb-79
;    64          0064  1 ! 1-019 - Make change to handling of strings longer than output buffer.  DGP
;    65          0065  1 !         27-Feb-79
;    66          0066  1 ! 1-020 - Make LUB$B_PRINT_POS a longword.  DGP 19-Mar-79
;    67          0067  1 ! 1-021 - Fix a few bugs in PRINT.  DGP 19-Mar-79
;    68          0068  1 ! 1-022 - Make another attempt to get comma formatting working.  DGP 02-Apr-79
;    69          0069  1 ! 1-023 - Semicolon formatting didn't check for overflow of right margin.
;    70          0070  1 !         DGP 03-Apr-79
;    71          0071  1 ! 1-024 - SCAN_TEXT not working properly for non-printing characters.  DGP
;    72          0072  1 !         10-Apr-79
;    73          0073  1 ! 1-025 - Fix an obscure error in SCAN_TEXT.  DGP 12-Apr-79
;    74          0074  1 ! 1-026 - Change macro CHK_CURSOR_POS to GTR from GTRU.  DGP 16-Apr-79
;    75          0075  1 ! 1-027 - TAB is always at least one space when updating printhead position.
;    76          0076  1 !         DGP 16-Apr-79
;    77          0077  1 ! 1-028 - Don't allow print_pos to go less than 0 for backspace characters.
;    78          0078  1 !         DGP 16-Apr-79
;    79          0079  1 ! 1-029 - Add support for CRLF delimiting a record.  DGP 25-Apr-79
;    80          0080  1 ! 1-030 - Remove Global attribute from BAS$$DO_WRITE.  DGP 10-May-79
;    81          0081  1 ! 1-031 - Change ISB$V_AST_GUARD to LUB$V_AST_GUARD.  JBS 15-MAY-1979
;    82          0082  1 ! 1-032 - Make BAS$$DO_WRITE global again for BASUDFWF.  DGP 15-May-79
;    83          0083  1 ! 1-033 - Make the margin 16 bits.  JBS 30-MAY-1979
;    84          0084  1 ! 1-034 - BAS$$UDF_WL0 now initializes LUB$V_FORM_CHAR to 0.  DGP 12-Jun-79
;    85          0085  1 ! 1-035 - Update LUB$A_BUF_PTR in case where string is too long to fit in
;    86          0086  1 !         buffer.  DGP 20-Jun-79
;    87          0087  1 ! 1-036 - Update LUB$A_BUF_PTR correctly if string is too long.  DGP 26-Jun-79
;    88          0088  1 ! 1-037 - Use the new conversion routines.  DGP 28-Jun-79
;    89          0089  1 ! 1-038 - Check V_NOMARGIN and allow infinite margin if set.  DGP 11-Jul-79
;    90          0090  1 ! 1-039 - Check for buffer overflow as well as right margin.  DGP 15-Jul-79
;    91          0091  1 ! 1-040 - Allow Mat Print to jump to REC level from UDF_WL1.  DGP 08-Aug-79
;    92          0092  1 ! 1-041 - Remove the BUILTIN ACTUALCOUNT.  DGP 04-Sep-79
;    93          0093  1 ! 1-042 - PRINT 10 digits for longword integer.  DGP 07-Sep-79
;    94          0094  1 ! 1-043 - Debug CRLF delimiting a record.  DGP 23-Sep-79
;    95          0095  1 ! 1-044 - Clear LUB$L_PRINT_POS for Mat Print if no format c'aracter.  DGP
;    96          0096  1 !         02-Oct-79
;    97          0097  1 ! 1-045 - Clear ISB$V_MAT_PRINT.  DGP 05-Oct-79
;    98          0098  1 ! 1-046 - Make UDF9 dispatch to the REC level.  DGP  12-Oct-79
;    99          0099  1 ! 1-047 - Try again to get 'CRLF' to delimit a record.  DGP 15-Oct-79
;   100          0100  1 ! 1-048 - Fix bug in printing long strings.  The pointer was not moving through
;   101          0101  1 !         the string.  Also, leave the cursor where it lies after each little
;   102          0102  1 !         piece of the long string.  DGP 01-Nov-79
;   103          0103  1 ! 1-049 - Pass one arg to BAS$$DO_WRITE to indicate whether to put out
;   104          0104  1 !         End-of-record.  DGP 06-Nov-79
;   105          0105  1 ! 1-050 - Last edit screwed the proper resetting of LUB$L_PRINT_POS in
;   106          0106  1 !         BAS$$DO_WRITE; fix same.  DGP 08-Nov-79
;   107          0107  1 ! 1-051 - Update columns 10-15 in the translation table in SCAN_TEXT.  DGP
;   108          0108  1 !         09-Nov-79
;   109          0109  1 ! 1-052 - Don't update LUB$A_BUF_PTR in CHK_CURSOR_POS macro.  DGP  09-Nov-79
;   110          0110  1 ! 1-053 - Pick up the scale factor from the ISB.  DGP 15-Nov-79
;   111          0111  1 ! 1-054 - If 'CRLF' is detected to break a record and they are the last two
;   112          0112  1 !         characters in the record, then don't put out a following null record.
;   113          0113  1 !         DGP 03-Jan-80
;   114          0114  1 ! 1-055 - Set ISB$V_PRINT_INI in WL0 and clear it if WL1 is called.  DGP 04-Jan-80
```

```
;  115          0115  1 ! 1-056 - Unconditionally initialize LUB$V_OUTBUF_DR to 1 in WL1.  DGP 07-Jan-80
;  116          0116  1 ! 1-057 - All Calls to BAS$$DO_WRITE write an entire record.  DGP 14-Jan-80
;  117          0117  1 ! 1-058 - Bug fix to PRINT with semicolon formatting that exceeds the left
;  118          0118  1 !         margin.  DGP 22-Dec-80
;  119          0119  1 ! 1-059 - PRINTing a record to the terminal longer than the buffer size inserts
;  120          0120  1 !         an extra CRLF.  DGP 13-Feb-80
;  121          0121  1 ! 1-060 - The macro CHK_CURSOR_POS did not count LF as one character position.
;  122          0122  1 !         It does now.  DGP 18-Feb-80
;  123          0123  1 ! 1-061 - Correction to edit 1-060.  LF should never count as a character
;  124          0124  1 !         position.  DGP  19-Feb-80
;  125          0125  1 ! 1-062 - The macro CHK_CURSOR_POS gets in infinite loop when it is trying
;  126          0126  1 !         to decide at what point the margin was exceeded.  Correct this.
;  127          0127  1 !         DGP 27-FEB-80
;  128          0128  1 ! 1-063 - Edit 1-062 introduced another problem with the buffer pointer.
;  129          0129  1 !         Fix this.
;  130          0130  1 ! 1-064 - The case of exceeding the right margin for terminal output with the
;  131          0131  1 !         first element of a Print statement which follows a Print statement that
;  132          0132  1 !         ended in a semicolon or comma causes the element to be printed twice.
;  133          0133  1 !         Fixed. DGP 01-Mar-80
;  134          0134  1 ! 1-065 - Do not unconditionally reset LUB$V_FORM_CHAR in WL0.  This is a
;  135          0135  1 !         belated finish to edit 55 to fix a bug.  DGP 07-Mar-80.
;  136          0136  1 ! 1-066 - In SCAN_TEXT, when checking for CRLF and last character encountered
;  137          0137  1 !         in the buffer is a CR, don't look beyond the buffer for the LF.
;  138          0138  1 !         REJ 05-May-80
;  139          0139  1 ! 1-067 - Put in the fix for checking last print zone properly.
;  140          0140  1 !         L_ELEM_SIZE is set to zero in MACRO CHK_PRINT_ZONE, and the value
;  141          0141  1 !         of BAS$K_COND_SUC in BASPAR.MAR is set to 2 in place of 3. FM 19-SEP-80
;  142          0142  1 ! 1-068 - Put in the fix for throwing out the next integer or floating number
;  143          0143  1 !         when the buffer is full.  FM 25-SEP-80
;  144          0144  1 !
;  145          0145  1 ! 1-069 - The fix for 1-065A, did not quite fix the problem, so try again.
;  146          0146  1 !         FM JAN-81.
;  147          0147  1 ! 1-070 - Lines with tabs were not split properly if the line exceeded
;  148          0148  1 !         the margin.  Fix CHK_CURSOR_POS and SCAN_TEXT to count
;  149          0149  1 !         tab print positions properly, and SCAN_TEXT now updates
;  150          0150  1 !         the buffer pointer to keep pace with the print position.
;  151          0151  1 !         PL 30-Jun-81.
;  152          0152  1 ! 1-071 - Add support for G & H.  PL 21-Aug-81
;  153          0153  1 ! 1-072 - Add support for packed decimal.  PL 5-Oct-81
;  154          0154  1 ! 1-073 - More edits for packed decimal.  PL 6-Jan-82
;  155          0155  1 ! 1-074 - BAS$$UDF_WL1 should call BAS$CVT_OUT_P_G to format packed strings,
;  156          0156  1 !         instead of BAS$CVT_P_T.  PLL 28-Jan-82
;  157          0157  1 ! 1-075 - Fix a bug in packed = BAS$CVT_OUT_P_G may deallocate and
;  158          0158  1 !         reallocate the dynamic output string.  PLL 4-Mar-82
;  159          0159  1 ! 1-076 - in BAS$$UDF_WL0, load BUF_PTR from RBUF_ADR.  This fixes a re-entrancy
;  160          0160  1 !         problem.  MDL 10-May-1983
;  161          0161  1 ! 1-077 - don't allow string elements to be split across lines when /ANSI.
;  162          0162  1 !         modify CHK_CURSOR_POS accordingly.  MDL 5-Jul-1983
;  163          0163  1 !--
;  164          0164  1
;  165          0165  1 !<BLF/PAGE>
```

```
;  167          0166  1 !
;  168          0167  1 ! SWITCHES:
;  169          0168  1 !
;  170          0169  1
;  171          0170  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;  172          0171  1
;  173          0172  1 !
;  174          0173  1 ! LINKAGES
;  175          0174  1 !
;  176          0175  1
;  177          0176  1 REQUIRE 'RTLIN:OTSLNK';                         ! define all linkages
;  178          0605  1
;  179          0606  1 !
;  180          0607  1 ! TABLE OF CONTENTS:
;  181          0608  1 !
;  182          0609  1
;  183          0610  1 FORWARD ROUTINE
;  184          0611  1     BAS$$UDF_WL0 : JSB_UDF0 NOVALUE,            ! initialization
;  185          0612  1     BAS$$UDF_WL1 : CALL_CCB NOVALUE,            ! format one user I/O list element
;  186          0613  1     BAS$$UDF_WL9 : JSB_UDF9 NOVALUE,            ! end of user I/O list - finish
;  187          0614  1     SCAN_TEXT : CALL_CCB,                       ! Scan text string for special characters
;  188          0615  1     BAS$$DO_WRITE : JSB_DO_WRITE NOVALUE;       ! Output routine
;  189          0616  1
;  190          0617  1 BUILTIN
;  191          0618  1     CVTPS,                                      ! Escape to MARS CVTPS instructions
;  192          0619  1     CVTLD,                                      ! Escape to MARS CVTLD instruction
;  193          0620  1     MOVTUC;                                     ! Escape to MARS MOVTUC instruction
;  194          0621  1
;  195          0622  1 !
;  196          0623  1 ! INCLUDE FILES:
;  197          0624  1 !
;  198          0625  1
;  199          0626  1 REQUIRE 'RTLML:BASPAR';                         ! Intermodule BASIC parameters and constants
;  200          0648  1
;  201          0649  1 REQUIRE 'RTLML:OTSISB';                         ! I/O statement block (ISB) offsets
;  202          0817  1
;  203          0818  1 REQUIRE 'RTLML:OTSLUB';                         ! Only needed to get LUB length!
;  204          0958  1
;  205          0959  1 REQUIRE 'RTLIN:OTSMAC';                         ! Macros
;  206          1153  1
;  207          1154  1 REQUIRE 'RTLIN:RTLPSECT';                       ! Define DECLARE_PSECTS macro
;  208          1249  1
;  209          1250  1 LIBRARY 'RTLSTARLE';                            ! STARLET library for macros and symbols
;  210          1251  1
;  211          1252  1 !
;  212          1253  1 ! EQUATED SYMBOLS:
;  213          1254  1 !
;  214          1255  1
;  215          1256  1 LITERAL
;  216          1257  1     K_FIELD_SIZE = 14;                          ! print zone size
;  217          1258  1
;  218          1259  1 !
;  219          1260  1 ! MACROS:
;  220          1261  1 !
;  221          1262  1
;  222          1263  1 MACRO
;  223          1264  1
```

```
;  224          1265  1      !+
;  225          1266  1      ! If the format character is a comma, see if a full print zone remains
;  226          1267  1      ! in the record.  If a full print zone is not available, then return a
;  227          1268  1      ! zero; else return a one.
;  228          1269  1      ! Note: this macro is only executed if the conversion or scan routine
;  229          1270  1      ! returned a success status.
;  230          1271  1      !-
;  231          1272  1
;  232        M 1273  1      CHK_PRINT_ZONE =
;  233        M 1274  1              CCB[LUB$A_BUF_PTR] = .CCB[LUB$A_BUF_PTR] + .DSC[DSC$W_LENGTH];
;  234        M 1275  1              IF .FORMAT_CHAR EQL BAS$K_COMMA_FOR
;  235        M 1276  1              THEN
;  236        M 1277  1                  BEGIN
;  237        M 1278  1                  CCB[LUB$A_BUF_PTR] = CH$FILL(' ',
;  238        M 1279  1                      (IF .CCB[LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR] GTRU K_FIELD_SIZE
;  239        M 1280  1                          THEN K_FIELD_SIZE - (.CCB [LUB$L_PRINT_POS] MOD K_FIELD_SIZE)
;  240        M 1281  1                          ELSE .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR]),
;  241        M 1282  1                      .CCB[LUB$A_BUF_PTR]);
;  242        M 1283  1                  CCB[LUB$L_PRINT_POS] = .CCB[LUB$L_PRINT_POS] + K_FIELD_SIZE -
;  243        M 1284  1                          (.CCB[LUB$L_PRINT_POS] MOD K_FIELD_SIZE);
;  244        M 1285  1                  END;
;  245        M 1286  1
;  246        M 1287  1
;  247        M 1288  1              !+
;  248        M 1289  1              ! Check for a format character and that the right margin isn't
;  249        M 1290  1              ! infinite.
;  250        M 1291  1              !-
;  251        M 1292  1
;  252        M 1293  1              IF .FORMAT_CHAR EQL BAS$K_COMMA_FOR
;  253        M 1294  1               OR .FORMAT_CHAR EQL BAS$K_SEMI_FORM
;  254        M 1295  1              THEN
;  255        M 1296  1                  BEGIN
;  256        M 1297  1                  IF ((.CCB[LUB$L_PRINT_POS] LEQU .CCB[LUB$W_R_MARGIN]) OR .CCB [LUB$V_NOMARGIN])
;  257        M 1298  1                  AND (.CCB[LUB$A_BUF_PTR] LEQU .CCB[LUB$A_BUF_END])
;  258        M 1299  1                  THEN
;  259        M 1300  1
;  260        M 1301  1                      !+
;  261        M 1302  1                      ! If this is a comma format, check to see if there is a full
;  262        M 1303  1                      ! print zone left on the line.  If this is a semi-colon, this
;  263        M 1304  1                      ! check is redundant, but the code is applicable so what's a
;  264        M 1305  1                      ! little redundancy?
;  265        M 1306  1                      ! If it does then return success.  If it doesn't, return failure
;  266        M 1307  1                      ! without resetting the pointers so that this item will be
;  267        M 1308  1                      ! written.
;  268        M 1309  1                      !-
;  269        M 1310  1
;  270        M 1311  1                      BEGIN
;  271        M 1312  1                      L_ELEM_SIZE = 0;
;  272        M 1313  1                      IF ((.CCB [LUB$L_PRINT_POS] +
;  273        M 1314  1                          (IF .FORMAT_CHAR EQL BAS$K_COMMA_FOR THEN K_FIELD_SIZE
;  274        M 1315  1                                                               ELSE 0)
;  275        M 1316  1                      LEQU .CCB [LUB$W_R_MARGIN]) OR .CCB [LUB$V_NOMARGIN])
;  276        M 1317  1                      THEN
;  277        M 1318  1                          BAS$K_SUCCESS
;  278        M 1319  1                      ELSE
;  279        M 1320  1                          BAS$K_COND_SUC
;  280        M 1321  1                      END
```

```
;  281          M 1322  1
;  282          M 1323  1                    ELSE
;  283          M 1324  1
;  284          M 1325  1                        !+
;  285          M 1326  1                        ! The comma causes this field to overflow the right margin
;  286          M 1327  1                        ! Check to see if there is anything else in the buffer.  If
;  287          M 1328  1                        ! there is, reset the pointers to before this element;  if
;  288          M 1329  1                        ! nothing else in the buffer, return a success status
;  289          M 1330  1                        ! Or, a semicolon format caused the field to overflow past
;  290          M 1331  1                        ! the buffer.  Save this one for the next record.
;  291          M 1332  1                        ! Check for the cursor position not being 0 prior to this
;  292          M 1333  1                        ! element 'cuz there could have been a series of statements
;  293          M 1334  1                        ! ending in comma or semi and if this is a terminal device,
;  294          M 1335  1                        ! then the data is PUT at each IO_END and the buffer doesn't
;  295          M 1336  1                        ! appear dirty although the print line is.
;  296          M 1337  1                        !-
;  297          M 1338  1
;  298          M 1339  1                        IF .CCB[LUB$V_OUTBUF_DR] OR (.SAV_PRINT_POS NEQ 0)
;  299          M 1340  1                        THEN
;  300          M 1341  1                            BEGIN
;  301          M 1342  1                            DSC[DSC$W_LENGTH] = 0;
;  302          M 1343  1                            CCB[LUB$A_BUF_PTR] = .A_SAV_PTR;
;  303          M 1344  1 !+
;  304          M 1345  1 ! If the buffer was overflowed, then leave the cursor where it was; if the margin
;  305          M 1346  1 ! overflowed, then start a new record.
;  306          M 1347  1 !-
;  307          M 1348  1                            IF (NOT .CCB [LUB$V_NOMARGIN]) AND (.CCB [LUB$L_PRINT_POS] GTRU .CCB [LUB$W_R_MARGIN
;  308          M 1349  1                            THEN
;  309          M 1350  1                                BAS$K_MAR_EXC
;  310          M 1351  1                            ELSE
;  311          M 1352  1                                BAS$K_BUF_EXC
;  312          M 1353  1                            END
;  313          M 1354  1                        ELSE
;  314          M 1355  1                            BEGIN
;  315          M 1356  1                            CCB[LUB$A_BUF_PTR] = .CCB[LUB$A_BUF_BEG] + .DSC[DSC$W_LENGTH];
;  316          M 1357  1                            FORMAT_CHAR = BAS$K_SEMI_FORM;
;  317          M 1358  1                            L_ELEM_SIZE = 0;
;  318          M 1359  1 !+
;  319          M 1360  1 ! If the buffer was overflowed, then leave the cursor where it was; if the margin
;  320          M 1361  1 ! overflowed, then start a new record.
;  321          M 1362  1 !-
;  322          M 1363  1                            IF (NOT .CCB [LUB$V_NOMARGIN]) AND (.CCB [LUB$L_PRINT_POS] GTRU .CCB [LUB$W_R_MARGIN
;  323          M 1364  1                            THEN
;  324          M 1365  1                                BAS$K_MAR_EXC
;  325          M 1366  1                            ELSE
;  326          M 1367  1                                BAS$K_BUF_EXC
;  327          M 1368  1                            END
;  328          M 1369  1                    END
;  329          M 1370  1                ELSE
;  330          M 1371  1                    1
;  331          M 1372  1 %;                                                !end of macro
;  332          M 1373  1
;  333          M 1374  1 MACRO
;  334          M 1375  1     ! check the present cursor position against the desired print size.
;  335          M 1376  1     ! Due to the possibility of having carriage control characters in
;  336          M 1377  1     ! the output the data stream, the cursor position is used to check
;  337          M 1378  1     ! the end of the record rather the number of characters in the record.
```

```
;  338          1379  1      !-
;  339          1380  1
;  340        M 1381  1      CHK_CURSOR_POS =
;  341        M 1382  1                  IF (NOT .CCB [LUB$V_NOMARGIN]) AND (.CCB[LUB$L_PRINT_POS] GTR .CCB[LUB$W_R_MARGIN])
;  342        M 1383  1                  THEN
;  343        M 1384  1                      !+
;  344        M 1385  1                      ! The cursor position has exceeded the print size.
;  345        M 1386  1                      ! Restore the pointers back to where they were before the
;  346        M 1387  1                      ! last MOVTUC.
;  347        M 1388  1                      !-
;  348        M 1389  1
;  349        M 1390  1                      BEGIN
;  350        M 1391  1                      RET_VAL = BAS$K_MAR_EXC;
;  351        M 1392  1
;  352        M 1393  1                      !+
;  353        M 1394  1                      ! Reset a few pointers (use the fact that the data is al-
;  354        M 1395  1                      ! ready in the destination field) and find out the length
;  355        M 1396  1                      ! of the string which fits within the right margin.  This
;  356        M 1397  1                      ! requires looking at each character individually.
;  357        M 1398  1                      !-
;  358        M 1399  1
;  359        M 1400  1                      CCB[LUB$L_PRINT_POS] = .L_SAV_PRINT_POS;
;  360        M 1401  1                      CCB[LUB$A_BUF_PTR] = .A_SAVE_BUF_PTR;
;  361        M 1402  1                      T_DSC[DSC$W_LENGTH] = .L_SAVE_LENGTH;
;  362        M 1403  1
;  363        M 1404  1                      !+
;  364        M 1405  1                      ! if ANSI, then get out - we don't split string elements
;  365        M 1406  1                      ! across lines.
;  366        M 1407  1                      !-
;  367        M 1408  1                      IF .CCB [LUB$V_ANSI] THEN EXITLOOP;
;  368        M 1409  1
;  369        M 1410  1                      WHILE .CCB[LUB$L_PRINT_POS] LSS .CCB[LUB$W_R_MARGIN]
;  370        M 1411  1                      DO
;  371        M 1412  1                          BEGIN
;  372        M 1413  1                          CCB[LUB$L_PRINT_POS] = (SELECTONE .(.CCB[LUB$A_BUF_PTR])<0,8,0> OF
;  373        M 1414  1                              SET
;  374        M 1415  1                              [K_TAB]: (.CCB[LUB$L_PRINT_POS] OR K_TAB_LIT) + 1;
;  375        M 1416  1                              [K_CR]: 0;
;  376        M 1417  1                              [K_BKSP]: MAX(0, .CCB[LUB$L_PRINT_POS] - 1);
;  377        M 1418  1                              [K_BLANK TO K_TILDE]: .CCB[LUB$L_PRINT_POS] + 1;
;  378        M 1419  1                              [OTHERWISE]: .CCB[LUB$L_PRINT_POS] + 0;
;  379        M 1420  1                              TES);
;  380        M 1421  1                          CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_BUF_PTR] + 1;
;  381        M 1422  1                          T_DSC[DSC$W_LENGTH] = .T_DSC[DSC$W_LENGTH] + 1;
;  382        M 1423  1                          END;                ! While loop
;  383        M 1424  1      !+
;  384        M 1425  1      ! Restore the buffer pointer so that it can be incremented properly by the
;  385        M 1426  1      ! code that does the common output stuff.
;  386        M 1427  1      !-
;  387        M 1428  1                      CCB [LUB$A_BUF_PTR] = .A_SAVE_BUF_PTR;
;  388        M 1429  1                      EXITLOOP;
;  389        M 1430  1                      END;
;  390          1431  1      %;
;  391          1432  1
;  392          1433  1      !
;  393          1434  1      ! PSECT DECLARATIONS:
;  394          1435  1      !
```

```
;  395          1436  1 DECLARE_PSECTS (BAS);                        ! declare PSECTs for BAS$ facility
;  396          1437  1 !
;  397          1438  1 ! OWN STORAGE:
;  398          1439  1 !
;  399          1440  1 !       None
;  400          1441  1 !
;  401          1442  1 ! EXTERNAL REFERENCES:
;  402          1443  1 !
;  403          1444  1 EXTERNAL
;  404          1445  1 !+
;  405          1446  1 ! Array of REC9 routines
;  406          1447  1 !-
;  407          1448  1     BAS$$AA_REC_PR9 : VECTOR;
;  408          1449  1
;  409          1450  1 EXTERNAL ROUTINE
;  410          1451  1 !     The following are general library routines available for any
;  411          1452  1 !     one's use (value is true if fits in field):
;  412          1453  1     BAS$CVT_OUT_H_G,                           ! Convert H float to G format
;  413          1454  1     BAS$CVT_OUT_H_E,                           ! Convert H float to E format
;  414          1455  1     BAS$CVT_OUT_G_G,                           ! Convert G float to E or F
;  415          1456  1     BAS$CVT_OUT_D_G,                           ! Basic G (E or F) output conversion
;  416          1457  1     BAS$CVT_OUT_P_G,                           ! Convert packed to G format
;  417          1458  1     BAS$$REC_WSL0 : JSB_REC0 NOVALUE,          ! initialize list directed output
;  418          1459  1     BAS$$REC_WSL1 : JSB_REC_WSL1 NOVALUE,      ! write list directed
;  419          1460  1     BAS$$REC_WSL9 : JSB_REC9 NOVALUE,          ! end list directed output
;  420          1461  1     BAS$$REC_MPR1 : JSB_REC1 NOVALUE,          ! Mat Print REC level
;  421          1462  1     STR$APPEND,                                ! append a string
;  422          1463  1     STR$DUPL_CHAR,                             ! create a string of specified char
;  423          1464  1     STR$FREE1_DX,                              ! deallocate dynamic string
;  424          1465  1     LIB$STOP : NOVALUE;                        ! signal an error
;  425          1466  1
```

```
;  427     1467  1 GLOBAL ROUTINE BAS$$UDF_WL0                    ! Write list directed UDF initialization
;  428     1468  1     : JSB_UDF0 NOVALUE =
;  429     1469  1
;  430     1470  1 !++
;  431     1471  1 ! FUNCTIONAL DESCRIPTION:
;  432     1472  1 !
;  433     1473  1 !       Initialize PRINT User data formatter (UDF)
;  434     1474  1 !
;  435     1475  1 ! FORMAL PARAMETERS:
;  436     1476  1 !
;  437     1477  1 !       NONE
;  438     1478  1 !
;  439     1479  1 ! IMPLICIT INPUTS:
;  440     1480  1 !
;  441     1481  1 !       LUB$V_AST_GUARD         Guard bit for AST reentrancy
;  442     1482  1 !       LUB$A_BUF_PTR           Pointer to next byte in user buffer
;  443     1483  1 !
;  444     1484  1 ! IMPLICIT OUTPUTS:
;  445     1485  1 !
;  446     1486  1 !       LUB$V_AST_GUARD         Guard bit for AST reentrancy
;  447     1487  1 !       LUB$A_BUF_BEG           Pointer to first byte of user buffer
;  448     1488  1 !       LUB$A_BUF_PTR           Adr of next byte of output
;  449     1489  1 !                               data buffer
;  450     1490  1 !       LUB$A_BUF_END           Adr of end of data buffer
;  451     1491  1 !       ISB$V_PRINT_INI         indicates that a PRINT has been initialized
;  452     1492  1 !
;  453     1493  1 ! ROUTINE VALUE:
;  454     1494  1 ! COMPLETION CODES:
;  455     1495  1 !
;  456     1496  1 !       NONE
;  457     1497  1 !
;  458     1498  1 ! SIDE EFFECTS:
;  459     1499  1 !
;  460     1500  1 !       NONE
;  461     1501  1 !
;  462     1502  1 !--
;  463     1503  1
;  464     1504  2     BEGIN
;  465     1505  2
;  466     1506  2     EXTERNAL REGISTER
;  467     1507  2         CCB : REF BLOCK [, BYTE];
;  468     1508  2
;  469     1509  2     !+
;  470     1510  2     ! A guard bit is used to ensure AST reentrancy.  The bit is set to 1
;  471     1511  2     ! at the top of the routine, tested for 1 at the bottom of the routine,
;  472     1512  2     ! and set to 0 upon exitting.  If the test for 1 fails at the bottom of
;  473     1513  2     ! the routine, then an AST has gone off and used this routine possibly
;  474     1514  2     ! changing the buffer pointers. Therefore this routine will loop back and
;  475     1515  2     ! run itself again in its entirety.
;  476     1516  2     !-
;  477     1517  2
;  478     1518  2     DO
;  479     1519  3         BEGIN
;  480     1520  3
;  481     1521  3         !+
;  482     1522  3         ! Set the guard bit
;  483     1523  3         !-
```

```
;  484          1524  3
;  485          1525  3          CCB [LUB$V_AST_GUARD] = 1;
;  486          1526  3
;  487          1527  3          !+
;  488          1528  3          ! Call record level to get buffer pointers.
;  489          1529  3          !-
;  490          1530  3
;  491          1531  3          BAS$$REC_WSL0 ();
;  492          1532  3
;  493          1533  3          !+
;  494          1534  3          ! set the beginning of the buffer if there is no format character pending
;  495          1535  3          !-
;  496          1536  3
;  497          1537  3          IF NOT .CCB [LUB$V_FORM_CHAR]
;  498          1538  3            THEN
;  499          1539  4                BEGIN
;  500          1540  4                  CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_RBUF_ADR];
;  501          1541  4                  CCB [LUB$A_BUF_BEG] = .CCB [LUB$A_BUF_PTR];
;  502          1542  3                END;
;  503          1543  3
;  504          1544  3          !+
;  505          1545  3          ! Set the print initialized bit.  The null print line (10       PRINT)
;  506          1546  3          ! does not have an element transmitter (compiler optimization).  So,
;  507          1547  3          ! the bit is set here and then can be cleared by element transmitters
;  508          1548  3          ! if any.
;  509          1549  3          !-
;  510          1550  3
;  511          1551  3          CCB [ISB$V_PRINT_INI] = 1;
;  512          1552  3
;  513          1553  3          !+
;  514          1554  3          ! Check the guard bit.  If it is now 0, then an AST has gone thru this routine
;  515          1555  3          ! Since the data base may have been altered in an unpredictable manner, it
;  516          1556  3          ! is necessary to redo the entire routine.  Note: in worst case processing,
;  517          1557  3          ! the run-time for this routine is essentially unbounded.
;  518          1558  3          !-
;  519          1559  3
;  520          1560  3              END
;  521          1561  2          UNTIL .CCB [LUB$V_AST_GUARD];                ! End of AST guard loop
;  522          1562  2
;  523          1563  2          CCB [LUB$V_AST_GUARD] = 0;
;  524          1564  1          END;


                                                    .TITLE  BAS$$UDF_WL
                                                    .IDENT  \1-077\

                                                    .EXTRN  BAS$$AA_REC_PR9
                                                    .EXTRN  BAS$CVT_OUT_H_G
                                                    .EXTRN  BAS$CVT_OUT_H_E
                                                    .EXTRN  BAS$CVT_OUT_G_G
                                                    .EXTRN  BAS$CVT_OUT_D_G
                                                    .EXTRN  BAS$CVT_OUT_P_G
                                                    .EXTRN  BAS$$REC_WS[0, BAS$$REC_WSL1
                                                    .EXTRN  BAS$$REC_WSL9, BAS$$REC_MPR1
                                                    .EXTRN  STR$APPEND, STR$DUPL_CHAR
                                                    .EXTRN  STR$FREE1_DX, LIB$STOP
```

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

                               52  DD 00000 BAS$$UDF_WL0::
                                                         PUSHL   R2                                  ; 1467
           52        A0        AB  9E 00002              MOVAB   -96(R11), R2                        ; 1525
           62                  20  88 00006 1$:          BISB2   #32, (R2)                           ;
                 00000000G     00  16 00009              JSB     BAS$$REC_WSL0                       ; 1531
 0A    FE  AB                  02  E0 0000F              BBS     #2, -2(CCB), 2$                     ; 1537
       B0  AB        EC        AB  D0 00014              MOVL    -20(CCB), -80(CCB)                  ; 1540
       BC  AB        B0        AB  D0 00019              MOVL    -80(CCB), -68(CCB)                  ; 1541
       97  AB                  08  88 0001E 2$:          BISB2   #8, -105(CCB)                       ; 1551
           52        A0        AB  9E 00022              MOVAB   -96(R11), R2                        ; 1561
 DC        62                  05  E1 00026              BBC     #5, (R2), 1$                        ;
           62                  20  8A 0002A              BICB2   #32, (R2)                           ; 1563
                               04  BA 0002D              POPR    #^M<R2>                             ; 1564
                               05  0002F                 RSB                                         ;

; Routine Size:  48 bytes,    Routine Base:  _BAS$CODE + 0000

;  525          1565  1
```

```
;  527          1566  1 GLOBAL ROUTINE BAS$$UDF_WL1 (ELEM_TYPE, ELEM_SIZE, ELEM_ADR, FORMAT_CHAR        ! format character
;  528          1567  1     ) : CALL_CCB NOVALUE =
;  529          1568  1
;  530          1569  1 !++
;  531          1570  1 ! FUNCTIONAL DESCRIPTION:
;  532          1571  1 !
;  533          1572  1 !       Write list-directed User Data Formatter.
;  534          1573  1 !       Accept an I/O element, format it, and put it in the record buffer.
;  535          1574  1 !       Calls record level processors to perform the actual I/O if the buffer
;  536          1575  1 !       is full or if non-forcible and end-of-record (no format character).
;  537          1576  1 !
;  538          1577  1 ! FORMAL PARAMETERS:
;  539          1578  1 !
;  540          1579  1 !       ELEM_TYPE.rlu.v         data type of the element
;  541          1580  1 !       ELEM_SIZE.rlu.v         size of the data element
;  542          1581  1 !       ELEM_ADR.rlu.r          adr of the data element to be written
;  543          1582  1 !                               Points to a descriptor for strings
;  544          1583  1 !       FORMAT_CHAR.rlu.v               type of format character which followed the data element
;  545          1584  1 !
;  546          1585  1 ! IMPLICIT INPUTS:
;  547          1586  1 !
;  548          1587  1 !       LUB$V_AST_GUARD         guard bit for AST reentrancy
;  549          1588  1 !       LUB$L_PRINT_POS         current cursor position
;  550          1589  1 !       LUB$V_OUTBUF_DR         indicates valid data in the output buffer.
;  551          1590  1 !       LUB$W_R_MARGIN          size of buffer specified in OPEN statement.
;  552          1591  1 !       LUB$V_FORM_CHAR         flag that a format character (',' or ';') was
;  553          1592  1 !                               seen on the last element.
;  554          1593  1 !       LUB$A_BUF_BEG           pointer to beginning of user buffer
;  555          1594  1 !       LUB$A_BUF_PTR           pointer to current position in the buffer.
;  556          1595  1 !       LUB$A_BUF_END           pointer to last byte of buffer + 1.
;  557          1596  1 !
;  558          1597  1 ! IMPLICIT OUTPUTS:
;  559          1598  1 !
;  560          1599  1 !       ISB$V_PRINT_INI         reset print intialize flag - there is at least one
;  561          1600  1 !                               element transmitter
;  562          1601  1 !       LUB$V_AST_GUARD         guard bit for AST reentrancy
;  563          1602  1 !       LUB$V_OUTBUF_DR         indicates valid data in output buffer
;  564          1603  1 !       LUB$V_FORM_CHAR         flag to indicate a format character was seen
;  565          1604  1 !       LUB$L_PRINT_POS         internal cursor position.
;  566          1605  1 !       LUB$A_BUF_PTR           next byte in the user buffer
;  567          1606  1 !
;  568          1607  1 ! ROUTINE VALUE:
;  569          1608  1 ! COMPLETION CODES:
;  570          1609  1 !
;  571          1610  1 !       NONE
;  572          1611  1 !
;  573          1612  1 ! SIDE EFFECTS:
;  574          1613  1 !
;  575          1614  1 !       If an AST goes off while we are in this routine and calls this routine,
;  576          1615  1 !       then this routine will be repeated.  It will continue to be repeated
;  577          1616  1 !       until there are no more ASTs using this routine.
;  578          1617  1 !
;  579          1618  1 !--
;  580          1619  1
;  581          1620  2     BEGIN
;  582          1621  2
;  583          1622  2     EXTERNAL REGISTER
```

```
;  584          1623  2          CCB : REF BLOCK [, BYTE];
;  585          1624  2
;  586          1625  2      MAP
;  587          1626  2          ELEM_ADR : REF VECTOR;                  . element is call-by-reference
;  588          1627  2
;  589          1628  2      LOCAL
;  590          1629  2          STR_PTR,                                . pointer into string being printed
;  591          1630  2          L_ELEM_SIZE,                            ! temp for ELEM_SIZE for strings
;  592          1631  2                                                  ! in case a string is longer than the buffer
;  593          1632  2          A_SAV_PTR,                              ! temp for saving LUB$A_BUF_PTR
;  594          1633  2          P,                                      ! temporary character string pointer
;  595          1634  2          DIFF,                                   ! number of bytes left in record buffer
;  596          1635  2          DSC : BLOCK [8, BYTE],                  ! static string descriptor for output field
;  597          1636  2          TEMP_DEC_DSC : BLOCK [8,BYTE];          ! place to store converted decimal
;  598          1637  2                                                  ! string
;  599          1638  2      LITERAL
;  600          1639  2          K_FIELD_SIZE = 14,                      ! Maximum size to be used by output conversion routine
;  601          1640  2          K_TEMP_BUF_SIZ = 14;
;  602          1641  2
;  603          1642  2      !+
;  604          1643  2      ! This loop is to ensure AST reentrancy.
;  605          1644  2      !-
;  606          1645  2
;  607          1646  2      DO
;  608          1647  3          BEGIN
;  609          1648  3          CCB [LUB$V_AST_GUARD] = 1;
;  610          1649  3          STR_PTR = 0;
;  611          1650  3  !+
;  612          1651  3  ! Reset the print initialize flag; there is at least one element transmitter.
;  613          1652  3  !-
;  614          1653  3          CCB [ISB$V_PRINT_INI] = 0;
;  615          1654  3          L_ELEM_SIZE = .ELEM_SIZE;
;  616          1655  3
;  617          1656  3          !+
;  618          1657  3          ! convert or move the field directly into the output buffer with
;  619          1658  3          ! the exception of the numerics using G format.
;  620          1659  3          !-
;  621          1660  3
;  622          1661  3          DSC [DSC$B_DTYPE] = 0;
;  623          1662  3          DSC [DSC$B_CLASS] = 0;
;  624          1663  3          DSC [DSC$W_LENGTH] = 0;
;  625          1664  3
;  626          1665  3          !+
;  627          1666  3          ! save the current pointer in the buffer
;  628          1667  3          ! Only needed the first time thru the loop
;  629          1668  3          !-
;  630          1669  3
;  631          1670  3          A_SAV_PTR = .CCB [LUB$A_BUF_PTR];
;  632          1671  3
;  633          1672  3          !+
;  634          1673  3          ! determine the length of each field taking into account the format character.
;  635          1674  3          ! In addition to the length, condition the bit which indicates the
;  636          1675  3          ! presence or absence of a format character.
;  637          1676  3          !-
;  638          1677  3
;  639          1678  3          CASE .FORMAT_CHAR FROM BAS$K_SEMI_FORM TO BAS$K_NO_FORM OF
;  640          1679  3              SET
```

```
;  641          1680  3
;  642          1681  3              [BAS$K_SEMI_FORM] :
;  643          1682  3                  CCB [LUB$V_FORM_CHAR] = 1;
;  644          1683  3
;  645          1684  3              [BAS$K_COMMA_FOR] :
;  646          1685  3                  CCB [LUB$V_FORM_CHAR] = 1;
;  647          1686  3
;  648          1687  3              [BAS$K_NO_FORM] :
;  649          1688  3                  CCB [[UB$V_FORM_CHAR] = 0;
;  650          1689  3              TES;
;  651          1690  3
;  652          1691  3
;  653          1692  3          IF .ELEM_TYPE EQL DSC$K_DTYPE_P THEN
;  654          1693  4          BEGIN
;  655          1694  4
;  656          1695  4          !+
;  657          1696  4          ! Convert packed decimal string to text so that it can be handled
;  658          1697  4          ! in the same way as text.  Can't convert it in it's own space so
;  659          1698  4          ! copy to a temporary area.
;  660          1699  4          !-
;  661          1700  4
;  662          1701  4          LOCAL
;  663          1702  4              STATUS;
;  664          1703  4
;  665          1704  4          TEMP_DEC_DSC [DSC$B_CLASS] = DSC$K_CLASS_D;
;  666          1705  4          TEMP_DEC_DSC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  667          1706  4          TEMP_DEC_DSC [DSC$A_POINTER] = 0;
;  668          1707  4          TEMP_DEC_DSC [DSC$W_LENGTH] = 0;
;  669          1708  4
;  670          1709  4          STATUS = (BAS$CVT_OUT_P_G (.ELEM_ADR, 0, L_ELEM_SIZE, TEMP_DEC_DSC));
;  671          1710  4          IF NOT .STATUS THEN LIB$STOP (.STATUS);
;  672          1711  4
;  673          1712  3          END;                                    ! end of packed conversion
;  674          1713  3
;  675          1714  3          !+
;  676          1715  3          ! This WHILE loop will repeat as often as necessary to put out the entire
;  677          1716  3          ! field.  If a field will not fit into the buffer currently it is
;  678          1717  3          ! important to note whether there is anything in the buffer to begin with.
;  679          1718  3          ! If there is then the buffer pointer is set to the value it had when
;  680          1719  3          ! this routine was entered.  Then the buffer is dumped.  If there is nothing
;  681          1720  3          ! in the buffer, then the assumption is that this is a string that is too
;  682          1721  3          ! long to fit into the buffer and must be written in sections.  The string
;  683          1722  3          ! is left in the buffer and the buffer is dumped.  The loop is then repeated
;  684          1723  3          ! until the whole string has been dumped.  The key to control is the flag
;  685          1724  3          ! LUB$OUTBUF_DR.
;  686          1725  3          !-
;  687          1726  3
;  688          1727  3          WHILE 1 DO
;  689          1728  4              BEGIN
;  690          1729  4
;  691          1730  4              LOCAL
;  692          1731  4                  WRITE_STATUS,               ! status returned from the conversion routines
;  693          1732  4                  SAV_PRINT_POS;              ! Temporary to save the current cursor position
;  694          1733  4
;  695          1734  4              SAV_PRINT_POS = .CCB [LUB$L_PRINT_POS];
;  696          1735  4              DSC [DSC$A_POINTER] = .CCB [LUB$A_BUF_PTR];
;  697          1736  4
```

```
;  698          1737  4          !+
;  699          1738  4          ! Perform the appropriate conversions.
;  700          1739  4          !-
;  701          1740  4
;  702          1741  6          IF NOT (WRITE_STATUS = (CASE .ELEM_TYPE FROM DSC$K_DTYPE_B TO DSC$K_DTYPE_H OF
;  703          1742  6                  SET
;  704          1743  6                  [DSC$K_DTYPE_B,DSC$K_DTYPE_W, DSC$K_DTYPE_L] :
;  705          1744  6
;  706          1745  6                      !+
;  707          1746  6                      ! Type integer - word or longword.  Print out as integer
;  708          1747  6                      ! Always print as F format.
;  709          1748  6                      !-
;  710          1749  6
;  711          1750  7                      BEGIN
;  712          1751  7
;  713          1752  7                      LITERAL
;  714          1753  7                          K_NUM_DIGITS = 10;
;  715          1754  7                      LOCAL
;  716          1755  7                          RET_LENGTH,         ! length returned by conversion routine
;  717          1756  7                          D_VALUE : VECTOR [2];       ! holds double precision floating value
;  718          1757  7
;  719          1758  8                      DSC [DSC$W_LENGTH] = (IF .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR] GTRU
;  720          1759  8                          K_FIELD_SIZE
;  721          1760  8
;  722          1761  8                          !+
;  723          1762  8                          ! Convert the integer value to a double precision value in
;  724          1763  8                          ! preparation for calling BAS$CVT_OUT_D_G
;  725          1764  8                          !-
;  726          1765  8
;  727          1766  7                          THEN K_FIELD_SIZE ELSE .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR]);
;  728          1767  7                      CVTLD (ELEM_ADR [0], D_VALUE [0]);
;  729          1768  7
;  730          1769  7                      IF BAS$CVT_OUT_D_G (D_VALUE, 0, RET_LENGTH, DSC, 0, K_NUM_DIGITS)
;  731          1770  7                      THEN
;  732          1771  8                          BEGIN
;  733          1772  8                          L_ELEM_SIZE = 0;
;  734          1773  8                          DSC [DSC$W_LENGTH] = .RET_LENGTH;
;  735          1774  8                          CCB [LUB$L_PRINT_POS] = .CCB [LUB$L_PRINT_POS] + .DSC [DSC$W_LENGTH];
;  736          1775  8                          CHK_PRINT_ZONE
;  737          1776  8                          END
;  738          1777  7                      ELSE
;  739          1778  7                          BAS$K_BUF_EXC
;  740          1779  7
;  741          1780  6                      END;
;  742          1781  6                  [DSC$K_DTYPE_F, DSC$K_DTYPE_D] :
;  743          1782  7                      BEGIN
;  744          1783  7
;  745          1784  7                      LOCAL
;  746          1785  7                          RET_LENGTH,         ! hold the length returned by conversion routine
;  747          1786  7                          D_VALUE : VECTOR [2];       ! holds double precision floating value
;  748          1787  7
;  749          1788  8                      DSC [DSC$W_LENGTH] = (IF .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR] GTRU
;  750          1789  7                          K_FIELD_SIZE THEN K_FIELD_SIZE ELSE .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR])
;  751          1790  7                      D_VALUE [0] = .ELEM_ADR [0];
;  752          1791  7                      D_VALUE [1] = (IF .ELEM_SIZE EQL %UPVAL THEN 0 ELSE .ELEM_ADR [1]);
;  753          1792  7
;  754          1793  7                      IF BAS$CVT_OUT_D_G (D_VALUE, 0, RET_LENGTH, DSC,
```

```
;  755          1794  7                    (IF .ELEM_SIZE EQL %UPVAL THEN 0 ELSE .CCB [ISB$B_SCALE_FAC]))
;  756          1795  7                THEN
;  757          1796  8                    BEGIN
;  758          1797  8                    L_ELEM_SIZE = 0;
;  759          1798  8                    DSC [DSC$W_LENGTH] = .RET_LENGTH;
;  760          1799  8                    CCB [LUB$L_PRINT_POS] = .CCB [LUB$L_PRINT_POS] + .DSC [DSC$W_LENGTH];
;  761          1800  8                    CHK_PRINT_ZONE
;  762          1801  8                    END
;  763          1802  7                ELSE
;  764          1803  7                    BAS$K_BUF_EXC
;  765          1804  7
;  766          1805  6                END;
;  767          1806  6            [DSC$K_DTYPE_G] :
;  768          1807  7                BEGIN
;  769          1808  7
;  770          1809  7                LOCAL
;  771          1810  7                    RET_LENGTH,                 !length retd by conversion routine
;  772          1811  7                    G_VALUE : VECTOR [2];       !holds G floating value
;  773          1812  7
;  774          1813  8                DSC [DSC$W_LENGTH] = (IF .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR] GTRU
;  775          1814  7                    K_FIELD_SIZE THEN K_FIELD_SIZE ELSE .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR])
;  776          1815  7                G_VALUE [0] = .ELEM_ADR [0];
;  777          1816  7                G_VALUE [1] = .ELEM_ADR [1];
;  778          1817  7
;  779          1818  7                IF BAS$CVT_OUT_G_G (G_VALUE, 0, RET_LENGTH,DSC)
;  780          1819  7                THEN
;  781          1820  8                    BEGIN
;  782          1821  8                    L_ELEM_SIZE = 0;
;  783          1822  8                    DSC [DSC$W_LENGTH] = .RET_LENGTH;
;  784          1823  8                    CCB [LUB$L_PRINT_POS] = .CCB [LUB$L_PRINT_POS] + .DSC [DSC$W_LENGTH];
;  785          1824  8                    CHK_PRINT_ZONE
;  786          1825  8                    END
;  787          1826  7                ELSE
;  788          1827  7                    BAS$K_BUF_EXC
;  789          1828  6                END;
;  790          1829  6            [DSC$K_DTYPE_H] :
;  791          1830  7                BEGIN
;  792          1831  7
;  793          1832  7                LOCAL
;  794          1833  7                    RET_LENGTH,                 !length retd by conversion routine
;  795          1834  7                    H_VALUE : VECTOR [4];       !holds H floating value
;  796          1835  7
;  797          1836  8                DSC [DSC$W_LENGTH] = (IF .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR] GTRU
;  798          1837  7                    K_FIELD_SIZE THEN K_FIELD_SIZE ELSE .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR])
;  799          1838  7                H_VALUE [0] = .ELEM_ADR [0];
;  800          1839  7                H_VALUE [1] = .ELEM_ADR [1];
;  801          1840  7                H_VALUE [2] = .ELEM_ADR [2];
;  802          1841  7                H_VALUE [3] = .ELEM_ADR [3];
;  803          1842  7
;  804          1843  7                IF BAS$CVT_OUT_H_G (H_VALUE, 0, RET_LENGTH, DSC)
;  805          1844  7                THEN
;  806          1845  8                    BEGIN
;  807          1846  8                    L_ELEM_SIZE = 0;
;  808          1847  8                    DSC [DSC$W_LENGTH] = .RET_LENGTH;
;  809          1848  8                    CCB [LUB$L_PRINT_POS] = .CCB [LUB$L_PRINT_POS] + .DSC [DSC$W_LENGTH];
;  810          1849  8                    CHK_PRINT_ZONE
;  811          1850  8                    END
```

```
;  812          1851  7                ELSE
;  813          1852  7                    BAS$K_BUF_EXC
;  814          1853  6                END;
;  815          1854  6            [DSC$K_DTYPE_T, DSC$K_DTYPE_P] :
;  816          1855  6
;  817          1856  6                !+
;  818          1857  6                ! text or packed decimal
;  819          1858  6                !-
;  820          1859  6
;  821          1860  7                BEGIN
;  822          1861  7                LOCAL
;  823          1862  7                    SRC_LOC,                    ! source text location
;  824          1863  7                    SCAN_STATUS;                ! status returned by SCAN_TEXT
;  825          1864  7
;  826          1865  7                !+
;  827          1866  7                ! Packed decimal has been converted to text and stored in
;  828          1867  7                ! a temporary descriptor.  SCAN_TEXT must move from this
;  829          1868  7                ! temporary area to the destination descriptor, instead of
;  830          1869  7                ! moving from ELEM_ADR.
;  831          1870  7                !-
;  832          1871  7
;  833          1872  7                IF .ELEM_TYPE EQL DSC$K_DTYPE_P
;  834          1873  7                THEN
;  835          1874  7                    SRC_LOC = .TEMP_DEC_DSC [DSC$A_POINTER]
;  836          1875  7                ELSE
;  837          1876  7                    SRC_LOC = .ELEM_ADR [1];
;  838          1877  7
;  839          1878  7                IF SCAN_STATUS = SCAN_TEXT ((.SRC_LOC + .STR_PTR), .L_ELEM_SIZE,
;  840          1879  7                        .CCB [LUB$A_BOF_END] - .CCB [LUB$A_BOF_PTR], DSC) THEN
;  841          1880  7
;  842          1881  7                !+
;  843          1882  7                ! The field fits into the buffer, now adjust for a
;  844          1883  7                ! comma format character and see if the field still
;  845          1884  7                ! fits.
;  846          1885  7                ! \ Will this overflow the buffer and stay within the
;  847          1886  7                ! right margin?\
;  848          1887  7                !-
;  849          1888  7
;  850          1889  8                    BEGIN
;  851          1890  8                    CHK_PRINT_ZONE
;  852          1891  8                    END
;  853          1892  7                ELSE
;  854          1893  8                    BEGIN
;  855          1894  8
;  856          1895  8                    !+
;  857          1896  8                    ! If LUB$V_OUTBUF_DR is true then there is other data
;  858          1897  8                    ! in the buffer.
;  859          1898  8                    ! If SCAN_STATUS is not BAS$K_CRLF (CRLF detected in record)
;  860          1899  8                    ! The buffer pointer should be reset
;  861          1900  8                    ! and the other data should be dumped.  If the bit is
;  862          1901  8                    ! false, then this is a big string which won't all
;  863          1902  8                    ! fit in the buffer at once.  Dump what is there and
;  864          1903  8                    ! come back for more.
;  865          1904  8                    ! If SCAN_STATUS is BAS$K_CRLF then a CRLF is in the
;  866          1905  8                    ! string and it will fit with the other data here.
;  867          1906  8                    ! Pretend it is all one big string.
;  868          1907  8                    !-
```

```
;  869          1908  8                    IF .CCB [LUB$V_OUTBUF_DR] AND (.SCAN_STATUS NEQ BAS$K_CRLF)
;  870          1909  9                    THEN
;  871          1910  8                        BEGIN
;  872          1911  9
;  873          1912  9                        !+
;  874          1913  9                        ! Set the length back to zero so that the entire
;  875          1914  9                        ! field will be scanned again
;  876          1915  9                        !-
;  877          1916  9
;  878          1917  9                        DSC [DSC$W_LENGTH] = 0;
;  879          1918  9                        CCB [LUB$A_BUF_PTR] = .A_SAV_PTR;
;  880          1919  9                        END
;  881          1920  9                    ELSE
;  882          1921  8
;  883          1922  8                    !+
;  884          1923  8                    ! Decrement the length of this string which is
;  885          1924  8                    ! longer than one record for the next iteration
;  886          1925  8                    !-
;  887          1926  8
;  888          1927  8                        BEGIN
;  889          1928  9                        STR_PTR = .STR_PTR + .DSC [DSC$W_LENGTH];
;  890          1929  9                        L_ELEM_SIZE = .L_ELEM_SIZE - .DSC [DSC$W_LENGTH];
;  891          1930  9                        CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_BUF_PTR] + .DSC [DSC$W_LENGTH]
;  892          1931  9                            - (IF .SCAN_STATUS EQL BAS$K_CRLF THEN 2 ELSE 0);
;  893          1932  9                        END;
;  894          1933  8
;  895          1934  8                    .SCAN_STATUS
;  896          1935  8                    END
;  897          1936  8                END;
;  898          1937  6
;  899          1938  6
;  900          1939  6                [INRANGE, OUTRANGE] : 0     ! this can not happen
;  901          1940  6                TES))
;  902          1941  5            THEN
;  903          1942  4
;  904          1943  4            !+
;  905          1944  4            ! This element won't fit in its entirety.  PUT the contents of the buffer
;  906          1945  4            ! and then loop back and write the remainder.
;  907          1946  4            !-
;  908          1947  4
;  909          1948  4                BEGIN
;  910          1949  5                BAS$$DO_WRITE (.WRITE_STATUS);
;  911          1950  5
;  912          1951  5                IF (.WRITE_STATUS EQL BAS$K_COND_SUC OR .WRITE_STATUS EQL BAS$K_CRLF)
;  913          1952  6                    AND .L_ELEM_SIZE EQL 0
;  914          1953  5                THEN
;  915          1954  5
;  916          1955  5                    BEGIN
;  917          1956  6
;  918          1957  6 !+
;  919          1958  6 ! This field fit but there was not a full print zone left; the
;  920          1959  6 ! format character was a comma.  We want to write this record
;  921          1960  6 ! and then start the next element on the next line at the
;  922          1961  6 ! left margin.
;  923          1962  6 ! Or, a CRLF was found and they are the last two characters in
;  924          1963  6 ! the record and we don't want to put out a null record after this
;  925          1964  6 ! record.  So, bail out here.
```

```
;  926          1965  6 ! Turn off the format character flag.  This avoids the situation where this is the
;  927          1966  6 ! last element in this I/O list and ends with a semicolon or comma.  Even though
;  928          1967  6 ! 'pre' carriage control has been set to 'LF', IO_BEG for the next I/O list will
;  929          1968  6 ! set it to null if the format character flag is on.
;  930          1969  6 !-
;  931          1970  6                     CCB [LUB$V_FORM_CHAR] = 0;
;  932          1971  6                     EXITLOOP;
;  933          1972  5                     END;
;  934          1973  5                 END
;  935          1974  4             ELSE
;  936          1975  5                 BEGIN
;  937          1976  5 !+
;  938          1977  5 ! everything fit this time.  Set the buffer dirty pointer.
;  939          1978  5 !-
;  940          1979  5                     CCB [LUB$V_OUTBUF_DR] = 1;
;  941          1980  5                     EXITLOOP;
;  942          1981  4                     END;
;  943          1982  4
;  944          1983  3             END;                                    ! WHILE loop
;  945          1984  3
;  946          1985  3 !+
;  947          1986  3 ! Free the dynamic string allocated for packed decimal conversion.
;  948          1987  3 !-
;  949          1988  3
;  950          1989  3         IF .ELEM_TYPE EQL DSC$K_DTYPE_P
;  951          1990  3         THEN
;  952          1991  3             STR$FREE1_DX (TEMP_DEC_DSC);
;  953          1992  3
;  954          1993  3 !+
;  955          1994  3 ! Check for MAT PRINT.  If the format character flag is not set, then this
;  956          1995  3 ! matrix is being printed one element/line.  Therefore it is necessary to get to
;  957          1996  3 ! the REC level and do a PUT.
;  958          1997  3 !-
;  959          1998  3         IF (.CCB [ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_MPR) AND NOT .CCB[LUB$V_FORM_CHAR]
;  960          1999  3         THEN
;  961          2000  4             BEGIN
;  962          2001  4             BAS$$REC_MPR1();
;  963          2002  4             CCB [LUB$L_PRINT_POS] = 0;
;  964          2003  3             END;
;  965          2004  3         END
;  966          2005  2     UNTIL .CCB [LUB$V_AST_GUARD];               ! End of AST guard loop
;  967          2006  2
;  968          2007  2     CCB [LUB$V_AST_GUARD] = 0;
;  969          2008  2     RETURN;
;  970          2009  1     END;                                        ! End of BAS$$UDF_WL1


                                    07FC 00000          .ENTRY  BAS$$UDF_WL1, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 1566
                                                                R10
                5E        C0    AE  9E 00002          MOVAB   -64(SP), SP                                   ;
          04    AE        A0    AB  9E 00006          MOVAB   -96(R11), 4(SP)                               ; 1648
          04    BE              20  88 0000B 1$:      BISB2   #32, @4(SP)                                   ;
                                6E  D4 0000F          CLRL    STR_PTR                                       ; 1649
          97    AB              08  8A 00011          BICB2   #8, -105(CCB)                                 ; 1653
```

```
                     0C   AE        08   AC  D0 00015          MOVL    ELEM_SIZE, L_ELEM_SIZE                  ; 1654
                                    38   AE  D4 0001A          CLRL    DSC                                     ; 1663
                          59        B0   AB  D0 0001D          MOVL    -80(CCB), A_SAV_PTR                     ; 1670
                02        01        10   AC  CF 00021          CASEL   FORMAT_CHAR, #1, #2                     ; 1678
              000C      0006              0006     00026 2$:    .WORD   3$-2$,-
                                                                         3$-2$,-
                                                                         4$-2$
                     FE   AB              04   88  0002C 3$:    BISB2   #4, -2(CCB)                             ; 1685
                                          04   11  00030        BRB     5$
                     FE   AB              04   8A  00032 4$:    BICB2   #4, -2(CCB)                             ; 1688
                                    08    AE   D4  00036 5$:    CLRL    8(SP)                                   ; 1692
                          15        04   AC   D1  00039        CMPL    ELEM_TYPE, #21
                                          2C   12  0003D        BNEQ    6$
                                    08    AE   D6  0003F        INCL    8(SP)
                     30   AE  020E0000   8F   D0  00042        MOVL    #34471936, TEMP_DEC_DSC                 ; 1707
                                    34    AE   D4  0004A        CLRL    TEMP_DEC_DSC+4                          ; 1706
                                    30    AE   9F  0004D        PUSHAB  TEMP_DEC_DSC                            ; 1709
                                    10    AE   9F  00050        PUSHAB  L_ELEM_SIZE
                                          7E   D4  00053        CLRL    -(SP)
                                    0C    AC   DD  00055        PUSHL   ELEM_ADR
                    00000000G       00         04  FB  00058    CALLS   #4, BAS$CVT_OUT_P_G
                                    09         50  E8  0005F    BLBS    STATUS, 6$                              ; 1710
                                               50  DD  00062    PUSHL   STATUS
                    00000000G       00         01  FB  00064    CALLS   #1, LIB$STOP
                                    57    C8   AB  9E  0006B 6$:  MOVAB   -56(CCB), R7                           ; 1734
                                    58         67  D0  0006F    MOVL    (R7), SAV_PRINT_POS
                                    56    B0   AB  9E  00072    MOVAB   -80(CCB), R6                            ; 1735
                                    3C    AE        66  D0  00076    MOVL    (R6), DSC+4
                          16                  04   AC  CF  0007A    CASEL   ELEM_TYPE, #6, #22                  ; 1741
              002E      0033              0033     0033     0007F 7$:    .WORD   9$-7$,-
              002E      002E              00F0     00F0     00087        .WORD   9$-7$,-
              002E      002E              002E     032D     0008F        .WORD   9$-7$,-
              032D      002E              002E     002E     00097        .WORD   8$-7$,-
              002E      002E              002E     002E     0009F        .WORD   20$-7$,-
                        023F              01A0     002E     000A7        .WORD   20$-7$,-
                                                                         .WORD   8$-7$,-
                                                                         .WORD   8$-7$,-
                                                                         .WORD   54$-7$,-
                                                                         .WORD   8$-7$,-
                                                                         .WORD   8$-7$,-
                                                                         .WORD   8$-7$,-
                                                                         .WORD   8$-7$,-
                                                                         .WORD   8$-7$,-
                                                                         .WORD   54$-7$,-
                                                                         .WORD   8$-7$,-
                                                                         .WORD   8$-7$,-
                                                                         .WORD   8$-7$,-
                                                                         .WORD   8$-7$,-
                                                                         .WORD   8$-7$,-
                                                                         .WORD   30$-7$,-
                                                                         .WORD   38$-7$
                                                 52  D4  000AD 8$:    CLRL    WRITE_STATUS
                                              0435   31  000AF        BRW     78$
                          50        B4   AB        66  C3  000B2 9$:    SUBL3   (R6), -76(CCB), R0               ; 1758
                                    0E             50  D1  000B7        CMPL    R0, #14
                                                   03  1B  000BA        BLEQU   10$
```

```
                    50        0E  D0 000BC        MOVL     #14, R0                            ;
               38   AE        ')  B0 000BF  10$:  MOVW     R0, DSC                            ;
               28   AE    0C  BC  6E 000C3        CVTLD    @ELEM_ADR, D_VALUE                 ; 1767
                              0A  DD 000C8        PUSHL    #10                                ; 1769
                              7E  D4 000CA        CLRL     -(SP)                              ;
                          40  AE  9F 000CC        PUSHAB   DSC                                ;
                          1C  AE  9F 000CF        PUSHAB   RET_LENGTH                         ;
                              7E  D4 000D2        CLRL     -(SP)                              ;
                          3C  AE  9F 000D4        PUSHAB   D_VALUE                            ;
         00000000G  00        06  FB 000D7        CALLS    #6, BAS$CVT_OUT_D_G                ;
                    03        50  E8 000DE        BLBS     R0, 11$                            ;
                            02C2  31 000E1        BRW      53$                                ;
                          0C  AE  D4 000E4  11$:  CLRL     L_ELEM_SIZE                        ; 1772
               38   AE    10  AE  B0 000E7        MOVW     RET_LENGTH, DSC                    ; 1773
                    50    38  AE  3C 000EC        MOVZWL   DSC, R0                            ; 1774
                    67        50  C0 000F0        ADDL2    R0, (R7)                           ;
                    50    38  AE  3C 000F3        MOVZWL   DSC, R0                            ;
                    66        50  C0 000F7        ADDL2    R0, (R6)                           ;
                              5A  D4 000FA        CLRL     R10                                ;
                    02    10  AC  D1 000FC        CMPL     FORMAT_CHAR, #2                    ;
                              36  12 00100        BNEQ     13$                                ;
                              5A  D6 00102        INCL     R10                                ;
          50   B4   AB        66  C3 00104        SUBL3    (R6), -76(CCB), R0                 ;
                    0E        50  D1 00109        CMPL     R0, #14                            ;
                              0E  1B 0010C        BLEQU    12$                                ;
 7E       00        67        01  7A 0010E        EMUL     #1, (R7), #0, -(SP)                ;
 50       50        8E        0E  7B 00113        EDIV     #14, (SP)+, R0, R0                 ;
          50        0E        50  C3 00118        SUBL3    R0, #14, R0                        ;
 50       20        6E        00  2C 0011C  12$:  MOVC5    #0, (SP), #32, R0, @0(R6)          ;
                          00  B6     00121                                                    ;
                    66        53  D0 00123        MOVL     R3, (R6)                           ;
 7E       00        67        01  7A 00126        EMUL     #1, (R7), #0, -(SP)                ;
 50       50        8E        0E  7B 0012B        EDIV     #14, (SP)+, R0, R0                 ;
          50        67        50  C3 00130        SUBL3    R0, (R7), R0                       ;
                    67    0E  A0  9E 00134        MOVAB    14(R0), (R7)                       ;
                    09        5A  E8 00138  13$:  BLBS     R10, 14$                           ;
                    01    10  AC  D1 0013B        CMPL     FORMAT_CHAR, #1                    ;
                              03  13 0013F        BEQL     14$                                ;
                            035D  31 00141        BRW      71$                                ;
 67       D4   AB             10  00  ED 00144  14$:  CMPZV    #0, #16, -44(CCB), (R7)        ;
                              05  1E 0014A        BGEQU    16$                                ;
          09   A1   AB        01  E1 0014C  15$:  BBC      #1, -95(CCB), 17$                  ;
               B4   AB        66  D1 00151  16$:  CMPL     (R6), -76(CCB)                     ;
                              03  1A 00155        BGTRU    17$                                ;
                            020D  31 00157        BRW      45$                                ;
          07   FE   AB        03  E0 0015A  17$:  BBS      #3, -2(CCB), 19$                   ;
                              58  D5 0015F  18$:  TSTL     SAV_PRINT_POS                      ;
                              03  12 00161        BNEQ     19$                                ;
                            021F  31 00163        BRW      50$                                ;
                          38  AE  B4 00166  19$:  CLRW     DSC                                ;
                    66        59  D0 00169        MOVL     A_SAV_PTR, (R6)                    ;
                            0226  31 0016C        BRW      51$                                ;
          50   B4   AB        66  C3 0016F  20$:  SUBL3    (R6), -76(CCB), R0                 ; 1788
                    0E        50  D1 00174        CMPL     R0, #14                            ;
                              03  1B 00177        BLEQU    21$                                ;
                    50        0E  D0 00179        MOVL     #14, R0                            ;
               38   AE        50  B0 0017C  21$:  MOVW     R0, DSC                            ;
```

```
                50      0C      AC  D0 00180          MOVL    ELEM_ADR, R0                    ; 1790
            28  AE              60  D0 00184          MOVL    (R0), D_VALUE
                                51  D4 00188          CLRL    R1                              ; 1791
                04      08      AC  D1 0018A          CMPL    ELEM_SIZE, #4
                                06  12 0018E          BNEQ    22$
                                51  D6 00190          INCL    R1
                                50  D4 00192          CLRL    R0
                                04  11 00194          BRB     23$
                50      04      A0  D0 00196 22$:     MOVL    4(R0), R0
            2C  AE              50  D0 0019A 23$:     MOVL    R0, D_VALUE+4
                04              51  E9 0019E          BLBC    R1, 24$                         ; 1794
                                7E  D4 001A1          CLRL    -(SP)
                                05  11 001A3          BRB     25$
                7E      FF70    CB  98 001A5 24$:     CVTBL   -144(CCB), -(SP)
                            3C  AE  9F 001AA 25$:     PUSHAB  DSC                             ; 1793
                            1C  AE  9F 001AD          PUSHAB  RET_LENGTH
                                7E  D4 001B0          CLRL    -(SP)
                            38  AE  9F 001B2          PUSHAB  D_VALUE
        00000000G           00      FB 001B5          CALLS   #5, BAS$CVT_OUT_D_G
                03                  E8 001BC          BLBS    R0, 26$
                                01E4 31 001BF         BRW     53$
                            0C  AE  D4 001C2 26$:     CLRL    L_ELEM_SIZE                     ; 1797
                38  AE          14  AE  B0 001C5      MOVW    RET_LENGTH, DSC                 ; 1798
                50              38  AE  3C 001CA      MOVZWL  DSC, R0                         ; 1799
                67              50  C0 001CE          ADDL2   R0, (R7)
                50              38  AE  3C 001D1      MOVZWL  DSC, R0
                66              50  C0 001D5          ADDL2   R0, (R6)
                                5A  D4 001D8          CLRL    R10
                02      10      AC  D1 001DA          CMPL    FORMAT_CHAR, #2
                                36  12 001DE          BNEQ    28$
                                5A  D6 001E0          INCL    R10
                50      B4      AB  66 C3 001E2       SUBL3   (R6), -76(CCB), R0
                                0E  50 D1 001E7       CMPL    R0, #14
                                    0E 1B 001EA       BLEQU   27$
        7E      00              67  01 7A 001EC       EMUL    #1, (R7), #0, -(SP)
        50      50              8E  0E 7B 001F1       EDIV    #14, (SP)+, R0, R0
                50              0E  50 C3 001F6       SUBL3   R0, #14, R0
        50      20              6E  00 2C 001FA 27$:  MOVC5   #0, (SP), #32, R0, @0(R6)
                                    00 B6 001FF
                                66  53 D0 00201      MOVL    R3, (R6)
        7E      00              67  01 7A 00204       EMUL    #1, (R7), #0, -(SP)
        50      50              8E  0E 7B 00209       EDIV    #14, (SP)+, R0, R0
                50              67  50 C3 0020E       SUBL3   R0, (R7), R0
                                67  0E A0 9E 00212    MOVAB   14(R0), (R7)
                                03  5A E9 00216 28$:  BLBC    R10, 29$
                                    0094 31 00219     BRW     36$
                                    0088 31 0021C 29$: BRW    35$
                50      B4      AB  66 C3 0021F 30$:  SUBL3   (R6), -76(CCB), R0              ; 1813
                                0E  50 D1 00224       CMPL    R0, #14
                                    03 1B 00227       BLEQU   31$
                                50  0E D0 00229      MOVL    #14, R0
                            38  AE  50 B0 0022C 31$:  MOVW    R0, DSC
                                50  0C AC D0 00230    MOVL    ELEM_ADR, R0                    ; 1815
                            28  AE  60 7D 00234      MOVQ    (R0), G_VALUE
                                38  AE 9F 00238      PUSHAB  DSC                             ; 1818
                                1C  AE 9F 0023B      PUSHAB  RET_LENGTH
                                    7E D4 0023E      CLRL    -(SP)
```

```
                                        34    AE  9F 00240         PUSHAB  G_VALUE                         :
                        00000000G  00         04  FB 00243         CALLS   #4, BAS$CVT_OUT_G_G             :
                                   03         50  E8 0024A         BLBS    R0, 32$                         :
                                            0156  31 0024D         BRW     53$                             :
                                        0C    AE  D4 00250 32$:    CLRL    L_ELEM_SIZE                     : 1821
                             38    AE   18    AE  B0 00253         MOVW    RET_LENGTH, DSC                 : 1822
                                   50   38    AE  3C 00258         MOVZWL  DSC, R0                         : 1823
                                   67         50  C0 0025C         ADDL2   R0, (R7)                        :
                                   50   38    AE  3C 0025F         MOVZWL  DSC, R0                         :
                                   66         50  C0 00263         ADDL2   R0, (R6)                        :
                                              5A  D4 00266         CLRL    R10                             :
                                   02   10    AC  D1 00268         CMPL    FORMAT_CHAR, #2                 :
                                              36  12 0026C         BNEQ    34$                             :
                                              5A  D6 0026E         INCL    R10                             :
                   50        B4    AB         66  C3 00270         SUBL3   (R6), -76(CCB), R0              :
                                   0E         50  D1 00275         CMPL    R0, #14                         :
                                              0E  1B 00278         BLEQU   33$                             :
     7E            00              67         01  7A 0027A         EMUL    #1, (R7), #0, -(SP)             :
     50            50              8E         0E  7B 0027F         EDIV    #14, (SP)+, R0, R0              :
                   50              0E         50  C3 00284         SUBL3   R0, #14, R0                     :
     50            20              6E         00  2C 00288 33$:    MOVC5   #0, (SP), #32, R0, @0(R6)       :
                                        00    B6     0028D                                                 :
                                   66         53  D0 0028F         MOVL    R3, (R6)                        :
     7E            00              67         01  7A 00292         EMUL    #1, (R7), #0, -(SP)             :
     50            50              8E         0E  7B 00297         EDIV    #14, (SP)+, R0, R0              :
                   50              67         50  C3 0029C         SUBL3   R0, (R7), R0                    :
                                   67   0E    A0  9E 002A0         MOVAB   14(R0), (R7)                    :
                                   09         5A  E8 002A4 34$:    BLBS    R10, 36$                        :
                                   01   10    AC  D1 002A7 35$:    CMPL    FORMAT_CHAR, #1                 :
                                              03  13 002AB         BEQL    36$                             :
                                            01F1  31 002AD         BRW     71$                             :
     67       D4   AB              10         00  ED 002B0 36$:    CMPZV   #0, #16, -44(CCB), (R7)         :
                                              03  1F 002B6         BLSSU   37$                             :
                                            FE96  31 002B8         BRW     16$                             :
                                            FE8E  31 002BB 37$:    BRW     15$                             :
                   50        B4    AB         66  C3 002BE 38$:    SUBL3   (R6), -76(CCB), R0              : 1836
                                   0E         50  D1 002C3         CMPL    R0, #14                         :
                                              03  1B 002C6         BLEQU   39$                             :
                                   50         0E  D0 002C8         MOVL    #14, R0                         :
                             38    AE         50  B0 002CB 39$:    MOVW    R0, DSC                         :
                                   50   0C    AC  D0 002CF         MOVL    ELEM_ADR, R0                    : 1838
                             20    AE         60  7D 002D3         MOVQ    (R0), H_VALUE                   :
                             28    AE   08    A0  7D 002D7         MOVQ    8(R0), H_VALUE+8                : 1840
                                        38    AE  9F 002DC         PUSHAB  DSC                             : 1843
                                        20    AE  9F 002DF         PUSHAB  RET_LENGTH                      :
                                              7E  D4 002E2         CLRL    -(SP)                           :
                                        2C    AE  9F 002E4         PUSHAB  H_VALUE                         :
                        00000000G  00         04  FB 002E7         CALLS   #4, BAS$CVT_OUT_H_G             :
                                   03         50  E8 002EE         BLBS    R0, 40$                         :
                                            00B2  31 002F1         BRW     53$                             :
                                        0C    AE  D4 002F4 40$:    CLRL    L_ELEM_SIZE                     : 1846
                             38    AE   1C    AE  B0 002F7         MOVW    RET_LENGTH, DSC                 : 1847
                                   50   38    AE  3C 002FC         MOVZWL  DSC, R0                         : 1848
                                   67         50  C0 00300         ADDL2   R0, (R7)                        :
                                   50   38    AE  3C 00303         MOVZWL  DSC, R0                         :
                                   66         50  C0 00307         ADDL2   R0, (R6)                        :
                                              5A  D4 0030A         CLRL    R10                             :
```

```
                                02      10  AC  D1 0030C          CMPL    FORMAT_CHAR, #2                                 ;
                                            36  12 00310          BNEQ    42$                                             ;
                                            5A  D6 00312          INCL    R10                                             ;
                50      B4  AB              66  C3 00314          SUBL3   (R6), -76(CCB), R0                              ;
                                            0E  50 D1 00319       CMPL    R0, #14                                         ;
                                                0E 1B 0031C       BLEQU   41$                                             ;
        7E      00          67              01  7A 0031E          EMUL    #1, (R7), #0, -(SP)                             ;
        50      50          8E              0E  7B 00323          EDIV    #14, (SP)+, R0, R0                              ;
                50          0E              50  C3 00328          SUBL3   R0, #14, R0                                     ;
        50      20          6E              00  2C 0032C 41$:     MOVC5   #0, (SP), #32, R0, @0(R6)                       ;
                                        00  B6     00331                                                                ;
                                        66  53 D0 00333          MOVL    R3, (R6)                                        ;
        7E      00          67              01  7A 00336          EMUL    #1, (R7), #0, -(SP)                             ;
        50      50          8E              0E  7B 0033B          EDIV    #14, (SP)+, R0, R0                              ;
                50          67              50  C3 00340          SUBL3   R0, (R7), R0                                    ;
                            67          0E  A0  9E 00344          MOVAB   14(R0), (R7)                                    ;
                            09              5A  E8 00348 42$:     BLBS    R10, 43$                                        ;
                            01          10  AC  D1 0034B          CMPL    FORMAT_CHAR, #1                                 ;
                                            03  13 0034F          BEQL    43$                                             ;
                                          014D  31 00351          BRW     71$                                             ;
    67  D4  AB                10              00  ED 00354 43$:     CMPZV   #0, #16, -44(CCB), (R7)                         ;
                                            05  1E 0035A          BGEQU   44$                                             ;
                19      A1  AB              01  E1 0035C          BBC     #1, -95(CCB), 48$                               ;
                    B4  AB                  66  D1 00361 44$:     CMPL    (R6), -76(CCB)                                  ;
                                            13  1A 00365          BGTRU   48$                                             ;
                                        0C  AE  D4 00367 45$:     CLRL    L_ELEM_SIZE                                     ;
                                02          5A  E9 0036A          BLBC    R10, 46$                                        ;
                                50          0E  D0 0036D          MOVL    #14, R0                                         ;
                                            02  11 00370          BRB     47$                                             ;
                                            50  D4 00372 46$:     CLRL    R0                                              ;
                                50          67  C0 00374 47$:     ADDL2   (R7), R0                                        ;
                                          00DC  31 00377          BRW     64$                                             ;
                03      FE  AB              03  E1 0037A 48$:     BBC     #3, -2(CCB), 49$                                ;
                                          FDE4  31 0037F          BRW     19$                                             ;
                                          FDDA  31 00382 49$:     BRW     18$                                             ;
                                50      38  AE  3C 00385 50$:     MOVZWL  DSC, R0                                         ;
                                66      BC BB40 9E 00389          MOVAB   @-68(CCB)[R0], (R6)                             ;
                            10  AC              01  D0 0038E          MOVL    #1, FORMAT_CHAR                             ;
                                            0C  AE  D4 00392          CLRL    L_ELEM_SIZE                                 ;
                03      A1  AB              01  E1 00395 51$:     BBC     #1, -95(CCB), 52$                               ;
                                          00FF  31 0039A          BRW     70$                                             ;
    67  D4  AB                10              00  ED 0039D 52$:     CMPZV   #0, #16, -44(CCB), (R7)                         ;
                                          00EF  31 003A3          BRW     69$                                             ;
                                52              08  D0 003A6 53$:     MOVL    #8, WRITE_STATUS                            ; 1843
                                          013B  31 003A9          BRW     78$                                             ;
                                06      08  AE  E9 003AC 54$:     BLBC    8(SP), 55$                                      ; 1872
                                51      34  AE  D0 003B0          MOVL    TEMP_DEC_DSC+4, SRC_LOC                         ; 1874
                                            08  11 003B4          BRB     56$                                             ;
                                50      0   AC  D0 003B6 55$:     MOVL    ELEM_ADR, R0                                    ; 1876
                                51      4   A0  D0 003BA          MOVL    4(R0), SRC_LOC                                  ;
                                        58  AE  9F 003BE 56$:     PUSHAB  DSC                                             ; 1878
                7E      B4  AB              66  C3 003C1          SUBL3   (R6), -76(CCB), -(SP)                           ; 1879
                                        14  AE  DD 003C6          PUSHL   L_ELEM_SIZE                                     ; 1878
                                        0C BE41 9F 003C9          PUSHAB  @STR_PTR[SRC_LOC]                               ;
                            0000V   CF          04  FB 003CD          CALLS   #4, SCAN_TEXT                               ;
                                57              50  D0 003D2          MOVL    R0, SCAN_STATUS                             ;
                                03              57  E8 003D5          BLBS    SCAN_STATUS, 57$                            ;
```

```
                                                00CE  31 003D8          BRW     73$
                                        56      B0  AB 9E 003DB 57$:    MOVAB   -80(CCB), R6                    ;1889
                                        50      38  AE 3C 003DF         MOVZWL  DSC, R0
                                        66          50 C0 003E3         ADDL2   R0, (R6)
                                                    5A D4 003E6         CLRL    R10
                                        02      10  AC D1 003E8         CMPL    FORMAT_CHAR, #2
                                                    3A 12 003EC         BNEQ    59$
                                                    5A D6 003EE         INCL    R10
                                50          B4  AB 66 C3 003F0         SUBL3   (R6), -76(CCB), R0
                                            0E     50 D1 003F5         CMPL    R0, #14
                                                   0F 1B 003F8         BLEQU   58$
                            7E              00  C8 AB 01 7A 003FA      EMUL    #1, -56(CCB), #0, -(SP)
                            50              50     8E 0E 7B 00400      EDIV    #14, (SP)+, R0, R0
                                            50     0E 50 C3 00405      SUBL3   R0, #14, R0
                            50              20     6E 00 2C 00409 58$: MOVC5   #0, (SP), #32, R0, @0(R6)
                                                00  B6      0040E
                                        66          53 D0 00410         MOVL    R3, (R6)
                            7E              00  C8 AB 01 7A 00413      EMUL    #1, -56(CCB), #0, -(SP)
                            50              50     8E 0E 7B 00419      EDIV    #14, (SP)+, R0, R0
                                            50  C8 AB 50 C3 0041E      SUBL3   R0, -56(CCB), R0
                                            C8  AB 0E A0 9E 00423      MOVAB   14(R0), -56(CCB)
                                        06          5A E8 00428 59$:    BLBS    R10, 60$
                                        01      10  AC D1 0042B         CMPL    FORMAT_CHAR, #1
                                                    70 12 0042F         BNEQ    71$
                    C8  AB      D4  AB          10     00 ED 00431 60$: CMPZV   #0, #16, -44(CCB), -56(CCB)
                                                    05 1E 00438         BGEQU   61$
                                29          A1  AB 01 E1 0043A         BBC     #1, -95(CCB), 65$
                                            B4  AB 66 D1 0043F 61$:     CMPL    (R6), -76(CCB)
                                                    23 1A 00443         BGTRU   65$
                                                0C  AE D4 00445         CLRL    L_ELEM_SIZE
                                        05          5A E9 00448         BLBC    R10, 62$
                                        50          0E D0 0044B         MOVL    #14, R0
                                                    02 11 0044E         BRB     63$
                                                    50 D4 00450 62$:    CLRL    R0
                                        50      C8  AB C0 00452 63$:    ADDL2   -56(CCB), R0
                        50              D4  AB      10     00 ED 00456 64$: CMPZV   #0, #16, -44(CCB), R0
                                                    43 1E 0045C         BGEQU   71$
                                3E          A1  AB 01 E0 0045E         BBS     #1, -95(CCB), 71$
                                            50     02 D0 00463         MOVL    #2, R0
                                                    3C 11 00466         BRB     72$
                                04          FE  AB 03 E0 00468 65$:     BBS     #3, -2(CCB), 66$
                                                    58 D5 0046D         TSTL    SAV_PRINT_POS
                                                    08 13 0046F         BEQL    67$
                                                38  AE B4 00471 66$:    CLRW    DSC
                                        66          59 D0 00474         MOVL    A_SAV_PTR, (R6)
                                                    10 11 00477         BRB     68$
                                        50      38  AE 3C 00479 67$:    MOVZWL  DSC, R0
                                        66      BC  BB40 9E 0047D       MOVAB   @-68(CCB)[R0], (R6)
                                            10      AC 01 D0 00482      MOVL    #1, FORMAT_CHAR
                                                0C  AE D4 00486         CLRL    L_ELEM_SIZE
                                            0E      AF 01 E0 00489 68$: BBS     #7, -95(CCB), 70$
                    C8  AB      D4  AB          10     00 ED 0048E      CMPZV   #0, #16, -44(CCB), -56(CCB)
                                                    05 1E 00495 69$:    BGEQU   70$
                                        50          06 D0 00497         MOVL    #6, R0
                                                    08 11 0049A         BRB     72$
                                        50          08 D0 0049C 70$:    MOVL    #8, R0
                                                    03 11 0049F         BRB     72$
```

```
                        50              01  D0 004A1 71$:   MOVL    #1, R0                                  ;
                        52              50  D0 004A4 72$:   MOVL    R0, WRITE_STATUS                        ;
                                        3E  11 004A7         BRB     78$                                     ;
                    0E      FE  AB      03  E1 004A9 73$:   BBC     #3, -2(CCB), 74$                        ; 1909
                        04              57  D1 004AE         CMPL    SCAN_STATUS, #4                         ;
                                        09  13 004B1         BEQL    74$                                     ;
                                38      AE  B4 004B3         CLRW    DSC                                     ; 1918
                            B0  AB      59  D0 004B6         MOVL    A_SAV_PTR, -80(CCB)                     ; 1919
                                        28  11 004BA         BRB     77$                                     ; 1909
                        50      38      AE  3C 004BC 74$:    MOVZWL  DSC, R0                                 ; 1929
                        6E              50  C0 004C0         ADDL2   R0, STR_PTR                             ;
                        50      38      AE  3C 004C3         MOVZWL  DSC, R0                                 ; 1930
                            0C  AE          50  C2 004C7     SUBL2   R0, L_ELEM_SIZE                         ;
                        51      38      AE  3C 004CB         MOVZWL  DSC, R1                                 ; 1931
                        51      B0      AB  C0 004CF         ADDL2   -80(CCB), R1                            ;
                        04              57  D1 004D3         CMPL    SCAN_STATUS, #4                         ; 1932
                                        05  12 004D6         BNEQ    75$                                     ;
                        50              02  D0 004D8         MOVL    #2, R0                                  ;
                                        02  11 004DB         BRB     76$                                     ;
                                        50  D4 004DD 75$:    CLRL    R0                                      ;
                B0  AB          51      50  C3 004DF 76$:    SUBL3   R0, R1, -80(CCB)                        ;
                        52              57  D0 004E4 77$:    MOVL    SCAN_STATUS, WRITE_STATUS               ; 1935
                        1E              52  E8 004E7 78$:    BLBS    WRITE_STATUS, 82$                       ; 1741
                        50              52  D0 004EA         MOVL    WRITE_STATUS, R0                        ; 1950
                                0000V   30  004ED            BSBW    BAS$$DO_WRITE                           ;
                        02              52  D1 004F0         CMPL    WRITE_STATUS, #2                        ; 1952
                                        05  13 004F3         BEQL    79$                                     ;
                        04              52  D1 004F5         CMPL    WRITE_STATUS, #4                        ;
                                        03  12 004F8         BNEQ    80$                                     ;
                                0C      AE  D5 004FA 79$:    TSTL    L_ELEM_SIZE                             ; 1953
                                        03  13 004FD 80$:    BEQL    81$                                     ;
                                    FB69    31 004FF         BRW     6$                                      ;
                            FE  AB          04  8A 00502 81$:   BICB2   #4, -2(CCB)                         ; 1970
                                            04  11 00506        BRB     83$                                 ; 1956
                            FE  AB          08  88 00508 82$:   BISB2   #8, -2(CCB)                         ; 1979
                                15      04      AC  D1 0050C 83$:   CMPL    ELEM_TYPE, #21                  ; 1989
                                            0A  12 00510        BNEQ    84$                                 ;
                                    30      AE  9F 00512        PUSHAB  TEMP_DEC_DSC                        ; 1991
                                00000000G   00  FB 00515        CALLS   #1, STR$FREE1_DX                    ;
                                        35  FF71    CB  91 0051C 84$:   CMPB    -143(CCB), #53              ; 1998
                                            0E  12 00521        BNEQ    85$                                 ;
                        09      FE  AB          02  E0 00523        BBS     #2, -2(CCB), 85$                ;
                                        00000000G   00  16 00528    JSB     BAS$$REC_MPR1                   ; 2001
                                            C8  AB  D4 0052E        CLRL    -56(CCB)                        ; 2002
                            04  AE          A0  AB  9E 00531 85$:   MOVAB   -96(R11), 4(SP)                 ; 2005
                        03  04  BE              05  E0 00536        BBS     #5, @4(SP), 86$                 ;
                                            FACD    31 0053B        BRW     1$                              ;
                            04  BE              20  8A 0053E 86$:   BICB2   #32, @4(SP)                     ; 2007
                                                04 00542        RET                                         ; 2009

; Routine Size:  1347 bytes,    Routine Base:  _BAS$CODE + 0030

;  971          2010  1
```

```
;  973          2011  1 GLOBAL ROUTINE BAS$$UDF_WL9 : JSB_UDF9 NOVALUE =
;  974          2012  1 !
;  975          2013  1 !++
;  976          2014  1 ! FUNCTIONAL DESCRIPTION:
;  977          2015  1 !
;  978          2016  1 !       Call the record level I/O end of list routine.  Reset the cursor position
;  979          2017  1 !       if a PUT was done
;  980          2018  1 !
;  981          2019  1 ! FORMAL PARAMETERS:
;  982          2020  1 !
;  983          2021  1 !       NONE
;  984          2022  1 !
;  985          2023  1 ! IMPLICIT INPUTS:
;  986          2024  1 !
;  987          2025  1 !       LUB$V_AST_GUARD         Guard for AST reentrancy
;  988          2026  1 !       LUB$V_FORM_CHAR         last element transmitter ended with a format char
;  989          2027  1 !
;  990          2028  1 ! IMPLICIT OUTPUTS:
;  991          2029  1 !
;  992          2030  1 !       LUB$V_AST_GUARD         guard for AST reentrancy
;  993          2031  1 !       LUB$L_PRINT_POS         current cursor position
;  994          2032  1 !
;  995          2033  1 ! ROUTINE VALUE:
;  996          2034  1 ! COMPLETION CODES:
;  997          2035  1 !
;  998          2036  1 !       NONE
;  999          2037  1 !
; 1000          2038  1 ! SIDE EFFECTS:
; 1001          2039  1 !
; 1002          2040  1 !       This routine will loop back and reexecute if it detects that it was
; 1003          2041  1 !       called by an AST while it was executing.
; 1004          2042  1 !
; 1005          2043  1 !--
; 1006          2044  1
; 1007          2045  2     BEGIN
; 1008          2046  2
; 1009          2047  2     EXTERNAL REGISTER
; 1010          2048  2         CCB : REF BLOCK [, BYTE];
; 1011          2049  2
; 1012          2050  2     !+
; 1013          2051  2     ! This outer loop is to detect an AST calling this routine while it is
; 1014          2052  2     ! executing.
; 1015          2053  2     !-
; 1016          2054  2
; 1017          2055  2     DO
; 1018          2056  3         BEGIN
; 1019          2057  3         CCB [LUB$V_AST_GUARD] = 1;              ! Initialize the guard bit
; 1020          2058  3 !+
; 1021          2059  3 ! Clear the bit which indicates an array list for MAT PRINT.
; 1022          2060  3 !-
; 1023          2061  3         CCB [ISB$V_MAT_PRINT] = 0;
; 1024          2062  3 !+
; 1025          2063  3 ! Dispatch to the REC level.
; 1026          2064  3 !-
; 1027          2065  3         JSB_REC9(BAS$$AA_REC_PR9 + .BAS$$AA_REC_PR9 [.CCB [ISB$B_STTM_TYPE] - ISB$K_BASSTTYLO + 1]);
; 1028          2066  3         ! Time to reset the cursor position to zero perhaps
; 1029          2067  3
```

```
;  1030          2068  3          IF NOT .CCB [LUB$V_FORM_CHAR] THEN CCB [LUB$L_PRINT_POS] = 0;
;  1031          2069  3
;  1032          2070  3          END
;  1033          2071  2      UNTIL .CCB [LUB$V_AST_GUARD];                ! End of AST guard loop
;  1034          2072  2
;  1035          2073  2      CCB [LUB$V_AST_GUARD] = 0;
;  1036          2074  1      END;


                                        52  DD 00000 BAS$$UDF_WL9::
                                                       PUSHL   R2                                      ; 2011
                                52       A0   AB  9E 00002                 MOVAB   -96(R11), R2          ; 2057
                                62            20  88 00006 1$:            BISB2   #32, (R2)             ;
                            97  AB            04  8A 00009                 BICB2   #4, -105(CCB)         ; 2061
                                50       FF71 CB  9A 0000D                 MOVZBL  -143(CCB), R0         ; 2065
                                50 00000000G0040  D0 00012                 MOVL    BAS$$AA_REC_PR9-104[R0], R0   ;
                                   00000000G0040  16 00018                 JSB     BAS$$AA_REC_PR9[R0]   ;
                        03      FE   AB           02  E0 00021             BBS     #2, -2(CCB), 2$       ; 2068
                                          C8  AB  D4 00026                 CLRL    -56(CCB)              ;
                                52       A0   AB  9E 00029 2$:            MOVAB   -96(R11), R2          ; 2071
                        D5      62            05  E1 0002D                 BBC     #5, (R2), 1$          ;
                                62            20  8A 00031                 BICB2   #32, (R2)             ; 2073
                                              04  BA 00034                 POPR    #^M<R2>               ; 2074
                                                  05 00036                 RSB                           ;

;  Routine Size:  55 bytes,    Routine Base:  _BAS$CODE + 0573

;  1037          2075  1
```

```
;  1039          2076  1 ROUTINE SCAN_TEXT (POS,                          ! adr of beginning position of source string
;  1040          2077  1              LENGTH,                             ! length of the source string
;  1041          2078  1              BUF_LENGTH,                         ! length of the buffer remaining
;  1042          2079  1              RET_STR                             ! adr of where to put the string
;  1043          2080  1          ) : CALL_CCB =
;  1044          2081  1
;  1045          2082  1 !++
;  1046          2083  1 ! FUNCTIONAL DESCRIPTION:
;  1047          2084  1 !
;  1048          2085  1 !       This routine puts a string of ASCII characters into the indicated buffer.
;  1049          2086  1 !       All non-printing characters, backspace, K_TAB, and carriage return
;  1050          2087  1 !       are identified in the string and their effect is reflected in the
;  1051          2088  1 !       internal cursor position.
;  1052          2089  1 !
;  1053          2090  1 ! FORMAL PARAMETERS:
;  1054          2091  1 !
;  1055          2092  1 !       POS.rlu.r               position of the start of the string to output
;  1056          2093  1 !       LENGTH.rlu.v            length of the string to write
;  1057          2094  1 !       BUF_LENGTH.rlu.v        length remaining of the output buffer
;  1058          2095  1 !       RET_STR.wt.dx           pointer to descriptor for the destination
;  1059          2096  1 !
;  1060          2097  1 ! IMPLICIT INPUTS:
;  1061          2098  1 !
;  1062          2099  1 !       LUB$L_PRINT_POS         current cursor position
;  1063          2100  1 !       LUB$W_R_MARGIN          maximum allowable cursor position
;  1064          2101  1 !       LUB$A_BUF_PTR           current position in the buffer
;  1065          2102  1 !
;  1066          2103  1 ! IMPLICIT OUTPUTS:
;  1067          2104  1 !
;  1068          2105  1 !       LUB$L_PRINT_POS         current cursor position
;  1069          2106  1 !
;  1070          2107  1 ! ROUTINE VALUE:
;  1071          2108  1 !
;  1072          2109  1 !       BAS$K_MAR_EXC           right margin exceeded
;  1073          2110  1 !       BAS$K_CRLF              embedded CRLF in the record
;  1074          2111  1 !       BAS$K_SUCCESS           successful completion
;  1075          2112  1 !
;  1076          2113  1 ! SIDE EFFECTS:
;  1077          2114  1 !
;  1078          2115  1 !       NONE
;  1079          2116  1 !
;  1080          2117  1 !--
;  1081          2118  1
;  1082          2119  2     BEGIN
;  1083          2120  2
;  1084          2121  2     MAP
;  1085          2122  2         POS : REF VECTOR [, BYTE],
;  1086          2123  2         RET_STR : REF BLOCK [8, BYTE];
;  1087          2124  2
;  1088          2125  2     EXTERNAL REGISTER
;  1089          2126  2         CCB : REF BLOCK [, BYTE];
;  1090          2127  2
;  1091          2128  2     LITERAL
;  1092          2129  2         K_STOP = %X'00',                           ! stop or escape character for MOVTUC
;  1093          2130  2         K_TILDE = %X'7E',                          ! tilde character
;  1094          2131  2         K_LF = %X'0A',                             ! line feed character
;  1095          2132  2         K_BKSP = %X'08',                           ! backspace character
```

```
;  1096          2133  2          K_TAB = %X'09',                         ! tab character
;  1097          2134  2          K_CR = %X'0D',                          ! carriage return character
;  1098          2135  2          K_BLANK = %X'20',                       ! blank character
;  1099          2136  2          K_TAB_LIT = %X'07';
;  1100          2137  2
;  1101          2138  2      BIND
;  1102          2139  2          ESC = UPLIT BYTE(K_STOP),
;  1103          2140  2
;  1104          2141  2          !+
;  1105          2142  2          ! This translation table is used for a MOVTUC instruction.  It trans-
;  1106          2143  2          ! lates all normal 7 bit alphanumerics into themselves and stops on
;  1107          2144  2          ! characters which affect the cursor position (backspace, tab, and CR).
;  1108          2145  2          !-
;  1109          2146  2
;  1110          2147  2          TRANS_TABLE = UPLIT BYTE(%X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X
;  1111          2148  2              '00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00'          ! column 0
;  1112          2149  2              %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00',
;  1113          2150  2              %X'00', %X'00', %X'00', %X'00',                ! column 1
;  1114          2151  2              %X'20', %X'21', %X'22', %X'23', %X'24', %X'25', %X'26', %X'27', %X'28', %X'29', %X'2A', %X'2B',
;  1115          2152  2              %X'2C', %X'2D', %X'2E', %X'2F',                ! column 2
;  1116          2153  2              %X'30', %X'31', %X'32', %X'33', %X'34', %X'35', %X'36', %X'37', %X'38', %X'39', %X'3A', %X'3B',
;  1117          2154  2              %X'3C', %X'3D', %X'3E', %X'3F',                ! column 3
;  1118          2155  2              %X'40', %X'41', %X'42', %X'43', %X'44', %X'45', %X'46', %X'47', %X'48', %X'49', %X'4A', %X'4B',
;  1119          2156  2              %X'4C', %X'4D', %X'4E', %X'4F',                ! column 4
;  1120          2157  2              %X'50', %X'51', %X'52', %X'53', %X'54', %X'55', %X'56', %X'57', %X'58', %X'59', %X'5A', %X'5B',
;  1121          2158  2              %X'5C', %X'5D', %X'5E', %X'5F',                ! column 5
;  1122          2159  2              %X'60', %X'61', %X'62', %X'63', %X'64', %X'65', %X'66', %X'67', %X'68', %X'69', %X'6A', %X'6B',
;  1123          2160  2              %X'6C', %X'6D', %X'6E', %X'6F',                ! column 6
;  1124          2161  2              %X'70', %X'71', %X'72', %X'73', %X'74', %X'75', %X'76', %X'77', %X'78', %X'79', %X'7A', %X'7B',
;  11_5          2162  2              %X'7C', %X'7D', %X'7E', %X'00',                ! column 7
;  1126          2163  2              %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00',
;  1127          2164  2              %X'00', %X'00', %X'00', %X'00',                ! column 8
;  1128          2165  2              %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00',
;  1129          2166  2              %X'00', %X'00', %X'00', %X'00',                ! column 9
;  1130          2167  2              %X'A0', %X'A1', %X'A2', %X'A3', %X'A4', %X'A5', %X'A6', %X'A7', %X'A8', %X'A9', %X'AA', %X'AB',
;  1131          2168  2              %X'AC', %X'AD', %X'AE', %X'AF',                ! column 10
;  1132          2169  2              %X'B0', %X'B1', %X'B2', %X'B3', %X'B4', %X'B5', %X'B6', %X'B7', %X'B8', %X'B9', %X'BA', %X'BB',
;  1133          2170  2              %X'BC', %X'BD', %X'BE', %X'BF',                ! column 11
;  1134          2171  2              %X'C0', %X'C1', %X'C2', %X'C3', %X'C4', %X'C5', %X'C6', %X'C7', %X'C8', %X'C9', %X'CA', %X'CB',
;  1135          2172  2              %X'CC', %X'CD', %X'CE', %X'CF',                ! column 12
;  1136          2173  2              %X'D0', %X'D1', %X'D2', %X'D3', %X'D4', %X'D5', %X'D6', %X'D7', %X'D8', %X'D9', %X'DA', %X'DB',
;  1137          2174  2              %X'DC', %X'DD', %X'DE', %X'DF',                ! column 13
;  1138          2175  2              %X'E0', %X'E1', %X'E2', %X'E3', %X'E4', %X'E5', %X'E6', %X'E7', %X'E8', %X'E9', %X'EA', %X'EB',
;  1139          2176  2              %X'EC', %X'ED', %X'EE', %X'EF',                ! column 14
;  1140          2177  2              %X'F0', %X'F1', %X'F2', %X'F3', %X'F4', %X'F5', %X'F6', %X'F7', %X'F8', %X'F9', %X'FA', %X'FB',
;  1141          2178  2              %X'FC', %X'FD', %X'FE', %X'00')                ! column 15
;  1142          2179  2          : VECTOR [256, BYTE];
;  1143          2180  2
;  1144          2181  2      LOCAL
;  1145          2182  2          L_SAVE_LENGTH,                          ! save length of string scanned so far
;  1146          2183  2          DEST_LENGTH,                            ! length of destination buffer
;  1147          2184  2          SRC_LENGTH,                             ! length of source string
;  1148          2185  2          A_SAVE_BUF_PTR,                         ! temp to save LUB$A_BUF_PTR
;  1149          2186  2          T_DSC : BLOCK [8, BYTE],                ! working desc to hold the fourth argument
;  1150          2187  2          END_POS,                                ! no. of characters in source string + current
;  1151          2188  2                                                  ! address in the buffer
;  1152          2189  2          L_SAV_PRINT_POS,                        ! temp to save the cursor position
```

```
;  1153          2190  2                                    ! in case the right margin is exceeded
;  1154          2191  2          T_POS,                    ! local to hold readonly arg. POS
;  1155          2192  2          RET_VAL,                  ! the status value returned to the caller.
;  1156          2193  2          MOV_VAL,                  ! receives the returned status from MOVTUC
;  1157          2194  2          T_BUF_PTR,                ! temp buffer ptr, to agree with print pos
;  1158          2195  2          ORIG_BUF_PTR;             ! another buf ptr, unmodified so
;  1159          2196  2                                    ! LUB$A_BUF_PTR can be restored
;  1160          2197  2
;  1161          2198  2      !+
;  1162          2199  2      ! Save off a number of pointers.  We will not overflow the buffer because
;  1163          2200  2      ! the MOVTUC which is used to load the buffer has a destination length operand,
;  1164          2201  2      ! however it is possible to exceed the right margin if there are one or more
;  1165          2202  2      ! TAB characters.  If the margin is exceeded, it is detected in the macro
;  1166          2203  2      ! CHK_CURSOR_POS, then the string is backed up and each character is moved
;  1167          2204  2      ! until the right margin is reached.  Normally, if an element does not fit
;  1168          2205  2      ! into the buffer, the buffer is dumped without it.  However, there is the
;  1169          2206  2      ! case where a string is longer than the buffer and must be put out in seg-
;  1170          2207  2      ! ments.  That is not to be decided in this routine.
;  1171          2208  2      !-
;  1172          2209  2
;  1173          2210  2      DEST_LENGTH = .BUF_LENGTH;
;  1174          2211  2      SRC_LENGTH = .LENGTH;
;  1175          2212  2      T_POS = .POS;
;  1176          2213  2      T_BUF_PTR = .CCB [LUB$A_BUF_PTR];
;  1177          2214  2      ORIG_BUF_PTR = .CCB [LUB$A_BUF_PTR];
;  1178          2215  2      T_DSC [DSC$W_LENGTH] = 0;
;  1179          2216  2      T_DSC [DSC$A_POINTER] = .RET_STR [DSC$A_POINTER];
;  1180          2217  2      END_POS = MINU (.SRC_LENGTH, .DEST_LENGTH) + .T_POS;
;  1181          2218  2      MOV_VAL = 1;
;  1182          2219  2
;  1183          2220  2      !+
;  1184          2221  2      ! This loop will find all of the cursor position affecting characters in
;  1185          2222  2      ! the string.
;  1186          2223  2      !-
;  1187          2224  2
;  1188          2225  2      WHILE .MOV_VAL NEQU 0 DO
;  1189          2226  3          BEGIN
;  1190          2227  3          RET_VAL = (IF .DEST_LENGTH GEQ .SRC_LENGTH THEN BAS$K_SUCCESS ELSE BAS$K_BUF_EXC);
;  1191          2228  3          A_SAVE_BUF_PTR = .T_BUF_PTR;
;  1192          2229  3          L_SAV_PRINT_POS = .CCB [LUB$L_PRINT_POS];
;  1193          2230  3          L_SAVE_LENGTH = .T_DSC [DSC$W_LENGTH];
;  1194          2231  3          MOV_VAL = MOVTUC (SRC_LENGTH, .T_POS, ESC, TRANS_TABLE, DEST_LENGTH, .T_DSC [DSC$A_POINTER]);
;  1195          2232  3
;  1196          2233  3          !+
;  1197          2234  3          ! Set RET_VAL to either 1 (success) or 0 (failure) based on whether
;  1198          2235  3          ! there was enough room in the buffer for the entire translated
;  1199          2236  3          ! string.  A value of 1 at this point may be changed in the macro
;  1200          2237  3          ! CHK_CURSOR_POS.
;  1201          2238  3          !-
;  1202          2239  3
;  1203          2240  3          IF .MOV_VAL EQLU 0
;  1204          2241  3          THEN
;  1205          2242  3
;  1206          2243  3          !+
;  1207          2244  3          ! MOVTUC was able to move the entire source string without encountering
;  1208          2245  3          ! an 'escape' character.  Update the cursor position and check to
;  1209          2246  3          ! see if it exceeds the right margin.
```

```
;  1210          2247  3          !-
;  1211          2248  3
;  1212          2249  4              BEGIN
;  1213          2250  4              !
;  1214          2251  4              ! SWITCHES LIST (EXPAND);
;  1215          2252  4              !
;  1216          2253  4              T_DSC [DSC$W_LENGTH] = .T_DSC [DSC$W_LENGTH] + (.END_POS - .T_POS);
;  1217          2254  4              CCB [LUB$L_PRINT_POS] = .CCB [LUB$L_PRINT_POS] + (.END_POS - .T_POS);
;  1218          2255  4              T_BUF_PTR = .T_BUF_PTR + .SRC_LENGTH;
;  1219          2256  4              CRK_CURSOR_POS;
;  1220          2257  4              END
;  1221          2258  3          ELSE
;  1222          2259  4              BEGIN
;  1223          2260  4
;  1224          2261  4              !+
;  1225          2262  4              ! an 'escape' character was detected in the source string.
;  1226          2263  4              ! increment the cursor position for the substring moved plus one
;  1227          2264  4              ! for the offending character.  the buffer pointer must be moved
;  1228          2265  4              ! to keep up with the cursor position.
;  1229          2266  4              !-
;  1230          2267  4
;  1231          2268  4              CCB [LUB$L_PRINT_POS] = .CCB [LUB$L_PRINT_POS] + (.MOV_VAL - .T_POS);
;  1232          2269  4              T_BUF_PTR = .T_BUF_PTR + (.MOV_VAL - .T_POS) + 1;
;  1233          2270  4              T_DSC [DSC$W_LENGTH] = .T_DSC [DSC$W_LENGTH] + (.MOV_VAL - .T_POS) + 1;
;  1234          2271  4
;  1235          2272  4  !+
;  1236          2273  4  ! Check for an occurrence of a CRLF which will delimit the record and insure that
;  1237          2274  4  ! the LF is included in the buffer.
;  1238          2275  4  !-
;  1239          2276  4  !+
;  1240          2277  4  ! We should do a LSSU instead of LEQU, to fix 1-065D.
;  1241          2278  4  !-
;  1242          2279  4              IF .(.MOV_VAL)<0, 8> EQL K_CR
;  1243          2280  4                  AND .(MOV_VAL) + 1 LSSU .POS + .LENGTH
;  1244          2281  4                  AND .(.(MOV_VAL) + 1)<0, 8> EQL K_LF
;  1245          2282  4              THEN
;  1246          2283  5                  BEGIN
;  1247          2284  5  !+
;  1248          2285  5  ! There is a CRLF in this record.  Set the length properly to include the CRLF
;  1249          2286  5  ! and return the fact that a delimiting CRLF was found.
;  1250          2287  5  !-
;  1251          2288  5                  T_DSC [DSC$W_LENGTH] = .T_DSC [DSC$W_LENGTH] + 1;
;  1252          2289  5                  CRK_CURSOR_POS;
;  1253          2290  5                  RET_VAL = BAS$K_CRLF;
;  1254          2291  5                  EXITLOOP;
;  1255          2292  5                  END
;  1256          2293  5
;  1257          2294  5              !+
;  1258          2295  5              ! Adjust the cursor position for the 'escape' character if the
;  1259          2296  5              ! 'escape' character was not CR followed by LF.  We don't want
;  1260          2297  5              ! to reset PRINT_POS for CRLF because it is a complete record
;  1261          2298  5              ! and must be evaluated as such for both comma zone formatting and
;  1262          2299  5              ! exceeding the right margin.
;  1263          2300  5              !-
;  1264          2301  5
;  1265          2302  4              ELSE
;  1266          2303  5                  CCB [LUB$L_PRINT_POS] = (SELECTONE .(.MOV_VAL)<0, 8, 0> OF
```

```
5                      SET
5                      [K_TAB] : ((.CCB [LUB$L_PRINT_POS] OR K_TAB_LIT) + 1);
5                      [K_CR] : 0;
5                      [K_BKSP] : MAX (0, .CCB [LUB$L_PRINT_POS] - 1);
5                      [OTHERWISE] : .CCB [LUB$L_PRINT_POS];
4                      TES);
4              SRϹ_LENGTH = .SRC_LENGTH - (.MOV_VAL - .T_POS) - 1;
4              T_DSC [DSC$A_POINTER] = .T_DSC [DSC$A_POINTER] + (.MOV_VAL - .T_POS);
4              DEST_LENGTH = .DEST_LENGTH - (.MOV_VAC - .T_POS) - 1;
4              T_POS = .MOV_VAL + 1;
4
4              !+
4              ! move the offending character by hand
4              !-
4
4              T_DSC [DSC$A_POINTER] = CH$MOVE (1, CH$PTR (.MOV_VAL), CH$PTR (.T_DSC [DSC$A_POINTER]));
4              CĦK_CURSOR_POS;
3              END;
3
2          END;                                        ! WHILE loop
2
2      !+
2      ! Set the length of the string which was transferred into the indicated
2      ! destination.  Restore original buffer pointer so that BAS$UDF_WL1 can
2      ! update it properly.
2      !-
2
2      CCB [LUB$A_BUF_PTR] = .ORIG_BUF_PTR;
2      RET_STR [DSC$W_LENGTH] = .T_DSC [DSC$W_LENGTH];
2      RETÜRN .RET_VAC;
1      END;                                        ! End of SCAN_TEXT
```

```
                                                00  005AA P.AAA:  .BYTE  0                                              ;
00 00 00 00 00 00 00 00 00 00 00 00 00 00       00  005AB P.AAB:  .BYTE  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -   ;
00 00 00 00 00 00 00 00 00 00 00 00 00 00       00  005BA                0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -   ;
2C 2Ɓ 2A 29 28 27 26 25 24 23 22 21 20 00       00  005C9                0, 0, 0, 0, 32, 33, 34, 35, 36, 37, 38, -      ;
3B 3A 39 38 37 36 35 34 33 32 31 30 2F 2E       2D  005D8                39, 40, 41, 42, 43, 44, 45, 46, 47, 48, -      ;
4A 49 48 47 46 45 44 43 42 41 40 3F 3E 3D       3C  005E7                49, 50, 51, 52, 53, 54, 55, 56, 57, 58, -      ;
59 58 57 56 55 54 53 52 51 50 4F 4E 4D 4C       4B  005F6                59, 60, 61, 62, 63, 64, 65, 66, 67, 68, -      ;
68 67 66 65 64 63 62 61 60 5F 5E 5D 5C 5B       5A  00605                69, 70, 71, 72, 73, 74, 75, 76, 77, 78, -      ;
77 76 75 74 73 72 71 70 6F 6E 6D 6C 6B 6A       69  00614                79, 80, 81, 82, 83, 84, 85, 86, 87, 88, -      ;
00 00 00 00 00 00 00 00 7E 7D 7C 7B 7A 79       78  00623                89, 90, 91, 92, 93, 94, 95, 96, 97, 98, -      .
00 00 00 00 00 00 00 00 00 00 00 00 00 00       00  00632                99, 100, 101, 102, 103, 104, 105, 106, -       ;
A4 A3 A2 A1 A0 00 00 00 00 00 00 00 00 00       00  00641                107, 108, 109, 110, 111, 112, 113, 114, -      ;
B3 B2 B1 B0 AF AE AD AC AB AA A9 A8 A7 A6       A5  00650                115, 116, 117, 118, 119, 120, 121, 122, -      ;
C2 C1 C0 BF BE BD BC BB BA B9 B8 B7 B6 B5       B4  0065F                1.3, 124, 125, 126, 0, 0, 0, 0, 0, 0, 0, -     ;
D1 D0 CF CE CD CC CB CA C9 C8 C7 C6 C5 C4       C3  0066E                0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -    ;
E0 DF DE DD DC DB DA D9 D8 D7 D6 D5 D4 D3       D2  0067D                0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -96, -     ;
EF EE ED EC EB EA E9 E8 E7 E6 E5 E4 E3 E2       E1  0068C                -95, -94, -93, -92, -91, -90, -89, -88, -      ;
FE FD FC FB FA F9 F8 F7 F6 F5 F4 F3 F2 F1       F0  0069B                -87, -86, -85, -84, -83, -82, -81, -80, -      ;
                                                00  006AA                -79, -78, -77, -76, -75, -74, -73, -72, -      ;
                                                                         -71, -70, -69, -68, -67, -66, -65, -64, -      ;
                                                                         -63, -62, -61, -60, -59, -58, -57, -56, -      ;
                                                                         -55, -54, -53, -52, -51, -50, -49, -48, -      ;
                                                                         -47, -46, -45, -44, -43, -42, -41, -40, -      ;
```

```
                                -39, -38, -37, -36, -35, -34, -33, -32, -
                                -31, -30, -29, -28, -27, -26, -25, -24, -
                                -23, -22, -21, -20, -19, -18, -17, -16, -
                                -15, -14, -13, -12, -11, -10, -9, -8, -7, -
                                -6, -5, -4, -3, -2, 0

                        ESC=                    P.AAA
                        TRANS_TABLE=            P.AAB


                                07FC 00000 SCAN_TEXT:
                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10                        ; 2076
                             5E            2C   C2 00002          SUBL2   #44, SP                          ;
                             57      08    AC   7D 00005          MOVQ    LENGTH, SRC_LENGTH               ; 2211
                         20  AE      04    AC   D0 00009          MOVL    POS, T_POS                       ; 2212
                         1C  AE      B0    AB   9E 0000E          MOVAB   -80(CCB), 28(SP)                 ; 2213
                         18  AE      1C    BE   D0 00013          MOVL    @28(SP), T_BUF_PTR               ;
                         14  AE      1C    BE   D0 00018          MOVL    @28(SP), ORIG_BUF_PTR            ; 2214
                                     24    AE   B4 0001D          CLRW    T_DSC                            ; 2215
                    50   10  AC              04   C1 00020          ADDL3   #4, RET_STR, R0                  ; 2216
                         28  AE              60   D0 00025          MOVL    (R0), T_DSC+4                    ;
                             50              57   D0 00029          MOVL    SRC_LENGTH, R0                   ; 2217
                             58              50   D1 0002C          CMPL    R0, DEST_LENGTH                  ;
                                             03   1B 0002F          BLEQU   1$                               ;
                             50              58   D0 00031          MOVL    DEST_LENGTH, R0                  ;
                    59       50      20  AE  C1   00034 1$:        ADDL3   T_POS, R0, END_POS               ;
                             56              01   D0 00039          MOVL    #1, MOV_VAL                      ; 2218
                         10  AE      C8  AB   9E   0003C          MOVAB   -56(CCB), 16(SP)                 ; 2229
                                         56   D5 00041 2$:        TSTL    MOV_VAL                          ; 2225
                                         03   12 00043            BNEQ    4$                               ;
                                      025B   31 00045 3$:         BRW     43$                              ;
                             57           58   D1 00048 4$:        CMPL    DEST_LENGTH, SRC_LENGTH          ; 2227
                                          06   19 0004B            BLSS    5$                               ;
                                 0C  AE       01   D0 0004D          MOVL    #1, RET_VAL                      ;
                                          04   11 00051            BRB     6$                               ;
                                 0C  AE       08   D0 00053 5$:     MOVL    #8, RET_VAL                      ;
                                 08  AE  18   AE   D0 00057 6$:     MOVL    T_BUF_PTR, A_SAVE_BUF_PTR        ; 2228
                                 04  AE  10   BE   D0 0005C          MOVL    @T6(SP), L_SAV_PRINT_POS         ; 2229
                                     5A  24   AE   3C 00061          MOVZWL  T_DSC, R10                       ; 2230
                                     6E       5A   D0 00065          MOVL    R10, L_SAVE_LENGTH               ;
                FE8E    CF  FE90 CF      20  BE       57   2F 00068  MOVTUC  SRC_LENGTH, @T_POS, ESC, TRANS_TABLE, -  ; 2231
                                 28  BE       58        00072                DEST_LENGTH, @T_DSC+4            ;
                                          02   1D 00075            BVS     7$                               ;
                                          51   D4 00077            CLRL    R1                               ;
                                     56       51   D0 00079 7$:     MOVL    R1, MOV_VAL                      ;
                                          03   13 0007C            BEQL    8$                               ; 2240
                                       0095   31 0007E            BRW     17$                              ;
                             50           59  20  AE   C3 00081 8$:     SUBL3   T_POS, END_POS, R0           ; 2253
                                 24  AE       50   A0 00086          ADDW2   R0, T_DSC                        ;
                                 10  BE       50   C0 0008A          ADDL2   R0, @T6(SP)                      ; 2254
                                 18  AE       57   C0 0008E          ADDL2   SRC_LENGTH, T_BUF_PTR            ; 2255
                                 AA  A1   AB       01   E0 00092    BBS     #1, -95(CCB), 2$                 ;
                10  BE      D4  AB           10       00   ED 00097  CMPZV   #0, #16, -44(CCB), @16(SP)      ;
                                              A1   18 0009E          BGEQ    2$                               ;
                                 0C  AE       06   D0 000A0          MOVL    #6, RET_VAL                      ;
                                 10  BE  04   AE   D0 000A4          MOVL    L_SAV_PRINT_POS, @16(SP)         ;
                                 1C  BE  08   AE   D0 000A9          MOVL    A_SAVE_BUF_PTR, @28(SP)          ;
```

```
                                24  AE              6E  B0 000AE           MOVW    L_SAVE_LENGTH, T_DSC                    ;
                            8E  A1  AB              04  E0 000B2           BBS     #4, -95(CCB), 3$                        ;
      10    BE        D4    AB      10              00  ED 000B7 9$:       CMPZV   #0, #16, -44(CCB), @16(SP)              ;
                                                    03  14 000BE           BGTR    10$                                     ;
                                                01DB  31 000C0           BRW     42$                                     ;
                                    51          1C  BE  D0 000C3 10$:      MOVL    @28(SP), R1                             ;
                                    50              61  9A 000C7           MOVZBL  (R1), R0                                ;
                                    09              50  91 000CA           CMPB    R0, #9                                  ;
                                                    09  12 000CD           BNEQ    11$                                     ;
                                51      10  BE          07  C9 000CF           BISL3   #7, @16(SP), R1                         ;
                                                    51  D6 000D4           INCL    R1                                      ;
                                                    29  11 000D6           BRB     14$                                     ;
                                    0D              50  91 000D8 11$:      CMPB    R0, #13                                 ;
                                                    04  12 000DB           BNEQ    12$                                     ;
                                                    50  D4 000DD           CLRL    R0                                      ;
                                                    29  11 000DF           BRB     16$                                     ;
                                    08              5C  91 000E1 12$:      CMPB    R0, #8                                  ;
                                                    0B  12 000E4           BNEQ    13$                                     ;
                                51      10  BE          01  C3 000E6           SUBL3   #1, @16(SP), R1                         ;
                                                    14  18 000EB           BGEQ    14$                                     ;
                                                    51  D4 000ED           CLRL    R1                                      ;
                                                    10  11 000EF           BRB     14$                                     ;
                                    20              50  91 000F1 13$:      CMPB    R0, #32                                 ;
                                                    10  1F 000F4           BLSSU   15$                                     ;
                                7E  8F              50  91 000F6           CMPB    R0, #126                                ;
                                                    0A  1A 000FA           BGTRU   15$                                     ;
                                51      10  BE          01  C1 000FC           ADDL3   #1, @16(SP), R1                         ;
                                    50              51  D0 00101 14$:      MOVL    R1, R0                                  ;
                                                    04  11 00104           BRB     16$                                     ;
                                    50      10  BE      D0 00106 15$:      MOVL    @16(SP), R0                             ;
                                10  BE              50  D0 0010A 16$:      MOVL    R0, @16(SP)                             ;
                                                1C  BE  D6 0010E           INCL    @28(SP)                                 ;
                                                24  AE  B6 00111           INCW    T_DSC                                   ;
                                                    A1  11 00114           BRB     9$                                      ;
                                52              56      20  AE  C3 00116 17$:      SUBL3   T_POS, MOV_VAL, R2                      ; 2268
                                            10  BE          52  C0 0011B           ADDL2   R2, @16(SP)                     ;
                                                50      18  AE  D0 0011F           MOVL    T_BUF_PTR, R0                   ; 2269
                                                18  AE  01  A240  9E 00123         MOVAB   1(R2)[R0], T_BUF_PTR            ;
                                                50      01  A24A  9E 00129         MOVAB   1(R2)[R10], R0                  ; 2270
                                                24  AE      50  B0 0012E           MOVW    R0, T_DSC                       ;
                                                    0D      66  91 00132           CMPB    (MOV_VAL), #13                  ; 2279
                                                            0F  12 00135           BNEQ    18$                             ;
                                                    51      01  A6  9E 00137       MOVAB   1(R6), R1                       ; 2280
                                50              04  AC      08  AC  C1 0013B       ADDL3   LENGTH, POS, R0                 ;
                                                    50          51  D1 00141       CMPL    R1, R0                          ;
                                                                03  1F 00144       BLSSU   19$                             ;
                                                            008D  31 00146 18$:    BRW     29$                             ;
                                                    0A      01  A6  91 00149 19$:      CMPB    1(MOV_VAL), #10             ; 2281
                                                            F7  12 0014D           BNEQ    18$                             ;
                                                24  AE          B6 0014F           INCW    T_DSC                           ; 2288
                                78  A1  AB                  01  E0 00152           BBS     #1, -95(CCB), 27$               ;
      10    BE        D4    AB      10                      00  ED 00157           CMPZV   #0, #16, -44(CCB), @16(SP)      ;
                                                            6F  18 0015E           BGEQ    27$                             ;
                                0C  AE                      06  D0 00160           MOVL    #6, RET_VAL                     ;
                                10  BE              04  AE  D0 00164           MOVL    L_SAV_PRINT_POS, @16(SP)        ;
                                1C  BE              08  AE  D0 00169           MOVL    A_SAVE_BUF_PTR, @28(SP)         ;
                                24  AE                  6E  B0 0016E           MOVW    L_SAVE_LENGTH, T_DSC            ;
```

```
                  5C    A1    AB            04  E0 00172        BBS     #4, -95(CCB), 28$                      ;
 10   BE     D4   AB          10            00  ED 00177 20$:   CMPZV   #0, #16, -44(CCB), @16(SP)             ;
                                            03  14 0017E        BGTR    21$                                    ;
                                          011B  31 00180        BRW     42$                                    ;
                              50      1C    BE  D0 00183 21$:   MOVL    @28(SP), R0                            ;
                              51            60  9A 00187        MOVZBL  (R0), R1                               ;
                              09            51  91 0018A        CMPB    R1, #9                                 ;
                                            09  12 0018D        BNEQ    22$                                    ;
                  50    10    BE            07  C9 0018F        BISL3   #7, @16(SP), R0                        ;
                                            50  D6 00194        INCL    R0                                     ;
                                            2B  11 00196        BRB     26$                                    ;
                              0D            51  91 00198 22$:   CMPB    R1, #13                                ;
                                            0C  13 0019B        BEQL    23$                                    ;
                              08            51  91 0019D        CMPB    R1, #8                                 ;
                                            0B  12 001A0        BNEQ    24$                                    ;
                  50    10    BE            01  C3 001A2        SUBL3   #1, @16(SP), R0                        ;
                                            1A  18 001A7        BGEQ    26$                                    ;
                                            50  D4 001A9 23$:   CLRL    R0                                     ;
                                            16  11 001AB        BRB     26$                                    ;
                              20            51  91 001AD 24$:   CMPB    R1, #32                                ;
                                            0D  1F 001B0        BLSSU   25$                                    ;
                        7E    8F            51  91 001B2        CMPB    R1, #126                               ;
                                            07  1A 001B6        BGTRU   25$                                    ;
                  50    10    BE            01  C1 001B8        ADDL3   #1, @16(SP), R0                        ;
                                            04  11 001BD        BRB     26$                                    ;
                              50      10    BE  D0 001BF 25$:   MOVL    @16(SP), R0                            ;
                        10    BE            50  D0 001C3 26$:   MOVL    R0, @16(SP)                            ;
                                      1C    BE  D6 001C7        INCL    @28(SP)                                ;
                                      24    AE  B6 001CA        INCW    T_DSC                                  ;
                                            A8  11 001CD        BRB     20$                                    ;
                        0C    AE            04  D0 001CF 27$:   MOVL    #4, RET_VAL                            ; 2290
                                          00CD  31 001D3 28$:   BRW     43$                                    ; 2283
                              09            66  91 001D6 29$:   CMPB    (MOV_VAL), #9                          ; 2305
                                            09  12 001D9        BNEQ    30$                                    ;
                  50    10    BE            07  C9 001DB        BISL3   #7, @16(SP), R0                        ;
                                            50  D6 001E0        INCL    R0                                     ;
                                            19  11 001E2        BRB     33$                                    ;
                              0D            66  91 001E4 30$:   CMPB    (MOV_VAL), #13                         ; 2306
                                            0C  13 001E7        BEQL    31$                                    ;
                              08            66  91 001E9        CMPB    (MOV_VAL), #8                          ; 2307
                                            0B  12 001EC        BNEQ    32$                                    ;
                  50    10    BE            01  C3 001EE        SUBL3   #1, @16(SP), R0                        ;
                                            08  18 001F3        BGEQ    33$                                    ;
                                            50  D4 001F5 31$:   CLRL    R0                                     ;
                                            04  11 001F7        BRB     33$                                    ;
                              50      10    BE  D0 001F9 32$:   MOVL    @16(SP), R0                            ; 2308
                        10    BE            50  D0 001FD 33$:   MOVL    R0, @16(SP)                            ; 2303
                  50          57            52  C3 00201        SUBL3   R2, SRC_LENGTH, R0                     ; 2310
                              57      FF    A0  9E 00205        MOVAB   -1(R0), SRC_LENGTH                     ;
                        28    AE            52  C0 00209        ADDL2   R2, T_DSC+4                            ; 2311
                  50          58            52  C3 0020D        SUBL3   R2, DEST_LENGTH, R0                    ; 2312
                              58      FF    A0  9E 00211        MOVAB   -1(R0), DEST_LENGTH                    ;
                        20    AE      01    A6  9E 00215        MOVAB   1(R6), T_POS                           ; 2313
                        28    BE            66  90 0021A        MOVB    (MOV_VAL), @T_DSC+4                    ; 2319
                                      28    AE  D6 0021E        INCL    T_DSC+4                                ;
                  03    A1    AB            01  E1 00221        BBC     #1, -95(CCB), 35$                      ;
                                          FE18  31 00226 34$:   BRW     2$                                     ;
```

```
                10   BE        D4   AB          10          00  ED 00229 35$:    CMPZV   #0, #16, -44(CCB), @16(SP)
                                                             F4  18 00230          BGEQ    34$
                                          0C    AE          06  D0 00232          MOVL    #6, RET_VAL
                                          10    BE    04    AE  D0 00236          MOVL    L_SAV_PRINT_POS, @16(SP)
                                          1C    BE    08    AE  D0 0023B          MOVL    A_SAVE_BUF_PTR, @28(SP)
                                          24    AE          6E  B0 00240          MOVW    L_SAVE_LENGTH, T_DSC
                                    5A    A1    AB          04  E0 00244          BBS     #4, -95(CCB), 43$
                10   BE        D4   AB          10          00  ED 00249 36$:    CMPZV   #0, #16, -44(CCB), @16(SP)
                                                             4C  15 00250          BLEQ    42$
                                          50    1C    BE    D0 00252          MOVL    @28(SP), R0
                                          51          60    9A 00256          MOVZBL  (R0), R1
                                          09          51    91 00259          CMPB    R1, #9
                                                      09    12 0025C          BNEQ    37$
                                    50    10    BE          07  C9 0025E          BISL3   #7, @16(SP), R0
                                                             50  D6 00263          INCL    R0
                                                             2B  11 00265          BRB     41$
                                          0D          51    91 00267 37$:     CMPB    R1, #13
                                                      0C    13 0026A          BEQL    38$
                                          08          51    91 0026C          CMPB    R1, #8
                                                      0B    12 0026F          BNEQ    39$
                                    50    10    BE          01  C3 00271          SUBL3   #1, @16(SP), R0
                                                             1A  18 00276          BGEQ    41$
                                                             50  D4 00278 38$:     CLRL    R0
                                                             16  11 0027A          BRB     41$
                                          20          51    91 0027C 39$:     CMPB    R1, #32
                                                      0D    1F 0027F          BLSSU   40$
                                          7E    8F    51    91 00281          CMPB    R1, #126
                                                      07    1A 00285          BGTRU   40$
                                    50    10    BE          01  C1 00287          ADDL3   #1, @16(SP), R0
                                                             04  11 0028C          BRB     41$
                                          50    10    BE    D0 0028E 40$:     MOVL    @16(SP), R0
                                    10    BE          50    D0 00292 41$:     MOVL    R0, @16(SP)
                                                1C    BE    D6 00296          INCL    @28(SP)
                                                24    AE    B6 00299          INCW    T_DSC
                                                            AB  11 0029C          BRB     36$
                                    1C    BE    08    AE    D0 0029E 42$:     MOVL    A_SAVE_BUF_PTR, @28(SP)
                                    1C    BE    14    AE    D0 002A3 43$:     MOVL    ORIG_BUF_PTR, @28(SP)
                                    10    BC    24    AE    B0 002A8          MOVW    T_DSC, @RET_STR
                                          50    0C    AE    D0 002AD          MOVL    RET_VAL, R0
                                                            04 002B1          RET

; Routine Size:  690 bytes,    Routine Base:  _BAS$CODE + 06AB

; 1298          2335  1
```

```
;1300     2336  1 GLOBAL ROUTINE BAS$$DO_WRITE(                    . Write one record with indicated carriage control
;1301     2337  1     CARRIAGE_CTRL
;1302     2338  1         ) : JSB_DO_WRITE NOVALUE =
;1303     2339  1
;1304     2340  1 !++
;1305     2341  1 ! FUNCTIONAL DESCRIPTION:
;1306     2342  1 !
;1307     2343  1 !       This routine centralizes calls to the record level write routine.  A
;1308     2344  1 !       few vlues in the data base are reset.
;1309     2345  1 !
;1310     2346  1 ! FORMAL PARAMETERS:
;1311     2347  1 !
;1312     2348  1 !       CARRIAGE_CTRL.rlu.v     indicates whether to insert CRLF or leave the cursor
;1313     2349  1 !
;1314     2350  1 ! IMPLICIT INPUTS:
;1315     2351  1 !
;1316     2352  1 !       CCB                     address of current control block
;1317     2353  1 !
;1318     2354  1 ! IMPLICIT OUTPUTS:
;1319     2355  1 !
;1320     2356  1 !       LUB$V_OUTBUF_DR         indicates valid data in the output buffer
;1321     2357  1 !       LUB$L_PRINT_POS         current cursor position
;1322     2358  1 !
;1323     2359  1 ! COMPLETION CODES:
;1324     2360  1 !
;1325     2361  1 !       NONE
;1326     2362  1 !
;1327     2363  1 ! SIDE EFFECTS:
;1328     2364  1 !
;1329     2365  1 !       NONE
;1330     2366  1 !
;1331     2367  1 !--
;1332     2368  1
;1333     2369  2     BEGIN
;1334     2370  2
;1335     2371  2     EXTERNAL REGISTER
;1336     2372  2         CCB : REF BLOCK [, BYTE];
;1337     2373  2
;1338     2374  2     BAS$$REC_WSL1 (.CARRIAGE_CTRL);
;1339     2375  2
;1340     2376  2     !+
;1341     2377  2     ! Reset the output buffer dirty bit
;1342     2378  2     !-
;1343     2379  2 !+
;1344     2380  2 ! Reset the cursor position only if a record was written.  Otherwise, the
;1345     2381  2 ! cursor was left at the end of the data written and the cursor position
;1346     2382  2 ! is still valid
;1347     2383  2 !-
;1348     2384  2
;1349     2385  2     IF .CCB [LUB$A_BUF_PTR] EQL .CCB [LUB$A_BUF_BEG] THEN CCB [LUB$L_PRINT_POS] = 0;
;1350     2386  2     CCB [LUB$V_OUTBUF_DR] = 0;
;1351     2387  2     RETURN;
;1352     2388  1     END;
```

```
                        00000000G  00  16 00000 BAS$$DO_WRITE::
                                                   JSB     BAS$$REC_WSL1                          ; 2374
                    BC  AB      B0  AB  D1 00006       CMPL    -80(CCB), -68(CCB)                     ; 2385
                                    03  12 0000B       BNEQ    1$                                     ;
                                C8  AB  D4 0000D       CLRL    -56(CCB)                               ;
                    FE  AB          08  8A 00010 1$:   BICB2   #8, -2(CCB)                           ; 2386
                                        05 00014       RSB                                            ; 2388

; Routine Size:  21 bytes,    Routine Base:  _BAS$CODE + 095D

; 1353          2389  1
; 1354          2390  1 END
; 1355          2391  1
; 1356          2392  0 ELUDOM



;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _BAS$CODE                      2418  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



;                               Library Statistics
;
;                                        --------  Symbols  --------      Pages       Processing
;       File                             Total     Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1      9776        14         0        581          00:01.1



;                                       COMMAND QUALIFIERS

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

; Size:         2161 code + 257 data bytes
; Run Time:        00:59.4
; Elapsed Time:    02:12.2
; Lines/CPU Min:    2415
; Lexemes/CPU-Min: 26343
; Memory Used:  609 pages
; Compilation Complete
```

BASUDFWL
LIS

BASUNLOCK
LIS

BASVECTOR
LIS

BASVIRTUA
LIS

BASVAL
LIS

BASVIRTIO
LIS

BASUNWIND
LIS

BASUPDATE
LIS

BASVECTR2
LIS