M 1

```
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRR      TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRR      TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSSS  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  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB     AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB     AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

FILEID**BASUDFWL


```
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
```

```
0 MODULE BAS$$UDF_WL (                                  ! BASIC Write List Directed UDF
0               IDENT = '1-077'                         ! File: BASUDFWL.B32 Edit:MDL1077
0               ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 ! FACILITY:  BASIC Support Library - not user callable
1 !
1 ! ABSTRACT:
1 !
1 ! ENVIRONMENT:  User access mode; reentrant AST level or not.
1 !
1 ! AUTHOR:       Donald G. Petersen;     CREATION DATE: 17-Mar-78
1 !
1 ! MODIFIED BY:
1 !
1 ! 0-14  - Store error in ISB$B_ERR_NO, don't signal it! JMT 14-Jan-78
1 !         Donald G. Petersen, 17-Mar-78: Version 1-01
1 ! 1-01  -  original
1 ! 1-02  - Change to JSB linkage.  DGP 14-Nov-78
1 ! 1-004 - Update copyright notice and add device names to REQUIRE
1 !           files.  JBS 29-NOV-78
1 ! 1-005 - Change LUB$B_PRINT_SIZ to LUB$B_R_MARGIN.  DGP 05-Dec-78
1 ! 1-006 - Change REQUIRE file names from FOR... to OTS...  JBS 07-DEC-78
1 ! 1-007 - Fix some failures to return values (new BLISS compiler) and
1 !           run through PRETTY to improve appearance.  JBS 20-DEC-78
1 ! 1-008 - Add a longword entry point to WL1.  DGP 27-Dec-78
1 ! 1-009 - Change one argument in call to BAS$CNV_OUT_G. DGP 03-Jan-78
1 ! 1-010 - Change references to ISB$A_BUF_BEG, BUF_END, and BUF_PTR to LUB.
1 !         DGP 05-Jan-79
1 ! 1-011 - Change some constants to symbolics.  DGP 10-Jan-79
1 ! 1-012 - Convert to CR format for data files.  DGP 11-Jan-79
1 ! 1-013 - Fix a bug in Print string.  DGP 16-Jan-79
1 ! 1-014 - Add a guard bit for AST reentrancy.  DGP 20-Jan-79
```

```
! 1-015 - Correct a typo in edit 014.  JBS 20-JAN-1979
! 1-016 - Change DO_WRITE to a Global routine BAS$$DO_WRITE so that it can
!         be called by another module which does I/O cleanup at the end of a
!         program.  DGP 22-Jan-79
! 1-017 - Use 32-bit addresses for externals.  JBS 27-JAN-1979
! 1-018 - Change buffer overflow handling in BAS$$UDF_WL1.  DGP 22-Feb-79
! 1-019 - Make change to handling of strings longer than output buffer.  DGP
!         27-Feb-79
! 1-020 - Make LUB$B_PRINT_POS a longword.  DGP 19-Mar-79
! 1-021 - Fix a few bugs in PRINT.  DGP 19-Mar-79
! 1-022 - Make another attempt to get comma formatting working.  DGP 02-Apr-79
! 1-023 - Semicolon formatting didn't check for overflow of right margin.
!         DGP 03-Apr-79
! 1-024 - SCAN_TEXT not working properly for non-printing characters.  DGP
!         10-Apr-79
! 1-025 - Fix an obscure error in SCAN_TEXT.  DGP 12-Apr-79
! 1-026 - Change macro CHK_CURSOR_POS to GTR from GTRU.  DGP 16-Apr-79
! 1-027 - TAB is always at least one space when updating printhead position.
!         DGP 16-Apr-79
! 1-028 - Don't allow print_pos to go less than 0 for backspace characters.
!         DGP 16-Apr-79
! 1-029 - Add support for CRLF delimiting a record.  DGP 25-Apr-79
! 1-030 - Remove Global attribute from BAS$$DO_WRITE.  DGP 10-May-79
! 1-031 - Change ISB$V_AST_GUARD to LUB$V_AST_GUARD.  JBS 15-MAY-1979
! 1-032 - Make BAS$$DO_WRITE global again for BASUDFWF.  DGP 15-May-79
! 1-033 - Make the margin 16 bits.  JBS 30-MAY-1979
! 1-034 - BAS$$UDF_WL0 now initializes LUB$V_FORM_CHAR to 0.  DGP 12-Jun-79
! 1-035 - Update LUB$A_BUF_PTR in case where string is too long to fit in
!         buffer.  DGP 20-Jun-79
! 1-036 - Update LUB$A_BUF_PTR correctly if string is too long.  DGP 26-Jun-79
! 1-037 - Use the new conversion routines.  DGP 28-Jun-79
! 1-038 - Check V_NOMARGIN and allow infinite margin if set.  DGP 11-Jul-79
! 1-039 - Check for buffer overflow as well as right margin.  DGP 15-Jul-79
! 1-040 - Allow Mat Print to jump to REC level from UDF_WL1.  DGP 08-Aug-79
! 1-041 - Remove the BUILTIN ACTUALCOUNT.  DGP 04-Sep-79
! 1-042 - PRINT 10 digits for longword integer.  DGP 07-Sep-79
! 1-043 - Debug CRLF delimiting a record.  DGP 23-Sep-79
! 1-044 - Clear LUB$L_PRINT_POS for Mat Print if no format character.  DGP
!         02-Oct-79
! 1-045 - Clear ISB$V_MAT_PRINT.  DGP 05-Oct-79
! 1-046 - Make UDF9 dispatch to the REC level.  DGP  12-Oct-79
! 1-047 - Try again to get 'CRLF' to delimit a record.  DGP 15-Oct-79
! 1-048 - Fix bug in printing long strings.  The pointer was not moving through
!         the string.  Also, leave the cursor where it lies after each little
!         piece of the long string.  DGP 01-Nov-79
! 1-049 - Pass one arg to BAS$$DO_WRITE to indicate whether to put out
!         End-of-record.  DGP 06-Nov-79
! 1-050 - Last edit screwed the proper resetting of LUB$L_PRINT_POS in
!         BAS$$DO_WRITE; fix same.  DGP 08-Nov-79
! 1-051 - Update columns 10-15 in the translation table in SCAN_TEXT.  DGP
!         09-Nov-79
! 1-052 - Don't update LUB$A_BUF_PTR in CHK_CURSOR_POS macro.  DGP  09-Nov-79
! 1-053 - Pick up the scale factor from the ISB.  DGP 15-Nov-79
! 1-054 - If 'CRLF' is detected to break a record and they are the last two
!         characters in the record, then don't put out a following null record.
!         DGP 03-Jan-80
! 1-055 - Set ISB$V_PRINT_INI in WL0 and clear it if WL1 is called.  DGP 04-Jan-80
```

```
1 ! 1-056 - Unconditionally initialize LUB$V_OUTBUF_DR to 1 in WL1.  DGP 07-Jan-80
1 ! 1-057 - All Calls to BAS$$DO_WRITE write an entire record.  DGP 14-Jan-80
1 ! 1-058 - Bug fix to PRINT with semicolon formatting that exceeds the left
1 !         margin.  DGP 22-Dec-80
1 ! 1-059 - PRINTing a record to the terminal longer than the buffer size inserts
1 !         an extra CRLF.  DGP 13-Feb-80
1 ! 1-060 - The macro CHK_CURSOR_POS did not count LF as one character position.
1 !         It does now.  DGP 18-Feb-80
1 ! 1-061 - Correction to edit 1-060.  LF should never count as a character
1 !         position.  DGP  19-Feb-80
1 ! 1-062 - The macro CHK_CURSOR_POS gets in infinite loop when it is trying
1 !         to decide at what point the margin was exceeded.  Correct this.
1 !         DGP 27-FEB-80
1 ! 1-063 - Edit 1-062 introduced another problem with the buffer pointer.
1 !         Fix this.
1 ! 1-064 - The case of exceeding the right margin for terminal output with the
1 !         first element of a Print statement which follows a Print statement that
1 !         ended in a semicolon or comma causes the element to be printed twice.
1 !         Fixed. DGP 01-Mar-80
1 ! 1-065 - Do not unconditionally reset LUB$V_FORM_CHAR in WLO.  This is a
1 !         belated finish to edit 55 to fix a bug.  DGP 07-Mar-80.
1 ! 1-066 - In SCAN_TEXT, when checking for CRLF and last character encountered
1 !         in the buffer is a CR, don't look beyond the buffer for the LF.
1 !         REJ 05-May-80
1 ! 1-067 - Put in the fix for checking last print zone properly.
1 !         L_ELEM_SIZE is set to zero in MACRO CHK_PRINT_ZONE, and the value
1 !         of BAS$K_COND_SUC in BASPAR.MAR is set to 2 in place of 3. FM 19-SEP-80
1 ! 1-068 - Put in the fix for throwing out the next integer or floating number
1 !         when the buffer is full.  FM 25-SEP-80
1 !
1 ! 1-069 - The fix for 1-065A, did not quite fix the problem, so try again.
1 !         FM JAN-81.
1 ! 1-070 - Lines with tabs were not split properly if the line exceeded
1 !         the margin.  Fix CHK_CURSOR_POS and SCAN_TEXT to count
1 !         tab print positions properly, and SCAN_TEXT now updates
1 !         the buffer pointer to keep pace with the print position.
1 !         PL 30-Jun-81.
1 ! 1-071 - Add support for G & H.  PL 21-Aug-81
1 ! 1-072 - Add support for packed decimal.  PL 5-Oct-81
1 ! 1-073 - More edits for packed decimal.  PL 6-Jan-82
1 ! 1-074 - BAS$$UDF_WL1 should call BAS$CVT_OUT_P_G to format packed strings,
1 !         instead of BAS$CVT_P_T.  PLL 28-Jan-82
1 ! 1-075 - Fix a bug in packed = BAS$CVT_OUT_P_G may deallocate and
1 !         reallocate the dynamic output string.  PLL 4-Mar-82
1 ! 1-076 - in BAS$$UDF_WLO, load BUF_PTR from RBUF_ADR.  This fixes a re-entrancy
1 !         problem.  MDL 10-May-1983
1 ! 1-077 - don't allow string elements to be split across lines when /ANSI.
1 !         modify CHK_CURSOR_POS accordingly.  MDL 5-Jul-1983
1 !--
1
1 !<BLF/PAGE>
```

```
!
! SWITCHES:
!

SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);

!
! LINKAGES
!

REQUIRE 'RTLIN:OTSLNK';                          ! define all linkages

!
! TABLE OF CONTENTS:
!

FORWARD ROUTINE
    BAS$$UDF_WL0 : JSB_UDF0 NOVALUE,             ! initialization
    BAS$$UDF_WL1 : CALL_CCB NOVALUE,             ! format one user I/O list element
    BAS$$UDF_WL9 : JSB_UDF9 NOVALUE,             ! end of user I/O list - finish
    SCAN_TEXT : CALL_CCB,                        ! Scan text string for special characters
    BAS$$DO_WRITE : JSB_DO_WRITE NOVALUE;        ! Output routine

BUILTIN
    CVTPS,                                       ! Escape to MARS CVTPS instructions
    CVTLD,                                       ! Escape to MARS CVTLD instruction
    MOVTUC;                                      ! Escape to MARS MOVTUC instruction

!
! INCLUDE FILES:
!

REQUIRE 'RTLML:BASPAR';                          ! Intermodule BASIC parameters and constants

REQUIRE 'RTLML:OTSISB';                          ! I/O statement block (ISB) offsets

REQUIRE 'RTLML:OTSLUB';                          ! Only needed to get LUB length!

REQUIRE 'RTLIN:OTSMAC';                          ! Macros

REQUIRE 'RTLIN:RTLPSECT';                        ! Define DECLARE_PSECTS macro

LIBRARY 'RTLSTARLE';                             ! STARLET library for macros and symbols

!
! EQUATED SYMBOLS:
!

LITERAL
    K_FIELD_SIZE = 14;                           ! print zone size

!
! MACROS:
!

MACRO
```

```
!+
! If the format character is a comma, see if a full print zone remains
! in the record.  If a full print zone is not available, then return a
! zero; else return a one.
! Note: this macro is only executed if the conversion or scan routine
! returned a success status.
!-

CHK_PRINT_ZONE =
        CCB[LUB$A_BUF_PTR] = .CCB[LUB$A_BUF_PTR] + .DSC[DSC$W_LENGTH];
        IF .FORMAT_CHAR EQL BAS$K_COMMA_FOR
        THEN
            BEGIN
            CCB[LUB$A_BUF_PTR] = CH$FILL(' ',
                (IF .CCB[LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR] GTRU K_FIELD_SIZE
                    THEN K_FIELD_SIZE - (.CCB [LUB$L_PRINT_POS] MOD K_FIELD_SIZE)
                    ELSE .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR]),
                .CCB[LUB$A_BUF_PTR]);
            CCB[LUB$L_PRINT_POS] = .CCB[LUB$L_PRINT_POS] + K_FIELD_SIZE -
                    (.CCB[LUB$L_PRINT_POS] MOD K_FIELD_SIZE);
            END;


        !+
        ! Check for a format character and that the right margin isn't
        ! infinite.
        !-

        IF .FORMAT_CHAR EQL BAS$K_COMMA_FOR
         OR .FORMAT_CHAR EQL BAS$K_SEMI_FORM
        THEN
            BEGIN
            IF ((.CCB[LUB$L_PRINT_POS] LEQU .CCB[LUB$W_R_MARGIN]) OR .CCB [LUB$V_NOMARGIN])
            AND (.CCB[LUB$A_BUF_PTR] LEQU .CCB[LUB$A_BUF_END])
            THEN

                !+
                ! If this is a comma format, check to see if there is a full
                ! print zone left on the line.  If this is a semi-colon, this
                ! check is redundant, but the code is applicable so what's a
                ! little redundancy?
                ! If it does then return success.  If it doesn't, return failure
                ! without resetting the pointers so that this item will be
                ! written.
                !-

                BEGIN
                L_ELEM_SIZE = 0;
                IF ((.CCB [LUB$L_PRINT_POS] +
                    (IF .FORMAT_CHAR EQL BAS$K_COMMA_FOR THEN K_FIELD_SIZE
                                                            ELSE 0)
                LEQU .CCB [LUB$W_R_MARGIN]) OR .CCB [LUB$V_NOMARGIN])
                THEN
                    BAS$K_SUCCESS
                ELSE
                    BAS$K_COND_SUC
                END
```

```
                    ELSE

                        !+
                        ! The comma causes this field to overflow the right margin
                        ! Check to see if there is anything else in the buffer. If
                        ! there is, reset the pointers to before this element;  if
                        ! nothing else in the buffer, return a success status
                        ! Or, a semicolon format caused the field to overflow past
                        ! the buffer.  Save this one for the next record.
                        ! Check for the cursor position not being 0 prior to this
                        ! element 'cuz there could have been a series of statements
                        ! ending in comma or semi and if this is a terminal device,
                        ! then the data is PUT at each IO_END and the buffer doesn't
                        ! appear dirty although the print line is.
                        !-

                        IF .CCB[LUB$V_OUTBUF_DR] OR (.SAV_PRINT_POS NEQ 0)
                        THEN
                            BEGIN
                            DSC[DSC$W_LENGTH] = 0;
                            CCB[LUB$A_BUF_PTR] = .A_SAV_PTR;
!+
! If the buffer was overflowed, then leave the cursor where it was; if the margin
! overflowed, then start a new record.
!-
                            IF (NOT .CCB [LUB$V_NOMARGIN]) AND (.CCB [LUB$L_PRINT_POS] GTRU .CCB [LUB$W_R_MARGIN
                            THEN
                                BAS$K_MAR_EXC
                            ELSE
                                BAS$K_BUF_EXC
                            END
                        ELSE
                            BEGIN
                            CCB[LUB$A_BUF_PTR] = .CCB[LUB$A_BUF_BEG] + .DSC[DSC$W_LENGTH];
                            FORMAT_CHAR = BAS$K_SEMI_FORM;
                            L_ELEM_SIZE = 0;
!+
! If the buffer was overflowed, then leave the cursor where it was; if the margin
! overflowed, then start a new record.
!-
                            IF (NOT .CCB [LUB$V_NOMARGIN]) AND (.CCB [LUB$L_PRINT_POS] GTRU .CCB [LUB$W_R_MARGIN
                            THEN
                                BAS$K_MAR_EXC
                            ELSE
                                BAS$K_BUF_EXC
                            END
                    END
                ELSE
                    1
%;                                                          !end of macro

MACRO
    ! check the present cursor position against the desired print size.
    ! Due to the possibility of having carriage control characters in
    ! the output the data stream, the cursor position is used to check
    ! the end of the record rather the number of characters in the record.
```

```
	!-

	CHK_CURSOR_POS =
		IF (NOT .CCB [LUB$V_NOMARGIN]) AND (.CCB[LUB$L_PRINT_POS] GTR .CCB[LUB$W_R_MARGIN])
		THEN
		    !+
		    ! The cursor position has exceeded the print size.
		    ! Restore the pointers back to where they were before the
		    ! last MOVTUC.
		    !-

		    BEGIN
		    RET_VAL = BAS$K_MAR_EXC;

		    !+
		    ! Reset a few pointers (use the fact that the data is al-
		    ! ready in the destination field) and find out the length
		    ! of the string which fits within the right margin.  This
		    ! requires looking at each character individually.
		    !-

		    CCB[LUB$L_PRINT_POS] = .L_SAV_PRINT_POS;
		    CCB[LUB$A_BUF_PTR] = .A_SAVE_BUF_PTR;
		    T_DSC[DSC$W_LENGTH] = .L_SAVE_LENGTH;

		    !+
		    ! if ANSI, then get out - we don't split string elements
		    ! across lines.
		    !-
		    IF .CCB [LUB$V_ANSI] THEN EXITLOOP;

		    WHILE .CCB[LUB$L_PRINT_POS] LSS .CCB[LUB$W_R_MARGIN]
		    DO
			BEGIN
			CCB[LUB$L_PRINT_POS] = (SELECTONE .(.CCB[LUB$A_BUF_PTR])<0,8,0> OF
			    SET
			    [K_TAB]: (.CCB[LUB$L_PRINT_POS] OR K_TAB_LIT) + 1;
			    [K_CR]: 0;
			    [K_BKSP]: MAX(0, .CCB[LUB$L_PRINT_POS] - 1);
			    [K_BLANK TO K_TILDE]: .CCB[LUB$L_PRINT_POS] + 1;
			    [OTHERWISE]: .CCB[LUB$L_PRINT_POS] + 0;
			    TES);
			CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_BUF_PTR] + 1;
			T_DSC[DSC$W_LENGTH] = .T_DSC[DSC$W_LENGTH] + 1;
			END;		! While loop
!+
! Restore the buffer pointer so that it can be incremented properly by the
! code that does the common output stuff.
!-
		    CCB [LUB$A_BUF_PTR] = .A_SAVE_BUF_PTR;
		    EXITLOOP;
		    END;
%;

!
! PSECT DECLARATIONS:
!
```

```
1 DECLARE_PSECTS (BAS);                                   ! declare PSECTs for BAS$ facility
1 !
1 ! OWN STORAGE:
1 !
1 !     None
1 !
1 ! EXTERNAL REFERENCES:
1 !
1 EXTERNAL
1 !+
1 ! Array of REC9 routines
1 !-
1     BAS$$AA_REC_PR9 : VECTOR;
1
1 EXTERNAL ROUTINE
1 !     The following are general library routines available for any
1 !     one's use (value is true if fits in field):
1     BAS$CVT_OUT_H_G,                                     ! Convert H float to G format
1     BAS$CVT_OUT_H_E,                                     ! Convert H float to E format
1     BAS$CVT_OUT_G_G,                                     ! Convert G float to E or F
1     BAS$CVT_OUT_D_G,                                     ! Basic G (E or F) output conversion
1     BAS$CVT_OUT_P_G,                                     ! Convert packed to G format
1     BAS$$REC_WSL0 : JSB_REC0 NOVALUE,                    ! initialize list directed output
1     BAS$$REC_WSL1 : JSB_REC_WSL1 NOVALUE,                ! write list directed
1     BAS$$REC_WSL9 : JSB_REC9 NOVALUE,                    ! end list directed output
1     BAS$$REC_MPR1 : JSB_REC1 NOVALUE,                    ! Mat Print REC level
1     STR$APPEND,                                          ! append a string
1     STR$DUPL_CHAR,                                       ! create a string of specified char
1     STR$FREE1_DX,                                        ! deallocate dynamic string
1     LIB$STOP : NOVALUE;                                  ! signal an error
1
```

```
GLOBAL ROUTINE BAS$$UDF_WL0                     ! Write list directed UDF initialization
    : JSB_UDF0 NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!       Initialize PRINT User data formatter (UDF)
!
! FORMAL PARAMETERS:
!
!       NONE
!
! IMPLICIT INPUTS:
!
!       LUB$V_AST_GUARD         Guard bit for AST reentrancy
!       LUB$A_BUF_PTR           Pointer to next byte in user buffer
!
! IMPLICIT OUTPUTS:
!
!       LUB$V_AST_GUARD         Guard bit for AST reentrancy
!       LUB$A_BUF_BEG           Pointer to first byte of user buffer
!       LUB$A_BUF_PTR           Adr of next byte of output
!                               data buffer
!       LUB$A_BUF_END           Adr of end of data buffer
!       ISB$V_PRINT_INI         indicates that a PRINT has been initialized
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    EXTERNAL REGISTER
        CCB : REF BLOCK [, BYTE];

    !+
    ! A guard bit is used to ensure AST reentrancy.  The bit is set to 1
    ! at the top of the routine, tested for 1 at the bottom of the routine,
    ! and set to 0 upon exitting.  If the test for 1 fails at the bottom of
    ! the routine, then an AST has gone off and used this routine possibly
    ! changing the buffer pointers. Therefore this routine will loop back and
    ! run itself again in its entirety.
    !-

    DO
        BEGIN

        !+
        ! Set the guard bit
        !-
```

```
                CCB [LUB$V_AST_GUARD] = 1;

                !+
                ! Call record level to get buffer pointers.
                !-

                BAS$$REC_WSL0 ();

                !+
                ! set the beginning of the buffer if there is no format character pending
                !-

                IF NOT .CCB [LUB$V_FORM_CHAR]
                  THEN
                      BEGIN
                        CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_RBUF_ADR];
                        CCB [LUB$A_BUF_BEG] = .CCB [LUB$A_BUF_PTR];
                      END;

                !+
                ! Set the print initialized bit.  The null print line (10       PRINT)
                ! does not have an element transmitter (compiler optimization).  So,
                ! the bit is set here and then can be cleared by element transmitters
                ! if any.
                !-

                CCB [ISB$V_PRINT_INI] = 1;

                !+
                ! Check the guard bit.  If it is now 0, then an AST has gone thru this routine
                ! Since the data base may have been altered in an unpredictable manner, it
                ! is necessary to redo the entire routine.  Note: in worst case processing,
                ! the run-time for this routine is essentially unbounded.
                !-

                END
            UNTIL .CCB [LUB$V_AST_GUARD];                    ! End of AST guard loop

            CCB [LUB$V_AST_GUARD] = 0;
            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_WSL0, 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
```

```
GLOBAL ROUTINE BAS$$UDF_WL1 (ELEM_TYPE, ELEM_SIZE, ELEM_ADR, FORMAT_CHAR         ! format character
    ) : CALL_CCB NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!       Write list-directed User Data Formatter.
!       Accept an I/O element, format it, and put it in the record buffer.
!       Calls record level processors to perform the actual I/O if the buffer
!       is full or if non-forcible and end-of-record (no format character).
!
! FORMAL PARAMETERS:
!
!       ELEM_TYPE.rlu.v         data type of the element
!       ELEM_SIZE.rlu.v         size of the data element
!       ELEM_ADR.rlu.r          adr of the data element to be written
!                               Points to a descriptor for strings
!       FORMAT_CHAR.rlu.v               type of format character which followed the data element
!
! IMPLICIT INPUTS:
!
!       LUB$V_AST_GUARD         guard bit for AST reentrancy
!       LUB$L_PRINT_POS         current cursor position
!       LUB$V_OUTBUF_DR         indicates valid data in the output buffer.
!       LUB$W_R_MARGIN          size of buffer specified in OPEN statement.
!       LUB$V_FORM_CHAR         flag that a format character (',' or ';') was
!                               seen on the last element.
!       LUB$A_BUF_BEG           pointer to beginning of user buffer
!       LUB$A_BUF_PTR           pointer to current position in the buffer.
!       LUB$A_BUF_END           pointer to last byte of buffer + 1.
!
! IMPLICIT OUTPUTS:
!
!       ISB$V_PRINT_INI         reset print intialize flag - there is at least one
!                               element transmitter
!       LUB$V_AST_GUARD         guard bit for AST reentrancy
!       LUB$V_OUTBUF_DR         indicates valid data in output buffer
!       LUB$V_FORM_CHAR         flag to indicate a format character was seen
!       LUB$L_PRINT_POS         internal cursor position.
!       LUB$A_BUF_PTR           next byte in the user buffer
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       If an AST goes off while we are in this routine and calls this routine,
!       then this routine will be repeated.  It will continue to be repeated
!       until there are no more ASTs using this routine.
!
!--

    BEGIN

    EXTERNAL REGISTER
```

```
        CCB : REF BLOCK [, BYTE];

    MAP
        ELEM_ADR : REF VECTOR;                          ! element is call-by-reference

    LOCAL
        STR_PTR,                                        ! pointer into string being printed
        L_ELEM_SIZE,                                    ! temp for ELEM_SIZE for strings
                                                        ! in case a string is longer than the buffer
        A_SAV_PTR,                                      ! temp for saving LUB$A_BUF_PTR
        P,                                              ! temporary character string pointer
        DIFF,                                           ! number of bytes left in record buffer
        DSC : BLOCK [8, BYTE],                          ! static string descriptor for output field
        TEMP_DEC_DSC : BLOCK [8,BYTE];                  ! place to store converted decimal
                                                        ! string
    LITERAL
        K_FIELD_SIZE = 14,                              ! Maximum size to be used by output conversion routine
        K_TEMP_BUF_SIZ = 14;

    !+
    ! This loop is to ensure AST reentrancy.
    !-

    DO
        BEGIN
        CCB [LUB$V_AST_GUARD] = 1;
        STR_PTR = 0;
!+
! Reset the print initialize flag; there is at least one element transmitter.
!-
        CCB [ISB$V_PRINT_INI] = 0;
        L_ELEM_SIZE = .ELEM_SIZE;

        !+
        !   convert or move the field directly into the output buffer with
        !  the exception of the numerics using G format.
        !-

        DSC [DSC$B_DTYPE] = 0;
        DSC [DSC$B_CLASS] = 0;
        DSC [DSC$W_LENGTH] = 0;

        !+
        ! save the current pointer in the buffer
        ! Only needed the first time thru the loop
        !-

        A_SAV_PTR = .CCB [LUB$A_BUF_PTR];

        !+
        ! determine the length of each field taking into account the format character.
        ! In addition to the length, condition the bit which indicates the
        ! presence or absence of a format character.
        !-

        CASE .FORMAT_CHAR FROM BAS$K_SEMI_FORM TO BAS$K_NO_FORM OF
            SET
```

```
          [BAS$K_SEMI_FORM] :
              CCB [LUB$V_FORM_CHAR] = 1;

          [BAS$K_COMMA_FOR] :
              CCB [LUB$V_FORM_CHAR] = 1;

          [BAS$K_NO_FORM] :
              CCB [LUB$V_FORM_CHAR] = 0;
          TES;


      IF .ELEM_TYPE EQL DSC$K_DTYPE_P THEN
      BEGIN

      !+
      ! Convert packed decimal string to text so that it can be handled
      ! in the same way as text.  Can't convert it in it's own space so
      ! copy to a temporary area.
      !-

      LOCAL
          STATUS;

      TEMP_DEC_DSC [DSC$B_CLASS] = DSC$K_CLASS_D;
      TEMP_DEC_DSC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
      TEMP_DEC_DSC [DSC$A_POINTER] = 0;
      TEMP_DEC_DSC [DSC$W_LENGTH] = 0;

      STATUS = (BAS$CVT_OUT_P_G (.ELEM_ADR, 0, L_ELEM_SIZE, TEMP_DEC_DSC));
      IF NOT .STATUS THEN LIB$STOP (.STATUS);

      END;                                        ! end of packed conversion

      !+
      ! This WHILE loop will repeat as often as necessary to put out the entire
      ! field.  If a field will not fit into the buffer currently it is
      ! important to note whether there is anything in the buffer to begin with.
      ! If there is then the buffer pointer is set to the value it had when
      ! this routine was entered.  Then the buffer is dumped.  If there is nothing
      ! in the buffer, then the assumption is that this is a string that is too
      ! long to fit into the buffer and must be written in sections.  The string
      ! is left in the buffer and the buffer is dumped.  The loop is then repeated
      ! until the whole string has been dumped.  The key to control is the flag
      ! LUB$OUTBUF_DR.
      !-

      WHILE 1 DO
          BEGIN

          LOCAL
              WRITE_STATUS,                       ! status returned from the conversion routines
              SAV_PRINT_POS;                      ! Temporary to save the current cursor position

          SAV_PRINT_POS = .CCB [LUB$L_PRINT_POS];
          DSC [DSC$A_POINTER] = .CCB [LUB$A_BUF_PTR];
```

```
!+
! Perform the appropriate conversions.
!-

IF NOT (WRITE_STATUS = (CASE .ELEM_TYPE FROM DSC$K_DTYPE_B TO DSC$K_DTYPE_H OF
        SET
        [DSC$K_DTYPE_B,DSC$K_DTYPE_W, DSC$K_DTYPE_L] :

            !+
            ! Type integer - word or longword.  Print out as integer
            ! Always print as F format.
            !-

            BEGIN

            LITERAL
                K_NUM_DIGITS = 10;
            LOCAL
                RET_LENGTH,             ! length returned by conversion routine
                D_VALUE : VECTOR [2];           ! holds double precision floating value

            DSC [DSC$W_LENGTH] = (IF .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR] GTRU
                K_FIELD_SIZE

                !+
                ! Convert the integer value to a double precision value in
                ! preparation for calling BAS$CVT_OUT_D_G
                !-

            THEN K_FIELD_SIZE ELSE .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR]);
            CVTLD (ELEM_ADR [0], D_VALUE [0]);

            IF BAS$CVT_OUT_D_G (D_VALUE, 0, RET_LENGTH, DSC, 0, K_NUM_DIGITS)
            THEN
                BEGIN
                L_ELEM_SIZE = 0;
                DSC [DSC$W_LENGTH] = .RET_LENGTH;
                CCB [LUB$L_PRINT_POS] = .CCB [LUB$L_PRINT_POS] + .DSC [DSC$W_LENGTH];
                CHK_PRINT_ZONE
                END
            ELSE
                BAS$K_BUF_EXC

            END;
        [DSC$K_DTYPE_F, DSC$K_DTYPE_D] :
            BEGIN

            LOCAL
                RET_LENGTH,             ! hold the length returned by conversion routine
                D_VALUE : VECTOR [2];           ! holds double precision floating value

            DSC [DSC$W_LENGTH] = (IF .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR] GTRU
                K_FIELD_SIZE THEN K_FIELD_SIZE ELSE .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR])
            D_VALUE [0] = .ELEM_ADR [0];
            D_VALUE [1] = (IF .ELEM_SIZE EQL %UPVAL THEN 0 ELSE .ELEM_ADR [1]);

            IF BAS$CVT_OUT_D_G (D_VALUE, 0, RET_LENGTH, DSC,
```

```
7                   (IF .ELEM_SIZE EQL %UPVAL THEN 0 ELSE .CCB [ISB$B_SCALE_FAC]))
7               THEN
8                   BEGIN
8                   L_ELEM_SIZE = 0;
8                   DSC [DSC$W_LENGTH] = .RET_LENGTH;
8                   CCB [LUB$L_PRINT_POS] = .CCB [LUB$L_PRINT_POS] + .DSC [DSC$W_LENGTH];
8                   CHK_PRINT_ZONE
8                   END
7               ELSE
7                   BAS$K_BUF_EXC
7
6               END;
6           [DSC$K_DTYPE_G] :
7               BEGIN
7
7               LOCAL
7                   RET_LENGTH,                     !length retd by conversion routine
7                   G_VALUE : VECTOR [2];           !holds G floating value
7
8               DSC [DSC$W_LENGTH] = (IF .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR] GTRU
7                   K_FIELD_SIZE THEN K_FIELD_SIZE ELSE .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR])
7               G_VALUE [0] = .ELEM_ADR [0];
7               G_VALUE [1] = .ELEM_ADR [1];
7
7               IF BAS$CVT_OUT_G_G (G_VALUE, 0, RET_LENGTH,DSC)
7               THEN
8                   BEGIN
8                   L_ELEM_SIZE = 0;
8                   DSC [DSC$W_LENGTH] = .RET_LENGTH;
8                   CCB [LUB$L_PRINT_POS] = .CCB [LUB$L_PRINT_POS] + .DSC [DSC$W_LENGTH];
8                   CHK_PRINT_ZONE
8                   END
7               ELSE
7                   BAS$K_BUF_EXC
6               END;
6           [DSC$K_DTYPE_H] :
7               BEGIN
7
7               LOCAL
7                   RET_LENGTH,                     !length retd by conversion routine
7                   H_VALUE : VECTOR [4];           !holds H floating value
7
8               DSC [DSC$W_LENGTH] = (IF .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR] GTRU
7                   K_FIELD_SIZE THEN K_FIELD_SIZE ELSE .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR])
7               H_VALUE [0] = .ELEM_ADR [0];
7               H_VALUE [1] = .ELEM_ADR [1];
7               H_VALUE [2] = .ELEM_ADR [2];
7               H_VALUE [3] = .ELEM_ADR [3];
7
7               IF BAS$CVT_OUT_H_G (H_VALUE, 0, RET_LENGTH, DSC)
7               THEN
8                   BEGIN
8                   L_ELEM_SIZE = 0;
8                   DSC [DSC$W_LENGTH] = .RET_LENGTH;
8                   CCB [LUB$L_PRINT_POS] = .CCB [LUB$L_PRINT_POS] + .DSC [DSC$W_LENGTH];
8                   CHK_PRINT_ZONE
8                   END
```

```
7                     ELSE
7                         BAS$K_BUF_EXC
6                     END;
6                 [DSC$K_DTYPE_T, DSC$K_DTYPE_P] :
6
6                     !+
6                     ! text or packed decimal
6                     !-
6
7                     BEGIN
7                     LOCAL
7                         SRC_LOC,                        ! source text location
7                         SCAN_STATUS;                    ! status returned by SCAN_TEXT
7
7                     !+
7                     ! Packed decimal has been converted to text and stored in
7                     ! a temporary descriptor.  SCAN_TEXT must move from this
7                     ! temporary area to the destination descriptor, instead of
7                     ! moving from ELEM_ADR.
7                     !-
7
7                     IF .ELEM_TYPE EQL DSC$K_DTYPE_P
7                     THEN
7                         SRC_LOC = .TEMP_DEC_DSC [DSC$A_POINTER]
7                     ELSE
7                         SRC_LOC = .ELEM_ADR [1];
7
7                     IF SCAN_STATUS = SCAN_TEXT ((.SRC_LOC + .STR_PTR), .L_ELEM_SIZE,
7                             .CCB [LUB$A_BUF_END] - .CCB [LUB$A_BUF_PTR], DSC) THEN
7
7                     !+
7                     ! The field fits into the buffer, now adjust for a
7                     ! comma format character and see if the field still
7                     ! fits.
7                     ! \ Will this overflow the buffer and stay within the
7                     ! right margin?\
7                     !-
7
8                         BEGIN
8                         CHK_PRINT_ZONE
8                         END
7                     ELSE
8                         BEGIN
8
8                         !+
8                         ! If LUB$V_OUTBUF_DR is true then there is other data
8                         ! in the buffer.
8                         ! If SCAN_STATUS is not BAS$K_CRLF (CRLF detected in record)
8                         ! The buffer pointer should be reset
8                         ! and the other data should be dumped.  If the bit is
8                         ! false, then this is a big string which won't all
8                         ! fit in the buffer at once.  Dump what is there and
8                         ! come back for more.
8                         ! If SCAN_STATUS is BAS$K_CRLF then a CRLF is in the
8                         ! string and it will fit with the other data here.
8                         ! Pretend it is all one big string.
8                         !-
```

```
;  8
;  9                              IF .CCB [LUB$V_OUTBUF_DR] AND (.SCAN_STATUS NEQ BAS$K_CRLF)
;  8                              THEN
;  9                                  BEGIN
;  9
;  9                                  !+
;  9                                  ! Set the length back to zero so that the entire
;  9                                  ! field will be scanned again
;  9                                  !-
;  9
;  9                                  DSC [DSC$W_LENGTH] = 0;
;  9                                  CCB [LUB$A_BUF_PTR] = .A_SAV_PTR;
;  9                                  END
;  8                              ELSE
;  8
;  8                              !+
;  8                              ! Decrement the length of this string which is
;  8                              ! longer than one record for the next iteration
;  8                              !-
;  8
;  9                                  BEGIN
;  9                                  STR_PTR = .STR_PTR + .DSC [DSC$W_LENGTH];
;  9                                  L_ELEM_SIZE = .L_ELEM_SIZE - .DSC [DSC$W_LENGTH];
;  9                                  CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_BUF_PTR] + .DSC [DSC$W_LENGTH]
;  9                                      - (IF .SCAN_STATUS EQL BAS$K_CRLF THEN 2 ELSE 0);
;  8                                  END;
;  8
;  8                              .SCAN_STATUS
;  8                              END
;  6                          END;
;  6
;  6
;  6                  [INRANGE, OUTRANGE] : 0      ! this can not happen
;  5                  TES))
;  4          THEN
;  4
;  4          !+
;  4          ! This element won't fit in its entirety.  PUT the contents of the buffer
;  4          ! and then loop back and write the remainder.
;  4          !-
;  4
;  5              BEGIN
;  5              BAS$$DO_WRITE (.WRITE_STATUS);
;  5
;  6              IF (.WRITE_STATUS EQL BAS$K_COND_SUC OR .WRITE_STATUS EQL BAS$K_CRLF)
;  5                  AND .L_ELEM_SIZE EQL 0
;  5              THEN
;  5
;  6                  BEGIN
;  6 !+
;  6 ! This field fit but there was not a full print zone left; the
;  6 ! format character was a comma.  We want to write this record
;  6 ! and then start the next element on the next line at the
;  6 ! left margin.
;  6 ! Or, a CRLF was found and they are the last two characters in
;  6 ! the record and we don't want to put out a null record after this
;  6 ! record.  So, bail out here.
```

```
6  ! Turn off the format character flag.  This avoids the situation where this is the
6  ! last element in this I/O list and ends with a semicolon or comma.  Even though
6  ! 'pre' carriage control has been set to 'LF', IO_BEG for the next I/O list will
6  ! set it to null if the format character flag is on.
6  !-
6                       CCB [LUB$V_FORM_CHAR] = 0;
6                       EXITLOOP;
5                       END;
5                   END
4               ELSE
5                   BEGIN
5  !+
5  ! everything fit this time.  Set the buffer dirty pointer.
5  !-
5                   CCB [LUB$V_OUTBUF_DR] = 1;
5                   EXITLOOP;
4                   END;
4
3               END;                                    ! WHILE loop
3
3  !+
3  ! Free the dynamic string allocated for packed decimal conversion.
3  !-
3
3           IF .ELEM_TYPE EQL DSC$K_DTYPE_P
3           THEN
3               STR$FREE1_DX (TEMP_DEC_DSC);
3
3  !+
3  ! Check for MAT PRINT.  If the format character flag is not set, then this
3  ! matrix is being printed one element/line.  Therefore it is necessary to get to
3  ! the REC level and do a PUT.
3  !-
3           IF (.CCB [ISB$B_STTM_TYPE] EQL ISB$K_ST_TY_MPR) AND NOT .CCB[LUB$V_FORM_CHAR]
3           THEN
4               BEGIN
4               BAS$$REC_MPR1();
4               CCB [LUB$L_PRINT_POS] = 0;
3               END;
3           END
2       UNTIL .CCB [LUB$V_AST_GUARD];                   ! End of AST guard loop
2
2       CCB [LUB$V_AST_GUARD] = 0;
2       RETURN;
1       END;                                            ! End of BAS$$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           06        04   AC   CF 0007A       CASEL   ELEM_TYPE, #6, #22                     ; 1741
  002E  0033         0033      0033           0007F 7$:   .WORD   9$-7$,-                                ;
  002E  002E         00F0      00F0           00087               9$-7$,-                                ;
  002E  002E         002E      032D           0008F               9$-7$,-                                ;
  032D  002E         002E      002E           00097               8$-7$,-                                ;
  002E  002E         002E      002E           0009F               20$-7$,-                               ;
        023F         01A0      002E           000A7               20$-7$,-                               ;
                                                                  8$-7$,-                                ;
                                                                  8$-7$,-                                ;
                                                                  54$-7$,-                               ;
                                                                  8$-7$,-                                ;
                                                                  8$-7$,-                                ;
                                                                  8$-7$,-                                ;
                                                                  8$-7$,-                                ;
                                                                  8$-7$,-                                ;
                                                                  8$-7$,-                                ;
                                                                  54$-7$,-                               ;
                                                                  8$-7$,-                                ;
                                                                  8$-7$,-                                ;
                                                                  8$-7$,-                                ;
                                                                  8$-7$,-                                ;
                                                                  8$-7$,-                                ;
                                                                  30$-7$,-                               ;
                                                                  38$-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            50  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           05   FB 001B5       CALLS   #5, BAS$CVT_OUT_D_G             ;
                  03           50   E8 001BC       BLBS    R0, 26$                         ;
                       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$:    CLRL    L_ELEM_SIZE                              :
                            05  5A  E9 0036A         BLBC    R10, 46$                                 :
                            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      0C  AC  D0 003B6 55$:    MOVL    ELEM_ADR, R0                             : 1876
                    51      04  A0  D0 003BA         MOVL    4(R0), SRC_LOC                           :
                            38  AE  9F 003BE 56$:    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$:     MOVZWL  DSC, R0                                 ;
                           66   BC BB40 9E 0047D          MOVAB   @-68(CCB)[R0], (R6)                     ;
              10           AC       01  D0 00482          MOVL    #1, FORMAT_CHAR                         ;
                                0C  AE  D4 00486          CLRL    L_ELEM_SIZE                             ;
              0E     A1    AB       01  E0 00489 68$:     BBS     #1, -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$$DO_WRITE                       ;
                     02          52  D1 004F0         CMPL    WRITE_STATUS, #2                    ; 1952
                                 05  13 004F3         BEQL    79$                                 ;
                     04          52  D1 004F5         CMPL    WRITE_STATUS, #4                    ;
                                 03  12 004F8         BNEQ    80$                                 ;
                           0C    AE  D5 004FA 79$:    TSTL    L_ELEM_SIZE                         ; 1953
                                 03  13 004FD 80$:    BEQL    81$                                 ;
                           FB69  31 004FF         BRW     6$                                  ;
              FE     AB          04  8A 00502 81$:    BICB2   #4, -2(CCB)                         ; 1970
                                 04  11 00506         BRB     83$                                 ; 1956
              FE     AB          08  88 00508 82$:    BISB2   #8, -2(CCB)                         ; 1979
                     15    04    AC  D1 0050C 83$:    CMPL    ELEM_TYPE, #21                      ; 1989
                                 0A  12 00510         BNEQ    84$                                 ;
                           30    AE  9F 00512         PUSHAB  TEMP_DEC_DSC                        ; 1991
         00000000G   00          01  FB 00515         CALLS   #1, STR$FREE1_DX                    ;
                     35  FF71    CB  91 0051C 84$:    CMPB    -143(CCB), #53                      ; 1998
                                 0E  12 00521         BNEQ    85$                                 ;
       09     FE     AB          02  E0 00523         BBS     #2, -2(CCB), 85$                    ;
                     00000000G   00  16 00528         JSB     BAS$$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
```

```
1 GLOBAL ROUTINE BAS$$UDF_WL9 : JSB_UDF9 NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Call the record level I/O end of list routine.  Reset the cursor position
1 !       if a PUT was done
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       LUB$V_AST_GUARD         Guard for AST reentrancy
1 !       LUB$V_FORM_CHAR         last element transmitter ended with a format char
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       LUB$V_AST_GUARD         guard for AST reentrancy
1 !       LUB$L_PRINT_POS         current cursor position
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       This routine will loop back and reexecute if it detects that it was
1 !       called by an AST while it was executing.
1 !
1 !--
1
2    BEGIN
2
2    EXTERNAL REGISTER
2       CCB : REF BLOCK [, BYTE];
2
2    !+
2    ! This outer loop is to detect an AST calling this routine while it is
2    ! executing.
2    !-
2
2    DO
3       BEGIN
3       CCB [LUB$V_AST_GUARD] = 1;              ! Initialize the guard bit
3 !+
3 ! Clear the bit which indicates an array list for MAT PRINT.
3 !-
3       CCB [ISB$V_MAT_PRINT] = 0;
3 !+
3 ! Dispatch to the REC level.
3 !-
3       JSB_REC9(BAS$$AA_REC_PR9 + .BAS$$AA_REC_PR9 [.CCB [ISB$B_STTM_TYPE] - ISB$K_BASSTTYLO + 1]);
3       ! Time to reset the cursor position to zero perhaps
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 0001A        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
```

```
1 ROUTINE SCAN_TEXT (POS,                              ! adr of beginning position of source string
1         LENGTH,                                      ! length of the source string
1         BUF_LENGTH,                                  ! length of the buffer remaining
1         RET_STR                                      ! adr of where to put the string
1     ) : CALL_CCB =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine puts a string of ASCII characters into the indicated buffer.
1 !       All non-printing characters, backspace, K_TAB, and carriage return
1 !       are identified in the string and their effect is reflected in the
1 !       internal cursor position.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       POS.rlu.r               position of the start of the string to output
1 !       LENGTH.rlu.v            length of the string to write
1 !       BUF_LENGTH.rlu.v        length remaining of the output buffer
1 !       RET_STR.wt.dx           pointer to descriptor for the destination
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       LUB$L_PRINT_POS         current cursor position
1 !       LUB$W_R_MARGIN          maximum allowable cursor position
1 !       LUB$A_BUF_PTR           current position in the buffer
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       LUB$L_PRINT_POS         current cursor position
1 !
1 ! ROUTINE VALUE:
1 !
1 !       BAS$K_MAR_EXC           right margin exceeded
1 !       BAS$K_CRLF              embedded CRLF in the record
1 !       BAS$K_SUCCESS           successful completion
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     MAP
2         POS : REF VECTOR [, BYTE],
2         RET_STR : REF BLOCK [8, BYTE];
2
2     EXTERNAL REGISTER
2         CCB : REF BLOCK [, BYTE];
2
2     LITERAL
2         K_STOP = %X'00',                             ! stop or escape character for MOVTUC
2         K_TILDE = %X'7E',                            ! tilde character
2         K_LF = %X'0A',                               ! line feed character
2         K_BKSP = %X'08',                             ! backspace character
```

```
        K_TAB = %X'09',                                 ! tab character
        K_CR = %X'0D',                                  ! carriage return character
        K_BLANK = %X'20',                               ! blank character
        K_TAB_LIT = %X'07';

    BIND
        ESC = UPLIT BYTE(K_STOP),

        !+
        ! This translation table is used for a MOVTUC instruction.  It trans-
        ! lates all normal 7 bit alphanumerics into themselves and stops on
        ! characters which affect the cursor position (backspace, tab, and CR).
        !-

        TRANS_TABLE = UPLIT BYTE(%X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X
            '00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00'        ! column 0
            %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00',
            %X'00', %X'00', %X'00', %X'00',             ! column 1
            %X'20', %X'21', %X'22', %X'23', %X'24', %X'25', %X'26', %X'27', %X'28', %X'29', %X'2A', %X'2B',
            %X'2C', %X'2D', %X'2E', %X'2F',             ! column 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',
            %X'3C', %X'3D', %X'3E', %X'3F',             ! column 3
            %X'40', %X'41', %X'42', %X'43', %X'44', %X'45', %X'46', %X'47', %X'48', %X'49', %X'4A', %X'4B',
            %X'4C', %X'4D', %X'4E', %X'4F',             ! column 4
            %X'50', %X'51', %X'52', %X'53', %X'54', %X'55', %X'56', %X'57', %X'58', %X'59', %X'5A', %X'5B',
            %X'5C', %X'5D', %X'5E', %X'5F',             ! column 5
            %X'60', %X'61', %X'62', %X'63', %X'64', %X'65', %X'66', %X'67', %X'68', %X'69', %X'6A', %X'6B',
            %X'6C', %X'6D', %X'6E', %X'6F',             ! column 6
            %X'70', %X'71', %X'72', %X'73', %X'74', %X'75', %X'76', %X'77', %X'78', %X'79', %X'7A', %X'7B',
            %X'7C', %X'7D', %X'7E', %X'00',             ! column 7
            %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00',
            %X'00', %X'00', %X'00', %X'00',             ! column 8
            %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00', %X'00',
            %X'00', %X'00', %X'00', %X'00',             ! column 9
            %X'A0', %X'A1', %X'A2', %X'A3', %X'A4', %X'A5', %X'A6', %X'A7', %X'A8', %X'A9', %X'AA', %X'AB',
            %X'AC', %X'AD', %X'AE', %X'AF',             ! column 10
            %X'B0', %X'B1', %X'B2', %X'B3', %X'B4', %X'B5', %X'B6', %X'B7', %X'B8', %X'B9', %X'BA', %X'BB',
            %X'BC', %X'BD', %X'BE', %X'BF',             ! column 11
            %X'C0', %X'C1', %X'C2', %X'C3', %X'C4', %X'C5', %X'C6', %X'C7', %X'C8', %X'C9', %X'CA', %X'CB',
            %X'CC', %X'CD', %X'CE', %X'CF',             ! column 12
            %X'D0', %X'D1', %X'D2', %X'D3', %X'D4', %X'D5', %X'D6', %X'D7', %X'D8', %X'D9', %X'DA', %X'DB',
            %X'DC', %X'DD', %X'DE', %X'DF',             ! column 13
            %X'E0', %X'E1', %X'E2', %X'E3', %X'E4', %X'E5', %X'E6', %X'E7', %X'E8', %X'E9', %X'EA', %X'EB',
            %X'EC', %X'ED', %X'EE', %X'EF',             ! column 14
            %X'F0', %X'F1', %X'F2', %X'F3', %X'F4', %X'F5', %X'F6', %X'F7', %X'F8', %X'F9', %X'FA', %X'FB',
            %X'FC', %X'FD', %X'FE', %X'00')             ! column 15
        : VECTOR [256, BYTE];

    LOCAL
        L_SAVE_LENGTH,                                  ! save length of string scanned so far
        DEST_LENGTH,                                    ! length of destination buffer
        SRC_LENGTH,                                     ! length of source string
        A_SAVE_BUF_PTR,                                 ! temp to save LUB$A_BUF_PTR
        T_DSC : BLOCK [8, BYTE],                        ! working desc to hold the fourth argument
        END_POS,                                        ! no. of characters in source string + current
                                                        ! address in the buffer
        L_SAV_PRINT_POS,                                ! temp to save the cursor position
```

```
                                                        ! in case the right margin is exceeded
        T_POS,                                          ! local to hold readonly arg. POS
        RET_VAL,                                        ! the status value returned to the caller.
        MOV_VAL,                                        ! receives the returned status from MOVTUC
        T_BUF_PTR,                                      ! temp buffer ptr, to agree with print pos
        ORIG_BUF_PTR;                                   ! another buf ptr, unmodified so
                                                        ! LUB$A_BUF_PTR can be restored

    !+
    ! Save off a number of pointers.  We will not overflow the buffer because
    ! the MOVTUC which is used to load the buffer has a destination length operand,
    ! however it is possible to exceed the right margin if there are one or more
    ! TAB characters.  If the margin is exceeded, it is detected in the macro
    ! CHK_CURSOR_POS, then the string is backed up and each character is moved
    ! until the right margin is reached.  Normally, if an element does not fit
    ! into the buffer, the buffer is dumped without it.  However, there is the
    ! case where a string is longer than the buffer and must be put out in seg-
    ! ments.  That is not to be decided in this routine.
    !-

    DEST_LENGTH = .BUF_LENGTH;
    SRC_LENGTH = .LENGTH;
    T_POS = .POS;
    T_BUF_PTR = .CCB [LUB$A_BUF_PTR];
    ORIG_BUF_PTR = .CCB [LUB$A_BUF_PTR];
    T_DSC [DSC$W_LENGTH] = 0;
    T_DSC [DSC$A_POINTER] = .RET_STR [DSC$A_POINTER];
    END_POS = MINU (.SRC_LENGTH, .DEST_LENGTH) + .T_POS;
    MOV_VAL = 1;

    !+
    ! This loop will find all of the cursor position affecting characters in
    ! the string.
    !-

    WHILE .MOV_VAL NEQU 0 DO
        BEGIN
        RET_VAL = (IF .DEST_LENGTH GEQ .SRC_LENGTH THEN BAS$K_SUCCESS ELSE BAS$K_BUF_EXC);
        A_SAVE_BUF_PTR = .T_BUF_PTR;
        L_SAV_PRINT_POS = .CCB [LUB$L_PRINT_POS];
        L_SAVE_LENGTH = .T_DSC [DSC$W_LENGTH];
        MOV_VAL = MOVTUC (SRC_LENGTH, .T_POS, ESC, TRANS_TABLE, DEST_LENGTH, .T_DSC [DSC$A_POINTER]);

        !+
        ! Set RET_VAL to either 1 (success) or 0 (failure) based on whether
        ! there was enough room in the buffer for the entire translated
        ! string.  A value of 1 at this point may be changed in the macro
        ! CHK_CURSOR_POS.
        !-

        IF .MOV_VAL EQLU 0
        THEN

        !+
        ! MOVTUC was able to move the entire source string without encountering
        ! an 'escape' character.  Update the cursor position and check to
        ! see if it exceeds the right margin.
```

```
3           !-
3
4               BEGIN
4               !
4               ! SWITCHES LIST (EXPAND);
4               !
4               T_DSC [DSC$W_LENGTH] = .T_DSC [DSC$W_LENGTH] + (.END_POS - .T_POS);
4               CCB [LUB$L_PRINT_POS] = .CCB [LUB$L_PRINT_POS] + (.END_POS - .T_POS);
4               T_BUF_PTR = .T_BUF_PTR + .SRC_LENGTH;
4               CHK_CURSOR_POS;
4               END
3           ELSE
4               BEGIN
4
4               !+
4               ! an 'escape' character was detected in the source string.
4               ! increment the cursor position for the substring moved plus one
4               ! for the offending character.  the buffer pointer must be moved
4               ! to keep up with the cursor position.
4               !-
4
4               CCB [LUB$L_PRINT_POS] = .CCB [LUB$L_PRINT_POS] + (.MOV_VAL - .T_POS);
4               T_BUF_PTR = .T_BUF_PTR + (.MOV_VAL - .T_POS) + 1;
4               T_DSC [DSC$W_LENGTH] = .T_DSC [DSC$W_LENGTH] + (.MOV_VAL - .T_POS) + 1;
4
4       !+
4       ! Check for an occurrence of a CRLF which will delimit the record and insure that
4       ! the LF is included in the buffer.
4       !-
4       !+
4       ! We should do a LSSU instead of LEQU, to fix 1-065D.
4       !-
4               IF .(.MOV_VAL)<0, 8> EQL K_CR
4                   AND .(MOV_VAL) + 1 LSSU .POS + .LENGTH
4                   AND .(.(MOV_VAL) + 1)<0, 8> EQL K_LF
4               THEN
5                   BEGIN
5       !+
5       ! There is a CRLF in this record.  Set the length properly to include the CRLF
5       ! and return the fact that a delimiting CRLF was found.
5       !-
5                   T_DSC [DSC$W_LENGTH] = .T_DSC [DSC$W_LENGTH] + 1;
5                   CHK_CURSOR_POS;
5                   RET_VAL = BAS$K_CRLF;
5                   EXITLOOP;
5                   END
5
5               !+
5               ! Adjust the cursor position for the 'escape' character if the
5               ! 'escape' character was not CR followed by LF.  We don't want
5               ! to reset PRINT_POS for CRLF because it is a complete record
5               ! and must be evaluated as such for both comma zone formatting and
5               ! exceeding the right margin.
5               !-
5
4               ELSE
5                   CCB [LUB$L_PRINT_POS] = (SELECTONE .(.MOV_VAL)<0, 8, 0> OF
```

```
                        SET
                        [K_TAB] : ((.CCB [LUB$L_PRINT_POS] OR K_TAB_LIT) + 1);
                        [K_CR] : 0;
                        [K_BKSP] : MAX (0, .CCB [LUB$L_PRINT_POS] - 1);
                        [OTHERWISE] : .CCB [LUB$L_PRINT_POS];
                        TES);
                SRC_LENGTH = .SRC_LENGTH - (.MOV_VAL - .T_POS) - 1;
                T_DSC [DSC$A_POINTER] = .T_DSC [DSC$A_POINTER] + (.MOV_VAL - .T_POS);
                DEST_LENGTH = .DEST_LENGTH - (.MOV_VAL - .T_POS) - 1;
                T_POS = .MOV_VAL + T;

                !+
                ! move the offending character by hand
                !-

                T_DSC [DSC$A_POINTER] = CH$MOVE (1, CH$PTR (.MOV_VAL), CH$PTR (.T_DSC [DSC$A_POINTER]));
                CHK_CURSOR_POS;
                END;

            END;                                        ! WHILE loop

        !+
        ! Set the length of the string which was transferred into the indicated
        ! destination.  Restore original buffer pointer so that BAS$UDF_WL1 can
        ! update it properly.
        !-

        CCB [LUB$A_BUF_PTR] = .ORIG_BUF_PTR;
        RET_STR [DSC$W_LENGTH] = .T_DSC [DSC$W_LENGTH];
        RETURN .RET_VAL;
        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 2B 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                  123, 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    @16(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$:       SUBL3   T_POS, END_POS, R0                            ; 2253
         24   AE          50  A0 00086           ADDW2   R0, T_DSC                                     ;
         10   BE          50  C0 0008A           ADDL2   R0, @16(SP)                                   ; 2254
         18   AE          57  C0 0008E           ADDL2   SRC_LENGTH, T_BUF_PTR                         ; 2255
     AA  A1   AB          01  E0 00092           BBS     #1, -95(CCB), 2$                              ;
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          50   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)          ; 2331
                    10  BC          24  AE  B0 002A8       MOVW    T_DSC, @RET_STR                ; 2332
                        50          0C  AE  D0 002AD       MOVL    RET_VAL, R0                    ; 2333
                                        04     002B1       RET                                    ; 2334

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

; 1298            2335  1
```

```
GLOBAL ROUTINE BAS$$DO_WRITE(                ! Write one record with indicated carriage control
    CARRIAGE_CTRL
        ) : JSB_DO_WRITE NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!       This routine centralizes calls to the record level write routine.  A
!       few vlues in the data base are reset.
!
! FORMAL PARAMETERS:
!
!       CARRIAGE_CTRL.rlu.v     indicates whether to insert CRLF or leave the cursor
!
! IMPLICIT INPUTS:
!
!       CCB                     address of current control block
!
! IMPLICIT OUTPUTS:
!
!       LUB$V_OUTBUF_DR         indicates valid data in the output buffer
!       LUB$L_PRINT_POS         current cursor position
!
! COMPLETION CODES:
!
!       NONE
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    EXTERNAL REGISTER
        CCB : REF BLOCK [, BYTE];

    BAS$$REC_WSL1 (.CARRIAGE_CTRL);

    !+
    ! Reset the output buffer dirty bit
    !-
!+
! Reset the cursor position only if a record was written.  Otherwise, the
! cursor was left at the end of the data written and the cursor position
! is still valid
!-

    IF .CCB [LUB$A_BUF_PTR] EQL .CCB [LUB$A_BUF_BEG] THEN CCB [LUB$L_PRINT_POS] = 0;
    CCB [LUB$V_OUTBUF_DR] = 0;
    RETURN;
    END;
```

```
                              0000000G  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
```

0033 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

BASVIRTUA
LIS

BASUDFWL
LIS

BASUNLOCK
LIS

BASVECTOR
LIS

BASVAL
LIS

BASVIRTIO
LIS

BASUNWIND
LIS

BASUPDATE
LIS

BASVECTR2
LIS