F 1

```
DDDDDDDDDDDD      EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUU         UUU     GGGGGGGGGGGG
DDDDDDDDDDDD      EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUU         UUU     GGGGGGGGGGGG
DDDDDDDDDDDD      EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUU         UUU     GGGGGGGGGGGG
DDD         DDD  EEE              BBB         BBB  UUU         UUU  GGG
DDD         DDD  EEE              BBB         BBB  UUU         UUU  GGG
DDD         DDD  EEE              BBB         BBB  UUU         UUU  GGG
DDD         DDD  EEE              BBB         BBB  UUU         UUU  GGG
DDD         DDD  EEE              BBB         BBB  UUU         UUU  GGG
DDD         DDD  EEE              BBB         BBB  UUU         UUU  GGG
DDD         DDD  EEEEEEEEEEEE     BBBBBBBBBBBB     UUU         UUU  GGG
DDD         DDD  EEEEEEEEEEEE     BBBBBBBBBBBB     UUU         UUU  GGG
DDD         DDD  EEEEEEEEEEEE     BBBBBBBBBBBB     UUU         UUU  GGG
DDD         DDD  EEE              BBB         BBB  UUU         UUU  GGG   GGGGGGGGG
DDD         DDD  EEE              BBB         BBB  UUU         UUU  GGG   GGGGGGGGG
DDD         DDD  EEE              BBB         BBB  UUU         UUU  GGG   GGGGGGGGG
DDD         DDD  EEE              BBB         BBB  UUU         UUU  GGG         GGG
DDD         DDD  EEE              BBB         BBB  UUU         UUU  GGG         GGG
DDD         DDD  EEE              BBB         BBB  UUU         UUU  GGG         GGG
DDDDDDDDDDDD     EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUUUUUUUUUUUUUU     GGGGGGGGG
DDDDDDDDDDDD     EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUUUUUUUUUUUUUU     GGGGGGGGG
DDDDDDDDDDDD     EEEEEEEEEEEEEEE  BBBBBBBBBBBB     UUUUUUUUUUUUUUU     GGGGGGGGG
```

```
                                                                        H 15
**FILE**ID**DBGDPC




DDDDDDDD    BBBBBBBB      GGGGGGGG  DDDDDDDD    PPPPPPPP      CCCCCCCC
DDDDDDDD    BBBBBBBB      GGGGGGGG  DDDDDDDD    PPPPPPPP      CCCCCCCC
DD      DD  BB      BB  GG          DD      DD  PP      PP  CC
DD      DD  BB      BB  GG          DD      DD  PP      PP  CC
DD      DD  BB      BB  GG          DD      DD  PP      PP  CC
DD      DD  BB      BB  GG          DD      DD  PP      PP  CC
DD      DD  BBBBBBBB    GG          DD      DD  PPPPPPPP    CC
DD      DD  BBBBBBBB    GG          DD      DD  PPPPPPPP    CC
DD      DD  BB      BB  GG  GGGGGG  DD      DD  PP          CC
DD      DD  BB      BB  GG  GGGGGG  DD      DD  PP          CC
DD      DD  BB      BB  GG      GG  DD      DD  PP          CC              ....
DD      DD  BB      BB  GG      GG  DD      DD  PP          CC              ....
DDDDDDDD    BBBBBBBB      GGGGGG    DDDDDDDD    PP            CCCCCCCC      ....
DDDDDDDD    BBBBBBBB      GGGGGG    DDDDDDDD    PP            CCCCCCCC      ....


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
```

```
MODULE DBGDPC   ( IDENT = 'V04-000') =
BEGIN

!
!****************************************************************************
!*                                                                          *
!*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
!*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
!*  ALL RIGHTS RESERVED.                                                    *
!*                                                                          *
!*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
!*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
!*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
!*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
!*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
!*  TRANSFERRED.                                                            *
!*                                                                          *
!*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
!*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
!*  CORPORATION.                                                            *
!*                                                                          *
!*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
!*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
!*                                                                          *
!*                                                                          *
!****************************************************************************
!

!++
! FACILITY:     DEBUG (DBG)
!
! ABSTRACT:
!       Analyzes PC correlation tables for DEBUG.
!
! ENVIRONMENT:  VAX/VMS, user mode, interrupts disabled.
!
! AUTHOR:       Carol Peters, CREATION DATE:    16 September 1977
!
! Version       3.01
!
! MODIFIED BY:
!       (PS = Ping Sager, RT = Rich Title, JF = John Francis)
!
! 3.01  15-Sep-81       PS      Correct LINE_END PC address calculation in
!                               PC_TO_LINE_LOOKUP.
! 3.02  23-Apr-82       RT      Fixed a bug in DBG$PC_TO_LINE_LOOKUP: the routine
!                               was assuming that chasing upscope pointers will
!                               always get you to a routine RST entry.
! 4.0   13-Dec-82       PS      Switched some old symbolization routines to
!                               use new code.
!        1-Mar-83       JF      Changed return values from DBG$PC_TO_LINE_LOOKUP
!                               so that SUCCESS and FAILURE are shown properly
!       12-Apr-83       RT      Fixed a bug in PC_TO_LINE
!       24-Dec-83       RT      Added comments and did some general cleanup
!--
```

```
! TABLE OF CONTENTS:
!
FORWARD ROUTINE
        dbg$line_to_pc_lookup,  ! Given line number associated it to a PC
        dbg$pc_to_line,         ! Matches a PC to a line number
        dbg$pc_to_line_lookup,  ! Given PC looks up associated line number
        proc_pc_cmd,            ! Processes a string of PC correlation commands
        find_eol,               ! Find end of line
        give_line_info: NOVALUE;! Give more info about line number


! INCLUDE FILES:
!
REQUIRE 'SRC$:DBGPROLOG.REQ';
LIBRARY 'LIB$:DBGGEN.L32';


! MACROS:
!
MACRO
        current_byte     = 0, 0, 8, 1%,           ! current top of record
        next_uns_byte    = 1, 0, 8, 0%,           ! byte argument to command
        next_uns_word    = 1, 0, 16, 0%,          ! word argument to command
        next_uns_long    = 1, 0, 32, 0%,          ! longword argument to command
        add_one_byte     = 1, 0, 8, 0%,           ! increment for top of record
        add_two_bytes    = 2, 0, 8, 0%,           ! ditto
        add_three_bytes  = 3, 0, 8, 0%,           ! ditto
        add_five_bytes   = 5, 0, 8, 0%;           ! ditto


! EQUATED SYMBOLS:
!
LITERAL
        line_open        = 1,
        line_closed      = 2;


! OWN STORAGE:
!
OWN
        dst_entry        : REF dst$record,
        dpc_entry        : REF BLOCK [, BYTE],
        start_pc,
        current_line,
        current_stmt,
        current_incr,
        current_pc,
        current_stmt_mode,
        current_mark,
        prev_line,
        prev_stmt,
        prev_incr,
        prev_pc,
        prev_stmt_mode,
        prev_mark,
        NUM_PC_TBLS,                         ! The number of PC-Correlation DST
                                             !         records for the current module
        current_table,
        report_next_line,
        report_next_stmt,
        report_prev_line,
        report_prev_stmt,
        pctbl_count;


! EXTERNAL REFERENCES:
!
EXTERNAL ROUTINE
        dbg$format_fao_out: NOVALUE,    ! Forward FAO string
        dbg$pc_to_symid;                ! Search Moudle SAT to locate RST
```

```
1 ROUTINE dbg$pc_to_line (match_pc_ptr, modpctbl,     pctbl_base,
1                         line_no_ptr,  stmt_no_ptr, line_pc) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine matches an address to a line number.
1 !       The caller, DBG$PC_TO_LINE_LOOKUP, does the work of finding
1 !       the PC/LINE table for the module containing the address.
1 !       A pointer to this table is passed to this routine.
1 !
1 !       Each PC correlation record that exists for the module
1 !       is sequentially analyzed until the desired address
1 !       is seen.
1 !
1 !       See the comments in DBG$PC_TO_LINE_LOOKUP for more details
1 !       about how this routine is used.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       match_pc_ptr    - The address to be matched.
1 !       modpctbl        - The address of the table of pointers to
1 !                         PC/LINE tables in this module. The first
1 !                         longword of the table is a count of PC/LINE
1 !                         tables, and the remaining longwords are
1 !                         pointers to the DST records containing the tables.
1 !       pctbl_base      - The address which is the base address for
1 !                         the PC/LINE tables
1 !       line_no_ptr     - An output parameter for the line number.
1 !       stmt_no_ptr     - An output parameter for the statement number.
1 !       line_pc         - An output parameter for the start pc of the
1 !                         selected line/stmt.
1 !
1 ! ROUTINE VALUE:
1 !
1 !       This routine returns one of three values: 0, 1, or 2.
1 !       Note that the caller, DBG$PC_TO_LINE_LOOKUP, may change
1 !       return status "1" to return status "3" if we did not get
1 !       an exact match. See that routine for further details
1 !       on how the return status is used.
1 !
1 !       0       - If no match can be made because pc/line tables are
1 !                 not available for the given address. This may occur
1 !                 because the module containing the address was not
1 !                 set or was compiled /NODEBUG, or because the address
1 !                 is in system space or in an RTL shareable image.
1 !       1       - If a line number/stmt number was found.
1 !       2       - If there are pc/line tables available for the
1 !                 module containing the given address, but no match
1 !                 was found. This occurs if the address is not within
1 !                 any line in the module. The use of the "TERM" record
1 !                 in PC/LINE tables terminates an address range for
1 !                 a line without starting a new line, and this can
1 !                 give rise to addresses without line numbers.
1 !--
2     BEGIN
2     MAP
2         MODPCTBL: REF VECTOR[,LONG];
```

```

! If we do not have a PC/LINE table, just return 0.
!
IF .MODPCTBL EQL 0 THEN RETURN 0;


! Set up the OWN variables that we use for reading the PC/LINE tables.
! This includes a count of the number of PC/LINE DST records in this
! module we have looked at so far (initialized to 1 here), a count
! of the total number of PC/LINE DST records in the module, a pointer
! to our position in the table of PC/LINE DST records,
! and a pointer to the first such DST record.
! If there are zero PC/LINE tables in this module, return 0 here.
!
PCTBL_COUNT = 1;
NUM_PC_TBLS = .MODPCTBL[0];
CURRENT_TABLE = MODPCTBL[1];
DST_ENTRY = .MODPCTBL[1];
IF .NUM_PC_TBLS EQL 0 THEN RETURN 0;


! Initialize the state variables (OWN variables in this module)
! that are used by PROC_PC_CMD.
!
current_line = 0;
current_stmt = 1;
current_incr = 1;
current_stmt_mode = FALSE;
current_pc = start_pc = .pctbl_base;
current_mark = line_closed;


! Call a routine that processes all PC correlation commands
! until a delta-PC command is seen. Then process that
! delta-PC command and return to this routine. If the processing
! is generally successful, return 1, otherwise return 0.
!
dpc_entry = dst_entry [dst$b_vflags];
REPEAT
    BEGIN
    prev_line = .current_line;
    prev_stmt = .current_stmt;
    prev_incr = .current_incr;
    prev_stmt_mode = .current_stmt_mode;
    prev_pc = .current_pc;
    prev_mark = .current_mark;


    ! If we PROC_PC_CMD fails we have come to the end
    ! of the PC/LINE table for this module, without finding
    ! a match. In this case, return 2, indicating that we
    ! are in a module with PC/LINE tables, but we could not
    ! match the given PC.
    !
    IF NOT proc_pc_cmd ( )
    THEN
```

```
                RETURN 2;


            ! Report a match to a line if:
            ! - the PC is within the range given by
            !     the previous PC and the current PC, and
            ! - the line is marked as being OPEN.
            !
            IF (.prev_pc LEQA .match_pc_ptr) AND
               (.match_pc_ptr LSSA .current_pc) AND
               (.prev_mark EQL line_open)
            THEN
                BEGIN
                .stmt_no_ptr = (IF .prev_stmt EQL 1 THEN 0
                                                    ELSE .prev_stmt); ! Huh?
                .line_no_ptr = .prev_line;
                .line_pc = .prev_pc;
                RETURN 1;
                END;


            ! Found nothing this round; continue trying.
            !
            END;            ! End of REPEAT.

        ! We have not found a match - return 2, indicating that we
        ! are in a module with PC/LINE tables, but we could not
        ! match the given PC.
        !
        RETURN 2;
        END;


                                        .TITLE  DBGDPC
                                        .IDENT  \V04-000\

                                        .PSECT  DBG$OWN,NOEXE,  PIC,2

                           00000 DST_ENTRY:
                                        .BLKB   4
                           00004 DPC_ENTRY:
                                        .BLKB   4
                           00008 START_PC:
                                        .BLKB   4
                           0000C CURRENT_LINE:
                                        .BLKB   4
                           00010 CURRENT_STMT:
                                        .BLKB   4
                           00014 CURRENT_INCR:
                                        .BLKB   4
                           00018 CURRENT_PC:
                                        .BLKB   4
                           0001C CURRENT_STMT_MODE:
                                        .BLKB   4
                           00020 CURRENT_MARK:
                                        .BLKB   4
```

```
                                                   00024 PREV_LINE:
                                                                    .BLKB    4
                                                   00028 PREV_STMT:
                                                                    .BLKB    4
                                                   0002C PREV_INCR:
                                                                    .BLKB    4
                                                   00030 PREV_PC:.BLKB    4
                                                   00034 PREV_STMT_MODE:
                                                                    .BLKB    4
                                                   00038 PREV_MARK:
                                                                    .BLKB    4
                                                   0003C NUM_PC_TBLS:
                                                                    .BLKB    4
                                                   00040 CURRENT_TABLE:
                                                                    .BLKB    4
                                                   00044 REPORT_NEXT_LINE:
                                                                    .BLKB    4
                                                   00048 REPORT_NEXT_STMT:
                                                                    .BLKB    4
                                                   0004C REPORT_PREV_LINE:
                                                                    .BLKB    4
                                                   00050 REPORT_PREV_STMT:
                                                                    .BLKB    4
                                                   00054 PCTBL_COUNT:
                                                                    .BLKB    4

                                                                    .EXTRN   DBG$FORMAT_FAO_OUT
                                                                    .EXTRN   DBG$PC_TO_SYMID

                                                                    .PSECT   DBG$CODE,NOWRT,  SHR,  PIC,0

                                              0004 00000 DBG$PC_TO_LINE:
                                                                    .WORD    Save R2                                          ; 0260
                      52 00000000'  EF   9E 00002                   MOVAB    NUM_PC_TBLS, R2                                  ;
                      50         08  AC  D0 00009                   MOVL     MODPCTBL, R0                                     ; 0321
                                     13  13 0000D                   BEQL     1$                                               ;
                 18   A2             01  D0 0000F                   MOVL     #1, PCTBL_COUNT                                  ; 0332
                      62             60  D0 00013                   MOVL     (R0), NUM_PC_TBLS                                ; 0333
                 04   A2         04  A0  9E 00016                   MOVAB    4(R0), CURRENT_TABLE                             ; 0334
                 C4   A2         04  A0  D0 0001B                   MOVL     4(R0), DST_ENTRY                                 ; 0335
                                     62  D5 00020                   TSTL     NUM_PC_TBLS                                      ; 0336
                                     73  13 00022 1$:               BEQL     6$                                               ;
                                 D0  A2  D4 00024                   CLRL     CURRENT_LINE                                     ; 0342
                 D4   A2             01  D0 00027                   MOVL     #1, CURRENT_STMT                                 ; 0343
                 D8   A2             01  D0 0002B                   MOVL     #1, CURRENT_INCR                                 ; 0344
                                 E0  A2  D4 0002F                   CLRL     CURRENT_STMT_MODE                                ; 0345
                      50         0C  AC  D0 00032                   MOVL     PCTBL_BASE, R0                                   ; 0346
                 CC   A2             50  D0 00036                   MOVL     R0, START_PC                                     ;
                 DC   A2             50  D0 0003A                   MOVL     R0, CURRENT_PC                                   ;
                 E4   A2             02  D0 0003E                   MOVL     #2, CURRENT_MARK                                 ; 0347
  C8    A2       C4   A2             02  C1 00042                   ADDL3    #2, DST_ENTRY, DPC_ENTRY                         ; 0355
                 E8   A2         D0  A2  7D 00048 2$:               MOVQ     CURRENT_LINE, PREV_LINE                          ; 0358
                 F8   A2         E0  A2  7D 0004D                   MOVQ     CURRENT_STMT_MODE, PREV_STMT_MODE                ; 0361
                 F0   A2         D8  A2  7D 00052                   MOVQ     CURRENT_INCR, PREV_INCR                          ; 0360
              0000V   CF             00  FB 00057                   CALLS    #0, PROC_PC_CMD                                  ; 0372
                      34             50  E9 0005C                   BLBC     R0, 5$                                           ;
                 04   AC         F4  A2  D1 0005F                   CMPL     PREV_PC, MATCH_PC_PTR                            ; 0382
```

```
                                   E2  1A 00064          BGTRU   2$                                    ;
           DC   A2          04     AC  D1 00066          CMPL    MATCH_PC_PTR, CURRENT_PC              ; 0383
                                   DB  1E 0006B          BGEQU   2$                                    ;
                01          FC     A2  D1 0006D          CMPL    PREV_MARK, #1                         ; 0384
                                   D5  12 00071          BNEQ    2$                                    ;
                01          EC     A2  D1 00073          CMPL    PREV_STMT, #1                         ; 0387
                                   04  12 00077          BNEQ    3$                                    ;
                                   50  D4 00079          CLRL    R0                                    ;
                                   04  11 0007B          BRB     4$                                    ;
                50          EC     A2  D0 0007D 3$:      MOVL    PREV_STMT, R0                         ; 0388
           14   BC                 50  D0 00081 4$:      MOVL    R0, @STMT_NO_PTR                      ; 0387
           10   BC          E8     A2  D0 00085          MOVL    PREV_LINE, @LINE_NO_PTR               ; 0389
           18   BC          F4     A2  D0 0008A          MOVL    PREV_PC, @LINE_PC                     ; 0390
                50                 01  D0 0008F          MOVL    #1, R0                                ; 0391
                                       04 00092          RET                                           ;
                50                 02  D0 00093 5$:      MOVL    #2, R0                                ; 0404
                                       04 00096          RET                                           ;
                                   50  D4 00097 6$:      CLRL    R0                                    ; 0405
                                       04 00099          RET                                           ;

; Routine Size:  154 bytes,    Routine Base:  DBG$CODE + 0000
```

```
1 GLOBAL ROUTINE DBG$LINE_TO_PC_LOOKUP (LINE_NUM, STMT_NUM, MC_PTR,
1                                   LINE_PC, LINE_END, FLAG) =
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine finds the absolute PC address associated with
1 !       a line number/statement number.
1 !
1 !       Each PC correlation record that exists for a single routine
1 !       is sequentially analyzed until the desired line number
1 !       is seen.
1 !
1 !       If a match cannot be made because an end of routine record or
1 !       an invalid record is recognized, then this routine returns
1 !       FALSE.
1 !
1 ! FORMAL PARAMETERS:
1 !       line_num        - the line number to find.
1 !       stmt_num        - the statement number to find.
1 !       mc_ptr          - module rstptr
1 !       line_pc         - where to store the computed address.
1 !       line_end        - a copy-back pointer for the line-end pc value.
1 !       flag            - flag set to indicate more line information is needed.
1 !
1 ! ROUTINE VALUE:
1 !       The routine value is TRUE if the desired line was successfully
1 !               found; it is FALSE otherwise.
1 !
1 !
1
2     BEGIN
2     MAP
2         MC_PTR: REF RST$ENTRY;
2
2     LOCAL
2         MODPCTBL: REF VECTOR[,LONG];
2
2
2      ! Adjust a statement number of 1 to 0 (%LINE 10.1 is equivalent
2      ! to %LINE 10, and the algorithm below coughs at statement numbers of 1
2      !
2     IF .STMT_NUM EQL 1 THEN STMT_NUM = 0;
2
2
2      ! Set up the OWN variables that we use for reading the PC/LINE tables.
2      ! This includes a count of the number of PC/LINE DST records in this
2      ! module we have looked at so far (initialized to 1 here), a count
2      ! of the total number of PC/LINE DST records in the module, a pointer
2      ! to our position in the table of PC/LINE DST records,
2      ! and a pointer to the first such DST record.
2      ! If there are zero PC/LINE tables in this module, return 0 here.
2      !
2     PCTBL_COUNT = 1;
2     MODPCTBL = .MC_PTR[RST$L_MODPCTBL];
2     IF .MODPCTBL EQL 0 THEN RETURN FALSE;
2     NUM_PC_TBLS = .MODPCTBL[0];
2     CURRENT_TABLE = MODPCTBL[1];
2     DST_ENTRY = .MODPCTBL[1];
```

```
IF .NUM_PC_TBLS EQL 0 THEN RETURN 0;


! Initialize state variables. These are OWN variables that
! are used by PROC_PC_CMD.
!
current_line = 0;
current_stmt = 1;
current_incr = 1;
current_stmt_mode = FALSE;
current_pc = start_pc = .mc_ptr[rst$l_pctbl_base];
current_mark = line_closed;


! Loop through the PC Correlation Tables for this module until the
! desired line number is found or the table ends.  To do this, we call
! PROC_PC_CMD to process all PC Correlation commands until a delta-PC
! command is found.  It then returns a PC and a line number and we
! check whether that is the line number we are looking for.  If not,
! we loop for the next line until the desired line is found or no PC
! Correlation commands remain.
!
dpc_entry = dst_entry [dst$b_vflags];
REPORT_PREV_LINE = 0;
REPORT_PREV_STMT = 1;
REPORT_NEXT_LINE = .LINE_NUM;
REPORT_NEXT_STMT = .STMT_NUM;
WHILE TRUE DO
    BEGIN


    ! Remember the previous values of all the state variables
    ! before getting the current values this time around.
    !
    PREV_LINE = .CURRENT_LINE;
    PREV_STMT = .CURRENT_STMT;
    PREV_INCR = .CURRENT_INCR;
    PREV_STMT_MODE = .CURRENT_STMT_MODE;
    PREV_PC = .CURRENT_PC;
    PREV_MARK = .CURRENT_MARK;


    ! Call PROC_PC_CMD to get the next PC - line number pair.
    ! When there are no more lines, exit this loop.
    !
    IF NOT PROC_PC_CMD() THEN EXITLOOP;


    ! Set report next line and stmt for the first time.
    !
    IF (.REPORT_NEXT_LINE EQL .LINE_NUM) AND
       (.REPORT_NEXT_STMT EQL .STMT_NUM)
    THEN
        BEGIN
        IF (.CURRENT_LINE GTR .LINE_NUM) OR
           ((.CURRENT_LINE EQL .LINE_NUM) AND
            (.CURRENT_STMT GTR .STMT_NUM))
    THEN
        BEGIN
        REPORT_NEXT_LINE = .CURRENT_LINE;
        REPORT_NEXT_STMT = .CURRENT_STMT;
        END;

    END;


! At this point we have Prev. line, current line, and given line info.
! So we define the reporting line information centered around given line.
! (we choose the closest two ends value).
!
! Define report prev. line.
!
IF .REPORT_PREV_LINE LSS .LINE_NUM
THEN
    BEGIN
    IF .PREV_LINE LSS .LINE_NUM
    THEN
        REPORT_PREV_LINE = MAX(.REPORT_PREV_LINE, .PREV_LINE)
    ELSE
        BEGIN
        IF ((.PREV_LINE EQL .LINE_NUM) AND
            (.PREV_STMT LSS .STMT_NUM))
        THEN
            BEGIN
            REPORT_PREV_LINE = .PREV_LINE;
            REPORT_PREV_STMT = .PREV_STMT;
            END;

        END;

    END
ELSE
    BEGIN
    IF ((.REPORT_PREV_LINE EQL .LINE_NUM) AND
        (.REPORT_PREV_STMT LSS .STMT_NUM))
    THEN
        BEGIN
        IF (.PREV_LINE EQL .LINE_NUM) AND
           (.PREV_STMT LSS .STMT_NUM)
        THEN
            REPORT_PREV_STMT = MAX(.PREV_STMT, .REPORT_PREV_STMT);

        END;

    END;


! Define report next line.
!
IF .REPORT_NEXT_LINE GTR .LINE_NUM
THEN
    BEGIN
    IF .CURRENT_LINE GTR .LINE_NUM
```

```
4                   THEN
4                       REPORT_NEXT_LINE = MIN(.REPORT_NEXT_LINE, .CURRENT_LINE)
4                   ELSE
5                       BEGIN
6                       IF ((.CURRENT_LINE EQL .LINE_NUM) AND
6                           (.CURRENT_STMT GTR .STMT_NUM))
5                       THEN
6                           BEGIN
6                           REPORT_NEXT_LINE = .CURRENT_LINE;
6                           REPORT_NEXT_STMT = .CURRENT_STMT;
5                           END;
5
4                       END;
4
4                   END
4
3               ELSE
4                   BEGIN
5                   IF ((.REPORT_NEXT_LINE EQL .LINE_NUM) AND
5                       (.REPORT_NEXT_STMT GTR .STMT_NUM))
4                   THEN
5                       BEGIN
5                       IF (.CURRENT_LINE EQL .LINE_NUM) AND
6                          (.CURRENT_STMT GTR .STMT_NUM)
5                       THEN
5                           REPORT_PREV_STMT = MIN(.CURRENT_STMT, .REPORT_NEXT_STMT);
5
4                       END;
4
3                   END;
3
3
3               ! Note that: above code did not take care of the equality condition.
3               ! it should be set up here and tested in give_line_info.
3               !
3               ! If the current line number is equivalent to the one we were
3               ! passed (and this includes the statement number), then we
3               ! return the corresponding PC to LINE_PC and we return TRUE.
3               ! If we are at the right line but there is no such statement
3               ! number, we clear LINE_PC and return FALSE.
3               !
3               IF .CURRENT_LINE EQL .LINE_NUM
3               THEN
4                   BEGIN
4                   IF  MAX (.CURRENT_STMT, 1) EQL MAX(.STMT_NUM, 1)
4                   THEN
5                       BEGIN
5                       .LINE_PC = .CURRENT_PC;
5                       IF NOT FIND_EOL(.LINE_END)
5                       THEN
6                           BEGIN
6                           IF .FLAG THEN GIVE_LINE_INFO(.LINE_NUM, .STMT_NUM);
6                           RETURN FALSE;
5                           END;
5
5                       RETURN TRUE;
5                       END
```

```
;   504        0634  5              ELSE
;   505        0635  4                  BEGIN
;   506        0636  5                  IF MAX(.CURRENT_STMT,1) GTR MAX(.STMT_NUM,1)
;   507        0637  5                  THEN
;   508        0638  5                      BEGIN
;   509        0639  6                      .LINE_PC = 0;
;   510        0640  6                      IF .FLAG THEN GIVE_LINE_INFO(.LINE_NUM, .STMT_NUM);
;   511        0641  6                      RETURN FALSE;
;   512        0642  6                      END;
;   513        0643  5
;   514        0644  5                  END;
;   515        0645  4
;   516        0646  4              END;
;   517        0647  3
;   518        0648  3          END;                                    ! End of WHILE loop over PC Corr Tbl
;   519        0649  2
;   520        0650  2
;   521        0651  2      !
;   522        0652  2      ! The desired line number was not found.  Clear LINE_PC and return FALSE
;   523        0653  2      ! as the routine value.
;   524        0654  2      !
;   525        0655  2      IF .FLAG THEN GIVE_LINE_INFO(.LINE_NUM, .STMT_NUM);
;   526        0656  2      .LINE_PC = 0;
;   527        0657  2      RETURN FALSE;
;   528        0658  1      END;




                                   001C 00000            .ENTRY  DBG$LINE_TO_PC_LOOKUP, Save R2,R3,R4                 ; 0406
              54 00000000'    EF    9E 00002             MOVAB   CURRENT_STMT, R4                                     ;
              01         08   AC    D1 00009             CMPL    STMT_NUM, #1                                         ; 0446
                              03    12 0000D             BNEQ    1$                                                   ;
                         08   AC    D4 0000F             CLRL    STMT_NUM                                             ;
          44  A4              01    D0 00012 1$:         MOVL    #1, PCTBL_COUNT                                      ; 0457
              50         0C   AC    D0 00016             MOVL    MC_PTR, R0                                           ; 0458
              51         2C   A0    D0 0001A             MOVL    44(R0), MODPCTBL                                     ;
                              11    13 0001E             BEQL    2$                                                   ; 0459
          2C  A4              61    D0 00020             MOVL    (MODPCTBL), NUM_PC_TBLS                              ; 0460
          30  A4         04   A1    9E 00024             MOVAB   4(MODPCTBL), CURRENT_TABLE                           ; 0461
          F0  A4         04   A1    D0 00029             MOVL    4(MODPCTBL), DST_ENTRY                               ; 0462
                         2C   A4    D5 0002E             TSTL    NUM_PC_TBLS                                          ; 0463
                              03    12 00031 2$:         BNEQ    3$                                                   ;
                            0170    31 00033             BRW     26$                                                  ;
                         FC   A4    D4 00036 3$:         CLRL    CURRENT_LINE                                         ; 0469
              64              01    D0 00039             MOVL    #1, CURRENT_STMT                                     ; 0470
          04  A4              01    D0 0003C             MOVL    #1, CURRENT_INCR                                     ; 0471
                         0C   A4    D4 00040             CLRL    CURRENT_STMT_MODE                                    ; 0472
              50         1C   A0    D0 00043             MOVL    28(R0), R0                                           ; 0473
          F8  A4              50    D0 00047             MOVL    R0, START_PC                                         ;
          08  A4              50    D0 0004B             MOVL    R0, CURRENT_PC                                       ;
          10  A4              02    D0 0004F             MOVL    #2, CURRENT_MARK                                     ; 0474
  F4    A4    F0  A4              02    C1 00053         ADDL3   #2, DST_ENTRY, DPC_ENTRY                             ; 0485
                         3C   A4    D4 00059             CLRL    REPORT_PREV_LINE                                     ; 0486
          40  A4              01    D0 0005C             MOVL    #1, REPORT_PREV_STMT                                 ; 0487
              52         04   AC    7D 00060             MOVQ    LINE_NUM, R2                                         ; 0488
```

```
   34  A4              52  7D 00064       MOVQ    R2, REPORT_NEXT_LINE                  ;
   14  A4      FC      A4  7D 00068 4$:   MOVQ    CURRENT_LINE, PREV_LINE               ; 0497
   24  A4      0C      A4  7D 0006D       MOVQ    CURRENT_STMT_MODE, PREV_STMT_MODE     ; 0500
   1C  A4      04      A4  7D 00072       MOVQ    CURRENT_INCR, PREV_INCR               ; 0499
0000V  CF              00  FB 00077       CALLS   #0, PROC_PC_CMD                       ; 0508
       03              50  E8 0007C       BLBS    R0, 5$                                ;
                     0116  31 0007F       BRW     24$                                   ;
       52      34      A4  D1 00082 5$:   CMPL    REPORT_NEXT_LINE, R2                  ; 0513
                       1E  12 00086       BNEQ    7$                                    ;
       53      38      A4  D1 00088       CMPL    REPORT_NEXT_STMT, R3                  ; 0514
                       18  12 0008C       BNEQ    7$                                    ;
       50      FC      A4  D0 0008E       MOVL    CURRENT_LINE, R0                      ; 0517
       52              50  D1 00092       CMPL    R0, R2                                ;
                       07  14 00095       BGTR    6$                                    ;
                       0D  12 00097       BNEQ    7$                                    ; 0518
       53              64  D1 00099       CMPL    CURRENT_STMT, R3                      ; 0519
                       08  15 0009C       BLEQ    7$                                    ;
   34  A4              50  D0 0009E 6$:   MOVL    R0, REPORT_NEXT_LINE                  ; 0522
   38  A4              64  D0 000A2       MOVL    CURRENT_STMT, REPORT_NEXT_STMT        ; 0523
       50      3C      A4  D0 000A6 7$:   MOVL    REPORT_PREV_LINE, R0                  ; 0535
       52              50  D1 000AA       CMPL    R0, R2                                ;
                       2A  18 000AD       BGEQ    10$                                   ;
       51      14      A4  D0 000AF       MOVL    PREV_LINE, R1                         ; 0538
       52              51  D1 000B3       CMPL    R1, R2                                ;
                       0E  18 000B6       BGEQ    9$                                    ;
       51              50  D1 000B8       CMPL    R0, R1                                ; 0540
                       03  18 000BB       BGEQ    8$                                    ;
       50              51  D0 000BD       MOVL    R1, R0                                ;
   3C  A4              50  D0 000C0 8$:   MOVL    R0, REPORT_PREV_LINE                  ;
                       39  11 000C4       BRB     12$                                   ;
                       37  12 000C6 9$:   BNEQ    12$                                   ; 0543
       53      18      A4  D1 000C8       CMPL    PREV_STMT, R3                         ; 0544
                       31  18 000CC       BGEQ    12$                                   ;
   3C  A4              51  D0 000CE       MOVL    R1, REPORT_PREV_LINE                  ; 0547
   40  A4      18      A4  D0 000D2       MOVL    PREV_STMT, REPORT_PREV_STMT           ; 0548
                       26  11 000D7       BRB     12$                                   ; 0535
                       24  12 000D9 10$:  BNEQ    12$                                   ; 0557
       53      40      A4  D1 000DB       CMPL    REPORT_PREV_STMT, R3                  ; 0558
                       1E  18 000DF       BGEQ    12$                                   ;
       52      14      A4  D1 000E1       CMPL    PREV_LINE, R2                         ; 0561
                       18  12 000E5       BNEQ    12$                                   ;
       53      18      A4  D1 000E7       CMPL    PREV_STMT, R3                         ; 0562
                       12  18 000EB       BGEQ    12$                                   ;
       50      18      A4  D0 000ED       MOVL    PREV_STMT, R0                         ; 0564
   40  A4              50  D1 000F1       CMPL    R0, REPORT_PREV_STMT                  ;
                       04  18 000F5       BGEQ    11$                                   ;
       50      40      A4  D0 000F7       MOVL    REPORT_PREV_STMT, R0                  ;
   40  A4              50  D0 000FB 11$:  MOVL    R0, REPORT_PREV_STMT                  ;
       50      34      A4  D0 000FF 12$:  MOVL    REPORT_NEXT_LINE, R0                  ; 0573
       52              50  D1 00103       CMPL    R0, R2                                ;
                       28  15 00106       BLEQ    15$                                   ;
       51      FC      A4  D0 00108       MOVL    CURRENT_LINE, R1                      ; 0576
       52              51  D1 0010C       CMPL    R1, R2                                ;
                       0E  15 0010F       BLEQ    14$                                   ;
       51              50  D1 00111       CMPL    R0, R1                                ; 0578
                       03  15 00114       BLEQ    13$                                   ;
       50              51  D0 00116       MOVL    R1, R0                                ;
```

```
    34   A4                50 D0 00119 13$:   MOVL    R0, REPORT_NEXT_LINE                      ;
                           35 11 0011D        BRB     17$                                       ;
                           33 12 0011F 14$:   BNEQ    17$                                       ; 0581
         53                64 D1 00121        CMPL    CURRENT_STMT, R3                          ; 0582
                           2E 15 00124        BLEQ    17$                                       ;
    34   A4                51 D0 00126        MOVL    R1, REPORT_NEXT_LINE                      ; 0585
    38   A4                64 D0 0012A        MOVL    CURRENT_STMT, REPORT_NEXT_STMT            ; 0586
                           24 11 0012E        BRB     17$                                       ; 0573
                           22 12 00130 15$:   BNEQ    17$                                       ; 0595
         53        38      A4 D1 00132        CMPL    REPORT_NEXT_STMT, R3                      ; 0596
                           1C 15 00136        BLEQ    17$                                       ;
         52        FC      A4 D1 00138        CMPL    CURRENT_LINE, R2                          ; 0599
                           16 12 0013C        BNEQ    17$                                       ;
         53                64 D1 0013E        CMPL    CURRENT_STMT, R3                          ; 0600
                           11 15 00141        BLEQ    17$                                       ;
         50                64 D0 00143        MOVL    CURRENT_STMT, R0                          ; 0602
    38   A4                50 D1 00146        CMPL    R0, REPORT_NEXT_STMT                      ;
                           04 15 0014A        BLEQ    16$                                       ;
         50        38      A4 D0 0014C        MOVL    REPORT_NEXT_STMT, R0                      ;
    40   A4                50 D0 00150 16$:   MOVL    R0, REPORT_PREV_STMT                      ;
         52        FC      A4 D1 00154 17$:   CMPL    CURRENT_LINE, R2                          ; 0618
                           03 13 00158        BEQL    19$                                       ;
                         FF0B 31 0015A 18$:   BRW     4$                                        ;
         51                64 D0 0015D 19$:   MOVL    CURRENT_STMT, R1                          ; 0621
                           03 14 00160        BGTR    20$                                       ;
         51                01 D0 00162        MOVL    #1, R1                                    ;
         50                53 D0 00165 20$:   MOVL    R3, R0                                    ;
                           03 14 00168        BGTR    21$                                       ;
         50                01 D0 0016A        MOVL    #1, R0                                    ;
         50                51 D1 0016D 21$:   CMPL    R1, R0                                    ;
                           14 12 00170        BNEQ    22$                                       ;
    10   BC        08      A4 D0 00172        MOVL    CURRENT_PC, @LINE_PC                      ; 0624
                   14      AC DD 00177        PUSHL   LINE_END                                  ; 0625
 0000V   CF                01 FB 0017A        CALLS   #1, FIND_EOL                              ;
         09                50 E9 0017F        BLBC    R0, 23$                                   ;
         50                01 D0 00182        MOVL    #1, R0                                    ; 0632
                              04 00185        RET                                               ;
                           D2 15 00186 22$:   BLEQ    18$                                       ; 0637
                   10      BC D4 00188        CLRL    @LINE_PC                                  ; 0640
         17        18      AC E9 0018B 23$:   BLBC    FLAG, 26$                                 ; 0641
                           0C BB 0018F        PUSHR   #^M<R2,R3>                                ;
 0000V   CF                02 FB 00191        CALLS   #2, GIVE_LINE_INFO                        ;
                           0E 11 00196        BRB     26$                                       ; 0642
         07        18      AC E9 00198 24$:   BLBC    FLAG, 25$                                 ; 0655
                           0C BB 0019C        PUSHR   #^M<R2,R3>                                ;
 0000V   CF                02 FB 0019E        CALLS   #2, GIVE_LINE_INFO                        ;
                   10      BC D4 001A3 25$:   CLRL    @LINE_PC                                  ; 0656
                           50 D4 001A6 26$:   CLRL    R0                                        ; 0658
                              04 001A8        RET                                               ;

; Routine Size:  425 bytes,    Routine Base:  DBG$CODE + 009A
```

```
1 GLOBAL ROUTINE dbg$pc_to_line_lookup (match_pc_ptr,line_no_ptr, stmt_no_ptr,
1                                         line_start,  line_end,    mod_symid) =
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine matches an address to a line number.
1 !       We need to do this in several situations:
1 !
1 !       1. When stepping by line, to determine when to stop stepping. (DBGEVENT)
1 !       2. When symbolizing a code address to put out "%LINE XX" (DBGSYMBLZ)
1 !       3. Putting out the SHOW CALLS display (DBGTBK)
1 !       4. Finding the start of the line for "EX/INS ^" (DBGLEVEL3)
1 !       5. Source display, as in EX/SOURCE .PC (DBGSOURCE)
1 !
1 !       The line number (and statement number, for BASIC) is returned.
1 !       Also returned are: the start and end address of the line,
1 !       and a pointer to the module RST entry for the module containing
1 !       the given address.
1 !
1 !       Each PC correlation record that exists for the module
1 !       is sequentially analyzed until the desired address is seen.
1 !
1 !       This routine is actually just a cover routine for DBG$PC_TO_LINE,
1 !       where the real work is done.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       match_pc_ptr     - the address to be matched.
1 !       line_no_ptr      - an output parameter for the line number.
1 !       stmt_no_ptr      - an output parameter for the statement number.
1 !       line_start       - an output parameter for the start pc of the
1 !                          selected line/stmt.
1 !       line_end         - an output parameter for the end pc of the
1 !                          selected line/stmt.
1 !       mod_symid        - An in/out parameter, as follows:
1 !
1 !                          If the caller has a SYMID for a block, routine,
1 !                          or module which contains the given address, then
1 !                          this symid can be passed in here. This saves
1 !                          a search of the Static Address Table.
1 !                          If the caller
1 !                          does not have a symid, then zero is passed in.
1 !                          Note that these are passed in with an extra level
1 !                          of indirection, e.g.,
1 !                          SYMID = 0;
1 !                          STATUS = DBG$PC_TO_LINE_LOOKUP(.ADDRESS,...,SYMID);
1 !
1 !                          In either case, this parameter is filled in with
1 !                          the address of the module containing MATCH_PC_PTR.
1 !
1 ! ROUTINE VALUE:
1 !
1 !       This routine can return four values: 0, 1, 2, or 3.
1 !       Most of the callers just test the result for
1 !       TRUE (meaning a match was found), or FALSE (meaning a match
1 !       was not found). So for these callers, 0 and 2 are the same,
1 !       and 1 and 3 are the same.
```

```
1 !
1 !         DBGEVENT needs more detailed information than just whether
1 !         a match was found, in order to decide whether to continue
1 !         stepping. It needs to know why a match was not found, or
1 !         if one was found, whether or not it was an exact match.
1 !         So for the DBGEVENT call, we return the following:
1 !
1 !         0         - If no match can be made because pc/line tables are
1 !                     not available for the given address. This may occur
1 !                     because the module containing the address was not
1 !                     set or was compiled /NODEBUG, or because the address
1 !                     is in system space or in an RTL shareable image.
1 !         1         - If a line number/stmt number was found, and we
1 !                     have an exact match to that line number.
1 !         2         - If there are pc/line tables available for the
1 !                     module containing the given address, but no match
1 !                     was found. This occurs if the address is not within
1 !                     any line in the module. The use of the "TERM" record
1 !                     in PC/LINE tables terminates an address range for
1 !                     a line without starting a new line, and this can
1 !                     give rise to addresses without line numbers.
1 !         3         - If there is a line number associated with the address, but
1 !                     it is not an exact match.
1 !
2       BEGIN
2       LOCAL
2           rstptr: REF rst$entry,    ! Module RST pointer
2           status;                   ! Return Status
2
2
2       ! If we do not know an RST entry for a program unit
2       ! containing the given address, we'll look
2       ! it up through the Program-level SAT.
2       ! If we already have the information
2       ! (passed in from the caller) then just set it up.
2       !
2       IF ..mod_symid EQL 0
2       THEN
3           BEGIN
3           status = dbg$pc_to_symid(.match_pc_ptr, rstptr);
3
3
3           ! If PC_TO_SYMID failed, then we do not have a module containing
3           ! the address in our module chain. Thus, return zero.
3           !
3           IF NOT .status THEN RETURN 0;
3           END
3
2       ELSE
2           rstptr = ..mod_symid;
2
2
2       ! Go upscope to the module level, just in case a caller passed in
2       ! a routine or block RST entry.
2       !
2       WHILE (.rstptr[rst$b_kind] NEQ rst$k_module) DO
2           rstptr = .rstptr[rst$l_upscopeptr];
```

```
;  644        0773  2
;  645        0774  2
;  646        0775  2     ! Set the return module RST.
;  647        0776  2     !
;  648        0777  2     .mod_symid = .rstptr;
;  649        0778  2
;  650        0779  2
;  651        0780  2     ! Now call the routine to do the real work. Pass along the three
;  652        0781  2     ! output parameters LINE_NO_PTR, STMT_NO_PTR, and LINE_START,
;  653        0782  2     ! to be filled in by DBG$PC_TO_LINE.
;  654        0783  2     !
;  655        0784  2     status = dbg$pc_to_line(.match_pc_ptr, .rstptr[rst$l_modpctbl],
;  656        0785  2                             .rstptr[rst$l_pctbl_base],
;  657        0786  2                             .line_no_ptr,  .stmt_no_ptr, .line_start);
;  658        0787  2
;  659        0788  2
;  660        0789  2     ! We get the return code from DBG$PC_TO_LINE. Here we check
;  661        0790  2     ! for the PC being an exact match. If not, we change the "1"
;  662        0791  2     ! return status to a "3" to indicate this. We also fill in the
;  663        0792  2     ! LINE_END output parameter, using the OWN variable CURRENT_PC
;  664        0793  2     ! that gets set in the processing of PC/LINE records.
;  665        0794  2     !
;  666        0795  2     IF .status EQL 1
;  667        0796  2     THEN
;  668        0797  3         BEGIN
;  669        0798  3         .line_end = .current_pc - 1;
;  670        0799  3         IF ..line_start NEQA .match_pc_ptr
;  671        0800  3         THEN
;  672        0801  3             status = 3;         ! not exact match.
;  673        0802  2         END;
;  674        0803  2
;  675        0804  2     RETURN .status;
;  676        0805  1     END;



                                       0000 00000        .ENTRY  DBG$PC_TO_LINE_LOOKUP, Save nothing          ; 0659
                  5E              04     C2 00002        SUBL2   #4, SP                                        ;
                            18    BC     D5 00005        TSTL    @MOD_SYMID                                    ; 0752
                                  11     12 00008        BNEQ    1$                                            ;
                                  5E     DD 0000A        PUSHL   SP                                            ; 0755
                            04    AC     DD 0000C        PUSHL   MATCH_PC_PTR                                  ;
        00000000G  00             02     FB 0000F        CALLS   #2, DBG$PC_TO_SYMID                           ;
                   06             50     E8 00016        BLBS    STATUS, 2$                                    ; 0761
                                  48     11 00019        BRB     4$                                            ;
                   6E       18    BC     D0 0001B  1$:   MOVL    @MOD_SYMID, RSTPTR                            ; 0765
                   51             6E     D0 0001F  2$:   MOVL    RSTPTR, R1                                    ; 0771
                   01       14    A1     91 00022        CMPB    20(R1), #1                                    ;
                                  06     13 00026        BEQL    3$                                            ;
                   6E       10    A1     D0 00028        MOVL    16(R1), RSTPTR                                ; 0772
                                  F1     11 0002C        BRB     2$                                            ;
                   51             6E     D0 0002E  3$:   MOVL    RSTPTR, R1                                    ; 0777
              18   BC             51     D0 00031        MOVL    R1, @MOD_SYMID                                ;
                   7E       0C    AC     7D 00035        MOVQ    STMT_NO_PTR, -(SP)                            ; 0786
                            08    AC     DD 00039        PUSHL   LINE_NO_PTR                                   ;
```

```
                                  1C    A1   DD 0003C          PUSHL   28(R1)                                    : 0785
                                  2C    A1   DD 0003F          PUSHL   44(R1)                                    : 0784
                                  04    AC   DD 00042          PUSHL   MATCH_PC_PTR                              :
                    FD73    CF          06   FB 00045          CALLS   #6, DBG$PC_TO_LINE                        :
                            01          50   D1 0004A          CMPL    STATUS, #1                                : 0795
                                        16   12 0004D          BNEQ    5$                                        :
             14  BC 00000000'   EF          01   C3 0004F          SUBL3   #1, CURRENT_PC, @LINE_END                 : 0798
                      04    AC    10    BC   D1 00058          CMPL    @LINE_START, MATCH_PC_PTR                 : 0799
                                        06   13 0005D          BEQL    5$                                        :
                            50          03   D0 0005F          MOVL    #3, STATUS                                : 0801
                                             04 00062          RET                                               : 0804
                                        50   D4 00063 4$:      CLRL    R0                                        : 0805
                                             04 00065 5$:      RET                                               :

; Routine Size:  102 bytes,    Routine Base:  DBG$CODE + 0243
```

```
1 ROUTINE PROC_PC_CMD =
1 !++
1 ! Functional description:
1 !
1 !       This routine processes PC correlation commands until a
1 !       delta-Pc command is seen. The delta-PC command is also processed.
1 !       Then this routine returns with all the contents of the
1 !       parameter pointers updated appropriately.
1 !
1 !       This routine moves from PC record to PC record as necessary. If
1 !       no more records are seen, this routine returns false. If
1 !       an error is seen in a PC correlation record, then this
1 !       routine sets the contents of line_ptr to zero and
1 !       returns false.
1 !
1 ! Inputs:
1 !
1 ! Implicit inputs:
1 !       None
1 !
1 ! Implicit outputs:
1 !       the contents of the line pointer, the increment pointer, the
1 !       statement pointer, the next_pc pointer, dpc_entry, and possible
1 !       dst_entry are updated to new values.
1 !
1 ! Routine value:
1 !       TRUE if all goes well, otherwise FALSE.
1 !
1 ! Side effects:
1 !       More of the correlation records for this routine are read.
1 !--
1
2       BEGIN
2
2       REPEAT
3               BEGIN
3
3
3               ! See whether the current record is exhausted. If
3               ! so, get a new record. If none are available,
3               ! return FALSE. Otherwise, set dpc_entry to point to
3               ! the address of the third byte of the correlation record.
3               !
4               IF dpc_entry[current_byte] GTR (.dst_entry[dst$b_length] +
4                               dst_entry[dst$b_length])
3               THEN
4                       BEGIN
4                       PCTBL_COUNT = .PCTBL_COUNT + 1;
4                       IF .PCTBL_COUNT GTR .NUM_PC_TBLS THEN RETURN FALSE;
4                       current_table = .current_table + 4;
4                       dst_entry = ..current_table;
4                       dpc_entry = dst_entry [dst$b_vflags];
3                       END;
3
3
3               ! Now process each command, either PC correlation or
3               ! delta-PC one at a time. Once a delta-PC command is
```

```
        ! processed, control returns from this routine to its
        ! caller.
        !
        CASE .dpc_entry [current_byte] FROM 1 TO dst$k_pccor_high OF
                SET


                ! Read the next two bytes as an unsigned word
                ! representing a delta-PC value.  Update the next_pc
                ! and update the dpc_entry address.
                !
                [dst$k_delta_pc_w]:
                        BEGIN
                        IF .current_stmt_mode
                        THEN
                                current_stmt = .current_stmt + 1
                        ELSE
                                current_line = .current_line +
                                                        .current_incr;

                        current_mark = line_open;
                        current_pc = .current_pc +
                                        .dpc_entry [next_uns_word];
                        dpc_entry = dpc_entry [add_three_bytes];
                        RETURN TRUE;
                        END;


                ! Read the next four bytes as an unsigned longword
                ! representing a delta-PC value.  Update the next_pc
                ! and update the dpc_entry address.
                !
                [dst$k_delta_pc_l]:
                        BEGIN
                        IF .current_stmt_mode
                        THEN
                                current_stmt = .current_stmt + 1
                        ELSE
                                current_line = .current_line +
                                                        .current_incr;

                        current_mark = line_open;
                        current_pc = .current_pc +
                                        .dpc_entry [next_uns_long];
                        dpc_entry = dpc_entry [add_five_bytes];
                        RETURN TRUE;
                        END;


                ! Increase the current line number by the value
                ! contained in the next unsigned byte.
                !
                [dst$k_incr_linum]:
                        BEGIN
                        current_line = .current_line + .dpc_entry [next_uns_byte];
                        IF .current_stmt_mode THEN current_stmt = 1;
                        dpc_entry = dpc_entry [add_two_bytes];
```

```
3                       END;
3
3               ! Increase the current line number by the value
3               ! contained in the next unsigned word.
3               !
3               [dst$k_incr_linum_w]:
4                       BEGIN
4                       IF .current_stmt_mode THEN current_stmt = 1;
4                       current_line = .current_line + .dpc_entry [next_uns_word];
4                       dpc_entry = dpc_entry [add_three_bytes];
3                       END;
3
3               ! Increase the current line number by the value
3               ! contained in the next unsigned longword.
3               !
3               [dst$k_incr_linum_l]:
4                       BEGIN
4                       IF .current_stmt_mode THEN current_stmt = 1;
4                       current_line = .current_line + .dpc_entry [next_uns_long];
4                       dpc_entry = dpc_entry [add_five_bytes];
3                       END;
3
3               ! Change the line increment from its present value to
3               ! the value contained in the next unsigned byte.
3               !
3               [dst$k_set_linum_incr]:
4                       BEGIN
4                       IF .current_stmt_mode THEN current_stmt = 1;
4                       current_incr = .dpc_entry [next_uns_byte];
4                       dpc_entry = dpc_entry [add_two_bytes];
3                       END;
3
3               ! Change the line increment from its present value to
3               ! the value contained in the next word.
3               !
3               [dst$k_set_linum_incr_w]:
4                       BEGIN
4                       IF .current_stmt_mode THEN current_stmt = 1;
4                       current_incr = .dpc_entry [next_uns_word];
4                       dpc_entry = dpc_entry [add_three_bytes];
3                       END;
3
3
3               ! Revert to a line increment of value 1.
3               !
3               [dst$k_reset_linum_incr]:
4                       BEGIN
4                       IF .current_stmt_mode THEN current_stmt = 1;
4                       current_incr = 1;
4                       dpc_entry = dpc_entry [add_one_byte];
3                       END;
3
3               [dst$k_beg_stmt_mode]:
```

```
4                       BEGIN
4                       IF .current_mark NEQ line_open
4                       THEN
4                               SIGNAL(dbg$_invdstrec);
4
4                       current_stmt = 1;
4                       current_stmt_mode = TRUE;
4                       dpc_entry = dpc_entry[add_one_byte];
3                       END;
3
3               [dst$k_end_stmt_mode]:
4                       BEGIN
4                       current_stmt = 1;
4                       current_stmt_mode = FALSE;
4                       dpc_entry = dpc_entry[add_one_byte];
3                       END;
3
3               [dst$k_set_linum_b]:
4                       BEGIN
4                       current_line = .dpc_entry[next_uns_byte];
4                       dpc_entry = dpc_entry[add_two_bytes];
3                       END;
3
3               [dst$k_set_linum]:
4                       BEGIN
4                       current_line = .dpc_entry[next_uns_word];
4                       dpc_entry = dpc_entry[add_three_bytes];
3                       END;
3
3               [dst$k_set_linum_l]:
4                       BEGIN
4                       current_line = .dpc_entry[next_uns_long];
4                       dpc_entry = dpc_entry[add_five_bytes];
3                       END;
3
3               [dst$k_set_stmtnum]:
4                       BEGIN
4                       current_stmt = .dpc_entry[next_uns_word];
4                       dpc_entry = dpc_entry[add_three_bytes];
3                       END;
3
3               [dst$k_set_pc]:
4                       BEGIN
4                       IF .current_mark NEQ line_closed
4                       THEN
4                               SIGNAL (dbg$_invdstrec);
4
4                       current_pc = .start_pc +
4                                       .dpc_entry[next_uns_byte];
4                       dpc_entry = dpc_entry[add_two_bytes];
3                       END;
3
3               [dst$k_set_pc_w]:
4                       BEGIN
4                       IF .current_mark NEQ line_closed
4                       THEN
4                               SIGNAL (dbg$_invdstrec);
```

```
                current_pc = .start_pc +
                                .dpc_entry[next_uns_word];
                dpc_entry = dpc_entry[add_three_bytes];
                END;

        [dst$k_set_pc_l]:
                BEGIN
                IF .current_mark NEQ line_closed
                THEN
                        SIGNAL (dbg$_invdstrec);

                current_pc = .start_pc +
                                .dpc_entry[next_uns_long];
                dpc_entry = dpc_entry[add_five_bytes];
                END;


        ! Set the current PC value to an absolute address.
        !
        [DST$K_SET_ABS_PC]:
                BEGIN
                IF .CURRENT_MARK NEQ LINE_CLOSED
                THEN
                    SIGNAL(DBG$_INVDSTREC);

                CURRENT_PC = .DPC_ENTRY[NEXT_UNS_LONG];
                DPC_ENTRY = DPC_ENTRY[ADD_FIVE_BYTES];
                END;

        [dst$k_term]:
                BEGIN
                current_pc = .current_pc +
                                .dpc_entry[next_uns_byte];
                current_mark = line_closed;
                dpc_entry = dpc_entry[add_two_bytes];
                RETURN TRUE;
                END;

        [dst$k_term_w]:
                BEGIN
                current_pc = .current_pc +
                                .dpc_entry[next_uns_word];
                current_mark = line_closed;
                dpc_entry = dpc_entry[add_three_bytes];
                RETURN TRUE;
                END;


        [dst$k_term_l]:
                BEGIN
                current_pc = .current_pc +
                                .dpc_entry[next_uns_long];
                current_mark = line_closed;
                dpc_entry = dpc_entry[add_five_bytes];
                RETURN TRUE;
                END;
```

```
3
3
3                                        ! This is a standard delta_PC command if the value is
3                                        ! less than or equal to zero. Otherwise it is an error.
3                                        ! If okay, set next_pc value, update the dpc_entry,
3                                        ! and return with success.
3                                        !
3                                        [OUTRANGE]:
4                                                BEGIN
4                                                IF .dpc_entry [current_byte] LSS
4                                                                        dst$k_delta_pc_low
4                                                OR .dpc_entry[current_byte] GTR
4                                                                        dst$k_delta_pc_high
4                                                THEN
4                                                        SIGNAL (dbg$_invdstrec);
4
4                                                IF .current_stmt_mode
4                                                THEN
4                                                        current_stmt = .current_stmt + 1
4                                                ELSE
4                                                        current_line = .current_line +
4                                                                                .current_incr;
4
4                                                current_pc = .current_pc -
4                                                                .dpc_entry [current_byte];
4                                                current_mark = line_open;
4                                                dpc_entry = dpc_entry [add_one_byte];
4                                                RETURN TRUE;
3                                                END;
3
3                                        TES;
2                                END;
2
2                        RETURN 0;
1                        END;
```

```
                                    001C 00000 PROC_PC_CMD:
                                                          .WORD    Save R2,R3,R4                          ; 0806
                 54 00000000G   00   9E 00002             MOVAB    LIB$SIGNAL, R4                         ;
                 53 00000000'   EF   9E 00009             MOVAB    DPC_ENTRY, R3                          ;
                 50         FC  B3   9A 00010 1$:         MOVZBL   @DST_ENTRY, R0                         ; 0850
                 50         FC  A3   C0 00014             ADDL2    DST_ENTRY, R0                          ;
                 50             63   D1 00018             CMPL     DPC_ENTRY, R0                          ; 0849
                                1B   15 0001B             BLEQ     3$                                     ;
                            50  A3   D6 0001D             INCL     PCTBL_COUNT                            ; 0853
            38    A3        50  A3   D1 00020             CMPL     PCTBL_COUNT, NUM_PC_TBLS               ; 0854
                                03   15 00025             BLEQ     2$                                     ;
                              01ED   31 00027             BRW      56$                                    ;
            3C    A3            04   C0 0002A 2$:         ADDL2    #4, CURRENT_TABLE                      ; 0855
            FC    A3        3C  B3   D0 0002E             MOVL     @CURRENT_TABLE, DST_ENTRY              ; 0856
      63    FC    A3            02   C1 00033             ADDL3    #2, DST_ENTRY, DPC_ENTRY               ; 0857
                  52            63   D0 00038 3$:         MOVL     DPC_ENTRY, R2                          ; 0866
      14          01            62   8F 0003B             CASEB    (R2), #1, #20                          ;
```

```
    00C2        00A1        008F        0055      0003F 4$:     .WORD   8$-4$,-                          ;
    0107        00EE        00E0        00D1      00047                 16$-4$,-                         ;
    0170        014F        012E        0119      0004F                 17$-4$,-                         ;
    018B        01B9        01A8        0127      00057                 21$-4$,-                         ;
    0120        0112        00B3        0075      0005F                 23$-4$,-                         ;
                                        01C8      00067                 25$-4$,-                         ;
                                                                        27$-4$,-                         ;
                                                                        29$-4$,-                         ;
                                                                        33$-4$,-                         ;
                                                                        38$-4$,-                         ;
                                                                        41$-4$,-                         ;
                                                                        45$-4$,-                         ;
                                                                        36$-4$,-                         ;
                                                                        51$-4$,-                         ;
                                                                        52$-4$,-                         ;
                                                                        47$-4$,-                         ;
                                                                        13$-4$,-                         ;
                                                                        19$-4$,-                         ;
                                                                        31$-4$,-                         ;
                                                                        34$-4$,-                         ;
                                                                        53$-4$                           ;
                                           62   95 00069        TSTB    (R2)                             ; 1102
                                           09   15 0006B        BLEQ    5$                               ;
                              0002832A     8F   DD 0006D        PUSHL   #164650                          ; 1105
                        64                 01   FB 00073        CALLS   #1, LIB$SIGNAL                   ;
                        05          18     A3   E9 00076 5$:    BLBC    CURRENT_STMT_MODE, 6$            ; 1107
                                    0C     A3   D6 0007A        INCL    CURRENT_STMT                     ; 1109
                                           05   11 0007D        BRB     7$                               ;
                  08    A3          10     A3   C0 0007F 6$:    ADDL2   CURRENT_INCR, CURRENT_LINE       ; 1112
                        50          00     B3   98 00084 7$:    CVTBL   @DPC_ENTRY, R0                   ; 1115
                  14    A3                 50   C2 00088        SUBL2   R0, CURRENT_PC                   ;
                  1C    A3                 01   D0 0008C        MOVL    #1, CURRENT_MARK                 ; 1116
                                           63   D6 00090        INCL    DPC_ENTRY                        ; 1117
                                           1D   11 00092        BRB     12$                              ; 1118
                        05          18     A3   E9 00094 8$:    BLBC    CURRENT_STMT_MODE, 9$            ; 0876
                                    0C     A3   D6 00098        INCL    CURRENT_STMT                     ; 0878
                                           05   11 0009B        BRB     10$                              ;
                  08    A3          10     A3   C0 0009D 9$:    ADDL2   CURRENT_INCR, CURRENT_LINE       ; 0881
                  1C    A3                 01   D0 000A2 10$:   MOVL    #1, CURRENT_MARK                 ; 0883
                        50          01     A2   3C 000A6        MOVZWL  1(R2), R0                        ; 0885
                  14    A3                 50   C0 000AA        ADDL2   R0, CURRENT_PC                   ;
                        63                 03   C0 000AE 11$:   ADDL2   #3, DPC_ENTRY                    ; 0886
                                         015F   31 000B1 12$:   BRW     55$                              ; 0887
                        05          18     A3   E9 000B4 13$:   BLBC    CURRENT_STMT_MODE, 14$           ; 0897
                                    0C     A3   D6 000B8        INCL    CURRENT_STMT                     ; 0899
                                           05   11 000BB        BRB     15$                              ;
                  08    A3          10     A3   C0 000BD 14$:   ADDL2   CURRENT_INCR, CURRENT_LINE       ; 0902
                  1C    A3                 01   D0 000C2 15$:   MOVL    #1, CURRENT_MARK                 ; 0904
                  14    A3          01     A2   C0 000C6        ADDL2   1(R2), CURRENT_PC                ; 0906
                                         0142   31 000CB        BRW     54$                              ; 0907
                        50          01     A2   9A 000CE 16$:   MOVZBL  1(R2), R0                        ; 0917
                  08    A3                 50   C0 000D2        ADDL2   R0, CURRENT_LINE                 ;
                        7C          18     A3   E9 000D6        BLBC    CURRENT_STMT_MODE, 32$           ; 0918
                  0C    A3                 01   D0 000DA        MOVL    #1, CURRENT_STMT                 ;
                                           76   11 000DE        BRB     32$                              ; 0919
                        04          18     A3   E9 000E0 17$:   BLBC    CURRENT_STMT_MODE, 18$           ; 0928
                  0C    A3                 01   D0 000E4        MOVL    #1, CURRENT_STMT                 ;
```

```
                                                                          J  1
DBGDPC                                                     16-Sep-1984 00:22:28    VAX-11 Bliss-32 V4.0-742                  Page  27
V04-000                                                    14-Sep-1984 12:16:51    DISK$VMSMASTER:[DEBUG.SRC]DBGDPC.B32;1     (6)

                  50        01  A2  3C 000E8 18$:    MOVZWL  1(R2), R0                                          ; 0929
            08    A3            50  C0 000EC         ADDL2   R0, CURRENT_LINE                                   ;
                                79  11 000F0         BRB     37$                                                ; 0930
                  04        18  A3  E9 000F2 19$:    BLBC    CURRENT_STMT_MODE, 20$                             ; 0939
            0C    A3            01  D0 000F6         MOVL    #1, CURRENT_STMT                                   ;
            08    A3        01  A2  C0 000FA 20$:    ADDL2   1(R2), CURRENT_LINE                                ; 0940
                                63  11 000FF         BRB     35$                                                ; 0941
                  04        18  A3  E9 00101 21$:    BLBC    CURRENT_STMT_MODE, 22$                             ; 0950
            0C    A3            01  D0 00105         MOVL    #1, CURRENT_STMT                                   ;
            10    A3        01  A2  9A 00109 22$:    MOVZBL  1(R2), CURRENT_INCR                                ; 0951
                                79  11 0010E         BRB     40$                                                ; 0952
                  04        18  A3  E9 00110 23$:    BLBC    CURRENT_STMT_MODE, 24$                             ; 0961
            0C    A3            01  D0 00114         MOVL    #1, CURRENT_STMT                                   ;
            10    A3        01  A2  3C 00118 24$:    MOVZWL  1(R2), CURRENT_INCR                                ; 0962
                                4C  11 0011D         BRB     37$                                                ; 0963
                  04        18  A3  E9 0011F 25$:    BLBC    CURRENT_STMT_MODE, 26$                             ; 0971
            0C    A3            01  D0 00123         MOVL    #1, CURRENT_STMT                                   ;
            10    A3            01  D0 00127 26$:    MOVL    #1, CURRENT_INCR                                   ; 0972
                                20  11 0012B         BRB     30$                                                ; 0973
                  01        1C  A3  D1 0012D 27$:    CMPL    CURRENT_MARK, #1                                   ; 0978
                                09  13 00131         BEQL    28$                                                ;
                      0002832A  8F  DD 00133         PUSHL   #164650                                            ; 0980
                  64            01  FB 00139         CALLS   #1, LIB$SIGNAL                                     ;
            0C    A3            01  D0 0013C 28$:    MOVL    #1, CURRENT_STMT                                   ; 0982
            18    A3            01  D0 00140         MOVL    #1, CURRENT_STMT_MODE                              ; 0983
                                07  11 00144         BRB     30$                                                ; 0984
            0C    A3            01  D0 00146 29$:    MOVL    #1, CURRENT_STMT                                   ; 0989
                            18  A3  D4 0014A         CLRL    CURRENT_STMT_MODE                                  ; 0990
                                63  D6 0014D 30$:    INCL    DPC_ENTRY                                          ; 0991
                                5C  11 0014F         BRB     44$                                                ; 0866
            08    A3        01  A2  9A 00151 31$:    MOVZBL  1(R2), CURRENT_LINE                                ; 0996
                                31  11 00156 32$:    BRB     40$                                                ; 0997
            08    A3        01  A2  3C 00158 33$:    MOVZWL  1(R2), CURRENT_LINE                                ; 1002
                                4B  11 0015D         BRB     43$                                                ; 1003
            08    A3        01  A2  D0 0015F 34$:    MOVL    1(R2), CURRENT_LINE                                ; 1008
                                7B  11 00164 35$:    BRB     49$                                                ; 1009
            0C    A3        01  A2  3C 00166 36$:    MOVZWL  1(R2), CURRENT_STMT                                ; 1014
                                3D  11 0016B 37$:    BRB     43$                                                ; 1015
                  02        1C  A3  D1 0016D 38$:    CMPL    CURRENT_MARK, #2                                   ; 1020
                                09  13 00171         BEQL    39$                                                ;
                      0002832A  8F  DD 00173         PUSHL   #164650                                            ; 1022
                  64            01  FB 00179         CALLS   #1, LIB$SIGNAL                                     ;
                  50            63  D0 0017C 39$:    MOVL    DPC_ENTRY, R0                                      ; 1025
                  51        01  A0  9A 0017F         MOVZBL  1(R0), R1                                          ;
            14    A3     04 B341    9E 00183         MOVAB   @START_PC[R1], CURRENT_PC                          ;
                  63            02  C0 00189 40$:    ADDL2   #2, DPC_ENTRY                                      ; 1026
                                56  11 0018C         BRB     50$                                                ; 0866
                  02        1C  A3  D1 0018E 41$:    CMPL    CURRENT_MARK, #2                                   ; 1031
                                09  13 00192         BEQL    42$                                                ;
                      0002832A  8F  DD 00194         PUSHL   #164650                                            ; 1033
                  64            01  FB 0019A         CALLS   #1, LIB$SIGNAL                                     ;
                  50            63  D0 0019D 42$:    MOVL    DPC_ENTRY, R0                                      ; 1036
                  51        01  A0  3C 001A0         MOVZWL  1(R0), R1                                          ;
            14    A3     04 B341    9E 001A4         MOVAB   @START_PC[R1], CURRENT_PC                          ;
                  63            03  C0 001AA 43$:    ADDL2   #3, DPC_ENTRY                                      ; 1037
                                35  11 001AD 44$:    BRB     50$                                                ; 0866
                  02        1C  A3  D1 001AF 45$:    CMPL    CURRENT_MARK, #2                                   ; 1042
```

```
                                       09  13 001B3          BEQL     46$                                      ;
                          0002832A     8F  DD 001B5          PUSHL    #164650                                  ; 1044
                     64                01  FB 001BB          CALLS    #1, LIB$SIGNAL                           ;
                     50                63  D0 001BE  46$:    MOVL     DPC_ENTRY, R0                            ; 1047
      14   A3    04  A3        01      A0  C1 001C1          ADDL3    1(R0), START_PC, CURRENT_PC              ;
                                       17  11 001C8          BRB      49$                                      ; 1048
                     02        1C      A3  D1 001CA  47$:    CMPL     CURRENT_MARK, #2                         ; 1056
                                       09  13 001CE          BEQL     48$                                      ;
                          0002832A     8F  DD 001D0          PUSHL    #164650                                  ; 1058
                     64                01  FB 001D6          CALLS    #1, LIB$SIGNAL                           ;
                     50                63  D0 001D9  48$:    MOVL     DPC_ENTRY, R0                            ; 1060
                14   A3        01      A0  D0 001DC          MOVL     1(R0), CURRENT_PC                        ;
                     63                05  C0 001E1  49$:    ADDL2    #5, DPC_ENTRY                            ; 1061
                                     FE29  31 001E4  50$:    BRW      1$                                       ; 0866
                     50        01      A2  9A 001E7  51$:    MOVZBL   1(R2), R0                                ; 1067
                14   A3                50  C0 001EB          ADDL2    R0, CURRENT_PC                           ;
                1C   A3                02  D0 001EF          MOVL     #2, CURRENT_MARK                         ; 1068
                     63                02  C0 001F3          ADDL2    #2, DPC_ENTRY                            ; 1069
                                       1B  11 001F6          BRB      55$                                      ; 1070
                     50        01      A2  3C 001F8  52$:    MOVZWL   1(R2), R0                                ; 1076
                14   A3                50  C0 001FC          ADDL2    R0, CURRENT_PC                           ;
                1C   A3                02  D0 00200          MOVL     #2, CURRENT_MARK                         ; 1077
                                     FEA7  31 00204          BRW      11$                                      ; 1078
                14   A3        01      A2  C0 00207  53$:    ADDL2    1(R2), CURRENT_PC                        ; 1086
                1C   A3                02  D0 0020C          MOVL     #2, CURRENT_MARK                         ; 1087
                     63                05  C0 00210  54$:    ADDL2    #5, DPC_ENTRY                            ; 1088
                     50                01  D0 00213  55$:    MOVL     #1, R0                                   ; 1089
                                           04 00216          RET                                               ;
                                       50  D4 00217  56$:    CLRL     R0                                       ; 1125
                                           04 00219          RET                                               ;

; Routine Size:  538 bytes,    Routine Base:  DBG$CODE + 02A9
```

```
1 ROUTINE FIND_EOL(LINE_END) =
1 !++
1 ! Functional description:
1 !       This routine processes PC correlation commands until
1 !       an end of line is found.
1 !
1 ! Inputs:
1 !       line_end        - a copy-back pointer for the value of the end-of-line
1 !
1 ! Implicit inputs:
1 !       None
1 !
1 ! Implicit outputs:
1 !       the contents of the line pointer, the increment pointer, the
1 !       statement pointer, the next_pc pointer, dpc_entry, and possible
1 !       dst_entry are updated to new values.
1 !
1 ! Routine value:
1 !       TRUE if all goes well, otherwise FALSE.
1 !
1 ! Side effects:
1 !       More of the correlation records for this routine are read.
1 !--
1
2       BEGIN
2
2       REPEAT
3               BEGIN
3
3
3               ! See whether the current record is exhausted. If
3               ! so, get a new record. If none are available,
3               ! return FALSE. Otherwise, set dpc_entry to point to
3               ! the address of the third byte of the correlation record.
3               !
4               IF dpc_entry[current_byte] GTR (.dst_entry[dst$b_length] +
4                               dst_entry[dst$b_length])
3               THEN
4                       BEGIN
4                       PCTBL_COUNT = .PCTBL_COUNT + 1;
4                       IF .PCTBL_COUNT GTR .NUM_PC_TBLS THEN RETURN FALSE;
4                       current_table = .current_table + 4;
4                       dst_entry = ..current_table;
4                       dpc_entry = dst_entry [dst$b_vflags];
3                       END;
3
3
3               ! Now process each command, either PC correlation or
3               ! delta-PC one at a time.
3               !
3               CASE .dpc_entry [current_byte] FROM 1 TO dst$k_pccor_high OF
3                       SET
3                       [dst$k_delta_pc_w]:
4                               BEGIN
4                               .line_end = (.current_pc - 1) +
4                                               .dpc_entry [next_uns_word];
4                               RETURN TRUE;
```

```
3                       END;
3
3               [dst$k_delta_pc_l]:
4                       BEGIN
4                       .line_end = (.current_pc - 1) +
4                                       .dpc_entry [next_uns_long];
4                       RETURN TRUE;
3                       END;
3
3               [dst$k_incr_linum]:
3                       dpc_entry = dpc_entry [add_two_bytes];
3
3               [dst$k_incr_linum_w]:
3                       dpc_entry = dpc_entry [add_three_bytes];
3
3               [dst$k_incr_linum_l]:
3                       dpc_entry = dpc_entry [add_five_bytes];
3
3               [dst$k_set_linum_incr]:
3                       dpc_entry = dpc_entry [add_two_bytes];
3
3               [dst$k_set_linum_incr_w]:
3                       dpc_entry = dpc_entry [add_three_bytes];
3
3               [dst$k_reset_linum_incr]:
3                       dpc_entry = dpc_entry [add_one_byte];
3
3               [dst$k_beg_stmt_mode]:
3                       dpc_entry = dpc_entry[add_one_byte];
3
3               [dst$k_end_stmt_mode]:
3                       dpc_entry = dpc_entry[add_one_byte];
3
3               [dst$k_set_linum_b]:
3                       dpc_entry = dpc_entry[add_two_bytes];
3
3               [dst$k_set_linum]:
3                       dpc_entry = dpc_entry[add_three_bytes];
3
3               [dst$k_set_linum_l]:
3                       dpc_entry = dpc_entry[add_five_bytes];
3
3               [dst$k_set_stmtnum]:
3                       dpc_entry = dpc_entry[add_three_bytes];
3
3               [dst$k_set_pc]:
4                       BEGIN
4                       .line_end = (.start_pc - 1) +
4                                       .dpc_entry[next_uns_byte];
4                       RETURN TRUE;
3                       END;
3
3               [dst$k_set_pc_w]:
4                       BEGIN
4                       .line_end = (.start_pc - 1) +
4                                       .dpc_entry[next_uns_word];
4                       RETURN TRUE;
```

```
3                                       END;
3
3                               [dst$k_set_pc_l]:
4                                       BEGIN
4                                       .line_end = (.start_pc - 1) +
4                                                       .dpc_entry[next_uns_long];
4                                       RETURN TRUE;
3                                       END;
3
3                               [DST$K_SET_ABS_PC]:
4                                       BEGIN
4                                       .LINE_END = .DPC_ENTRY[NEXT_UNS_LONG] - 1;
4                                       RETURN TRUE;
3                                       END;
3
3                               [dst$k_term]:
4                                       BEGIN
4                                       .line_end = (.current_pc - 1) +
4                                                       .dpc_entry[next_uns_byte];
4                                       RETURN TRUE;
3                                       END;
3
3                               [dst$k_term_w]:
4                                       BEGIN
4                                       .line_end = (.current_pc - 1) +
4                                                       .dpc_entry[next_uns_word];
4                                       RETURN TRUE;
3                                       END;
3
3                               [dst$k_term_l]:
4                                       BEGIN
4                                       .line_end = (.current_pc - 1) +
4                                                       .dpc_entry[next_uns_long];
4                                       RETURN TRUE;
3                                       END;
3
3                               [OUTRANGE]:
4                                       BEGIN
4                                       IF .dpc_entry [current_byte] LSS
4                                                               dst$k_delta_pc_low
4                                       OR .dpc_entry[current_byte] GTR
4                                                               dst$k_delta_pc_high
4                                       THEN
4                                               SIGNAL (dbg$_invdstrec);
4
4                                       .line_end = (.current_pc - 1) -
4                                                       .dpc_entry [current_byte];
4                                       RETURN TRUE;
3                                       END;
3
3                               TES;
3
2                       END;
2
2               RETURN 0;
1               END;
```

```
                                         000C 00000 FIND_EOL:
                                                              .WORD   Save R2,R3                          ; 1126
                           53 00000000'    EF   9E 00002       MOVAB   DPC_ENTRY, R3
                           50         FC   B3   9A 00009 1$:   MOVZBL  @DST_ENTRY, R0                      ; 1162
                           50         FC   A3   C0 0000D       ADDL2   DST_ENTRY, R0
                           50              63   D1 00011       CMPL    DPC_ENTRY, R0                       ; 1161
                                           1B   15 00014       BLEQ    3$
                                      50   A3   D6 00016       INCL    PCTBL_COUNT                         ; 1165
                      38   A3         50   A3   D1 00019       CMPL    PCTBL_COUNT, NUM_PC_TBLS            ; 1166
                                           03   15 0001E       BLEQ    2$
                                         00AE   31 00020       BRW     22$
                      3C   A3              04   C0 00023 2$:   ADDL2   #4, CURRENT_TABLE                   ; 1167
                      FC   A3         3C   B3   D0 00027       MOVL    @CURRENT_TABLE, DST_ENTRY           ; 1168
                 63   FC   A3              02   C1 0002C       ADDL3   #2, DST_ENTRY, DPC_ENTRY            ; 1169
                           52              63   D0 00031 3$:   MOVL    DPC_ENTRY, R2                       ; 1176
                 14        01              62   8F 00034       CASEB   (R2), #1, #20
    0054        005E        0054         0046      00038 4$:   .WORD   6$-4$,-
    0050        0050        0050         005E      00040               9$-4$,-
    0074        006A        0064         005E      00048               11$-4$,-
    007C        0046        0084         005E      00050               9$-4$,-
    0059        0054        0059         008A      00058               11$-4$,-
                                         008A      00060               8$-4$,-
                                                                       8$-4$,-
                                                                       8$-4$,-
                                                                       11$-4$,-
                                                                       13$-4$,-
                                                                       14$-4$,-
                                                                       16$-4$,-
                                                                       11$-4$,-
                                                                       18$-4$,-
                                                                       6$-4$,-
                                                                       17$-4$,-
                                                                       19$-4$,-
                                                                       10$-4$,-
                                                                       9$-4$,-
                                                                       10$-4$,-
                                                                       19$-4$
                                           62   95 00062       TSTB    (R2)                                ; 1280
                                           0D   15 00064       BLEQ    5$
                              0002832A     8F   DD 00066       PUSHL   #164650                             ; 1283
               00000000G   00              01   FB 0006C       CALLS   #1, LIB$SIGNAL
                           50         00   B3   98 00073 5$:   CVTBL   @DPC_ENTRY, R0                      ; 1286
                 50   14   A3              50   C3 00077       SUBL3   R0, CURRENT_PC, R0
                                           4A   11 0007C       BRB     20$                                 ; 1285
                           50         01   A2   3C 0007E 6$:   MOVZWL  1(R2), R0                           ; 1181
                           50         14   A3   C0 00082 7$:   ADDL2   CURRENT_PC, R0
                                           40   11 00086       BRB     20$                                 ; 1180
                                           63   D6 00088 8$:   INCL    DPC_ENTRY                           ; 1214
                                           0D   11 0008A       BRB     12$
                           63              02   C0 0008C 9$:   ADDL2   #2, DPC_ENTRY                       ; 1217
                                           08   11 0008F       BRB     12$
                           63              05   C0 00091 10$:  ADDL2   #5, DPC_ENTRY                       ; 1223
                                           03   11 00094       BRB     12$                                 ;
```

```
                          63          03  C0 00096 11$:    ADDL2   #3, DPC_ENTRY                              ; 1226
                                    FF6D  31 00099 12$:    BRW     1$                                         ;
                          50     01   A2  9A 0009C 13$:    MOVZBL  1(R2), R0                                  ; 1231
                                      04  11 000A0         BRB     15$                                        ;
                          50     01   A2  3C 000A2 14$:    MOVZWL  1(R2), R0                                  ; 1238
                          50     04   A3  C0 000A6 15$:    ADDL2   START_PC, R0                               ;
                                      1C  11 000AA         BRB     20$                                        ; 1237
            50       04   A3     01   A2  C1 000AC 16$:    ADDL3   1(R2), START_PC, R0                        ; 1245
                                      14  11 000B2         BRB     20$                                        ; 1244
       04   BC       01   A2          01  C3 000B4 17$:    SUBL3   #1, 1(R2), @LINE_END                       ; 1251
                                      11  11 000BA         BRB     21$                                        ; 1252
                          50     01   A2  9A 000BC 18$:    MOVZBL  1(R2), R0                                  ; 1258
                                      C0  11 000C0         BRB     7$                                         ; 1265
            50       14   A3     01   A2  C1 000C2 19$:    ADDL3   1(R2), CURRENT_PC, R0                      ; 1272
                     04   BC     FF   A0  9E 000C8 20$:    MOVAB   -1(R0), @LINE_END                          ; 1271
                          50          01  D0 000CD 21$:    MOVL    #1, R0                                     ; 1273
                                          04 000D0         RET                                                ;
                                      50  D4 000D1 22$:    CLRL    R0                                         ; 1295
                                          04 000D3         RET                                                ;

; Routine Size:  212 bytes,    Routine Base:  DBG$CODE + 04C3
```

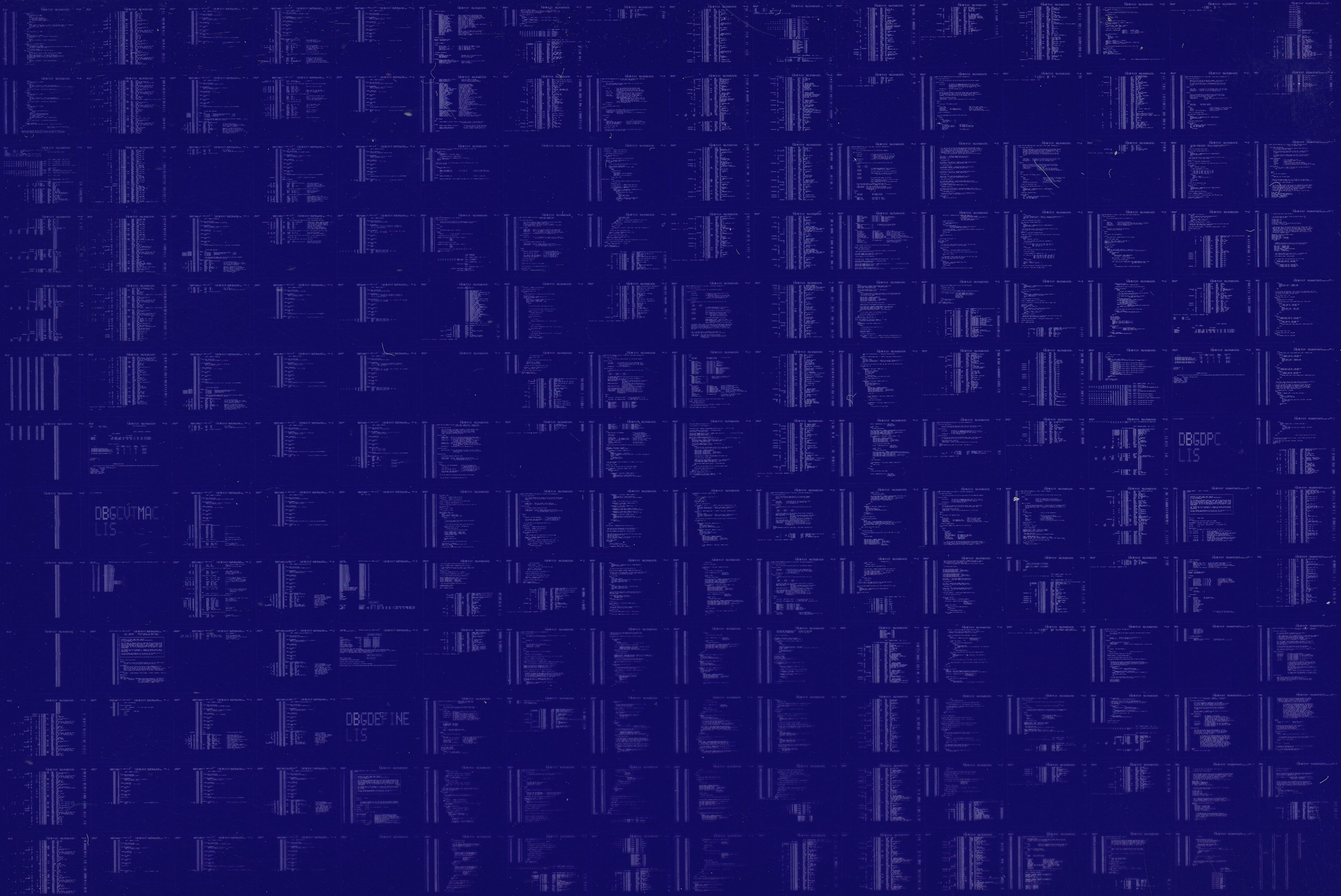
```
ROUTINE GIVE_LINE_INFO(LINE_NUM, STMT_NUM): NOVALUE =
!
! FUNCTION
!       This routine gives prev., current, next line information to the user
!       when the desired line is not found.
!
! INPUTS
!       REPORT_PREV_LINE - Previous line
!       REPORT_PREV_STMT - Previous statement
!       LINE_NUM         - Current line
!       STMT_NUM         - Current statement
!       REPORT_NEXT_LINE - Next line
!       REPORT_NEXT_STMT - Next statement
!
! OUTPUTS
!       Informational message is displayed.  No return value.
!

    BEGIN

    LOCAL
        BUFFER: VECTOR[80, BYTE],          ! Output buffer
        BUF_DESC: VECTOR[2, LONG];         ! Output buffer string descriptor


    IF .STMT_NUM EQL 0 THEN STMT_NUM = 1;
    IF .REPORT_PREV_STMT EQL 0 THEN REPORT_PREV_STMT = 1;
    IF .REPORT_NEXT_STMT EQL 0 THEN REPORT_NEXT_STMT = 1;

    BUF_DESC[0] = 79;
    BUF_DESC[1] = BUFFER[1];

    IF (.REPORT_PREV_LINE EQL 0) AND
       (.LINE_NUM EQL .REPORT_NEXT_LINE) AND
       (.REPORT_PREV_STMT EQL 1) AND
       (.STMT_NUM EQL .REPORT_NEXT_STMT)
    THEN
        BEGIN
        DBG$FORMAT_FAO_OUT(BUF_DESC, UPLIT BYTE
            (%ASCIC 'no line information available'));
        BUFFER[0] = 79 - .BUF_DESC[0];
        SIGNAL(DBG$_LINEINFO, 1, BUFFER);
        RETURN 0;
        END;

    DBG$FORMAT_FAO_OUT(BUF_DESC, UPLIT BYTE(%ASCIC 'no line !UL'), .LINE_NUM);
    IF .STMT_NUM GTR 1
    THEN
        DBG$FORMAT_FAO_OUT(BUF_DESC, UPLIT BYTE(%ASCIC '.!UL'), .STMT_NUM);

    IF NOT (.REPORT_PREV_LINE EQL 0 AND .REPORT_PREV_STMT EQL 1)
    THEN
        BEGIN
        DBG$FORMAT_FAO_OUT(BUF_DESC, UPLIT BYTE
                (%ASCIC ', previous line is !UL'), .REPORT_PREV_LINE);

        IF .REPORT_PREV_STMT GTR 1
```

```
; 1227          1353 3          THEN
; 1228          1354 3              DBG$FORMAT_FAO_OUT(BUF_DESC, UPLIT BYTE(%ASCIC '.!UL'), .REPORT_PREV_STMT);
; 1229          1355 2          END;
; 1230          1356 2
; 1231          1357 3      IF NOT (.REPORT_NEXT_LINE EQL .LINE_NUM  AND
; 1232          1358 3              .REPORT_NEXT_STMT EQL .STMT_NUM)
; 1233          1359 2      THEN
; 1234          1360 3          BEGIN
; 1235          1361 3          DBG$FORMAT_FAO_OUT(BUF_DESC, UPLIT BYTE
; 1236          1362 3                  (%ASCIC ', next line is !UL'), .REPORT_NEXT_LINE);
; 1237          1363 3
; 1238          1364 3          IF .REPORT_NEXT_STMT GTR 1
; 1239          1365 3          THEN
; 1240          1366 3              DBG$FORMAT_FAO_OUT(BUF_DESC, UPLIT BYTE(%ASCIC '.!UL'), .REPORT_NEXT_STMT);
; 1241          1367 2          END;
; 1242          1368 2
; 1243          1369 2      BUFFER[0] = 79 - .BUF_DESC[0];
; 1244          1370 2      SIGNAL(DBG$_LINEINFO, 1, BUFFER);
; 1245          1371 2      RETURN 0;
; 1246          1372 1      END;



                                                                                .PSECT  DBG$PLIT,NOWRT,  SHR,  PIC,0

6D  72  6F  66  6E  69  20  65  6E  69  6C  20  6F  6E  1D  00000 P.AAA:   .ASCII  <29>\no line information available\          :
65  6C  62  61  6C  69  61  76  61  20  6E  6F  69  74  61  0000F                                                                  :
            4C  55  21  20  65  6E  69  6C  20  6F  6E  0B  0001E P.AAB:   .ASCII  <11>\no line !UL\                            :
                                        4C  55  21  2E  04  0002A P.AAC:   .ASCII  <4>\.!UL\                                    :
6E  69  6C  20  73  75  6F  69  76  65  72  70  20  2C  16  0002F P.AAD:   .ASCII  <22>\, previous line is !UL\                 :
                                4C  55  21  20  73  69  20  65  0003E                                                          :
                                        4C  55  21  2E  04  00046 P.AAE:   .ASCII  <4>\.!UL\                                    :
73  69  20  65  6E  69  6C  20  74  78  65  6E  20  2C  12  0004B P.AAF:   .ASCII  <18>\, next line is !UL\                     :
                                        4C  55  21  20  0005A                                                                  :
                                        4C  55  21  2E  04  0005E P.AAG:   .ASCII  <4>\.!UL\                                    :



                                                                                .PSECT  DBG$CODE,NOWRT,  SHR,  PIC,0

                                               001C 00000 GIVE_LINE_INFO:
                                                                                .WORD   Save R2,R3,R4                          : 1296
                                 54 0000000G   00  9E 00002                     MOVAB   DBG$FORMAT_FAO_OUT, R4                 :
                                 53 00000000'  EF  9E 00009                     MOVAB   P.AAA, R3                              :
                                 52 00000000'  EF  9E 00010                     MOVAB   REPORT_PREV_STMT, R2                   :
                                 5E         A8  AE  9E 00017                     MOVAB   -88(SP), SP                            :
                                            08  AC  D5 0001B                     TSTL    STMT_NUM                               : 1321
                                                04  12 0001E                     BNEQ    1$                                     :
                             08  AC             01  D0 00020                     MOVL    #1, STMT_NUM                           :
                                                62  D5 00024 1$:                 TSTL    REPORT_PREV_STMT                       : 1322
                                                03  12 00026                     BNEQ    2$                                     :
                                 62             01  D0 00028                     MOVL    #1, REPORT_PREV_STMT                   :
                                            F8  A2  D5 0002B 2$:                 TSTL    REPORT_NEXT_STMT                       : 1323
                                                04  12 0002E                     BNEQ    3$                                     :
                             F8  A2             01  D0 00030                     MOVL    #1, REPORT_NEXT_STMT                   :
                                 6E         4F  8F  9A 00034 3$:                 MOVZBL  #79, BUF_DESC                          : 1325
                             04  AE         09  AE  9E 00038                     MOVAB   BUFFER+1, BUF_DESC+4                   : 1326
                             FC   A2 D5 0003D         TSTL    REPORT_PREV_LINE                      ; 1328
                                  1D 12 00040         BNEQ    4$                                    ;
                 F4   A2     04   AC D1 00042         CMPL    LINE_NUM, REPORT_NEXT_LINE            ; 1329
                                  16 12 00047         BNEQ    4$                                    ;
                      01          62 D1 00049         CMPL    REPORT_PREV_STMT, #1                  ; 1330
                                  11 12 0004C         BNEQ    4$                                    ;
                 F8   A2     08   AC D1 0004E         CMPL    STMT_NUM, REPORT_NEXT_STMT            ; 1331
                                  0A 12 00053         BNEQ    4$                                    ;
                                  53 DD 00055         PUSHL   R3                                    ; 1334
                             04   AE 9F 00057         PUSHAB  BUF_DESC                              ;
                      64          02 FB 0005A         CALLS   #2, DBG$FORMAT_FAO_OUT                ;
                                  70 11 0005D         BRB     9$                                    ; 1336
                             04   AC DD 0005F 4$:     PUSHL   LINE_NUM                              ; 1341
                             1E   A3 9F 00062         PUSHAB  P.AAB                                 ;
                             08   AE 9F 00065         PUSHAB  BUF_DESC                              ;
                      64          03 FB 00068         CALLS   #3, DBG$FORMAT_FAO_OUT                ;
                      01     08   AC D1 0006B         CMPL    STMT_NUM, #1                          ; 1342
                                  0C 15 0006F         BLEQ    5$                                    ;
                             08   AC DD 00071         PUSHL   STMT_NUM                              ; 1344
                             2A   A3 9F 00074         PUSHAB  P.AAC                                 ;
                             08   AE 9F 00077         PUSHAB  BUF_DESC                              ;
                      64          03 FB 0007A         CALLS   #3, DBG$FORMAT_FAO_OUT                ;
                      50     FC   A2 D0 0007D 5$:     MOVL    REPORT_PREV_LINE, R0                  ; 1346
                                  05 12 00081         BNEQ    6$                                    ;
                      01          62 D1 00083         CMPL    REPORT_PREV_STMT, #1                  ;
                                  1B 13 00086         BEQL    7$                                    ;
                                  50 DD 00088 6$:     PUSHL   R0                                    ; 1350
                             2F   A3 9F 0008A         PUSHAB  P.AAD                                 ; 1349
                             08   AE 9F 0008D         PUSHAB  BUF_DESC                              ;
                      64          03 FB 00090         CALLS   #3, DBG$FORMAT_FAO_OUT                ;
                      01          62 D1 00093         CMPL    REPORT_PREV_STMT, #1                  ; 1352
                                  0B 15 00096         BLEQ    7$                                    ;
                                  62 DD 00098         PUSHL   REPORT_PREV_STMT                      ; 1354
                      46          A3 9F 0009A         PUSHAB  P.AAE                                 ;
                             08   AE 9F 0009D         PUSHAB  BUF_DESC                              ;
                      64          03 FB 000A0         CALLS   #3, DBG$FORMAT_FAO_OUT                ;
                 04   AC     F4   A2 D1 000A3 7$:     CMPL    REPORT_NEXT_LINE, LINE_NUM            ; 1357
                                  07 12 000A8         BNEQ    8$                                    ;
                 08   AC     F8   A2 D1 000AA         CMPL    REPORT_NEXT_STMT, STMT_NUM            ; 1358
                                  1E 13 000AF         BEQL    9$                                    ;
                             F4   A2 DD 000B1 8$:     PUSHL   REPORT_NEXT_LINE                      ; 1362
                             4B   A3 9F 000B4         PUSHAB  P.AAF                                 ; 1361
                             08   AE 9F 000B7         PUSHAB  BUF_DESC                              ;
                      64          03 FB 000BA         CALLS   #3, DBG$FORMAT_FAO_OUT                ;
                      01     F8   A2 D1 000BD         CMPL    REPORT_NEXT_STMT, #1                  ; 1364
                                  0C 15 000C1         BLEQ    9$                                    ;
                             F8   A2 DD 000C3         PUSHL   REPORT_NEXT_STMT                      ; 1366
                             5E   A3 9F 000C6         PUSHAB  P.AAG                                 ;
                             08   AE 9F 000C9         PUSHAB  BUF_DESC                              ;
                      64          03 FB 000CC         CALLS   #3, DBG$FORMAT_FAO_OUT                ;
08    AE         4F   8F          6E 83 000CF 9$:     SUBB3   BUF_DESC, #79, BUFFER                 ; 1369
                             08   AE 9F 000D5         PUSHAB  BUFFER                                ; 1370
                                  01 DD 000D8         PUSHL   #1                                    ;
                       00028703   8F DD 000DA         PUSHL   #165635                               ;
          00000000G   00          03 FB 000E0         CALLS   #3, LIB$SIGNAL                        ;
                                     04 000E7         RET                                           ; 1372
```

```
; Routine Size:  232 bytes,    Routine Base:  DBG$CODE + 0597

; 1247          1373  1
; 1248          1374  1 END
; 1249          1375  0 ELUDOM



                                                        .EXTRN  LIB$SIGNAL

;                               PSECT SUMMARY
;
;       Name                    Bytes                   Attributes
;
;    DBG$OWN                       88  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;    DBG$CODE                    1663  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(0)
;    DBG$PLIT                      99  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(0)


;                       Library Statistics
;
;                                               -------- Symbols --------       Pages       Processing
;       File                                    Total    Loaded   Percent       Mapped      Time
;
;    _$255$DUA28:[SYSLIB]LIB.L32;1              18619         0         0         1000       00:01.8
;    _$255$DUA28:[DEBUG.OBJ]STRUCDEF.L32;1         32         0         0            7       00:00.1
;    _$255$DUA28:[DEBUG.OBJ]DBGLIB.L32;1         1545        56         3           97       00:01.8
;    _$255$DUA28:[DEBUG.OBJ]DSTRECRDS.L32;1
;                                                 418       127        30           31       00:00.3
;    _$255$DUA28:[DEBUG.OBJ]DBGMSG.L32;1          386         2         0           22       00:00.3
;    _$255$DUA28:[DEBUG.OBJ]DBGGEN.L32;1          150         0         0           12       00:00.3



;                               COMMAND QUALIFIERS

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

; Size:           1663 code + 187 data bytes
; Run Time:          00:35.2
; Elapsed Time:      02:05.1
; Lines/CPU Min:       2343
; Lexemes/CPU-Min:  12071
; Memory Used:   221 pages
; Compilation Complete
```

0079 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



0080 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

DBGEVALOP
LIS

DBGENCDEC
LIS