```
TTTTTTTTTTTTTTT  RRRRRRRRRRRR        AAAAAAAAA        CCCCCCCCCCCC  EEEEEEEEEEEEEEE
TTTTTTTTTTTTTTT  RRRRRRRRRRRR        AAAAAAAAA        CCCCCCCCCCCC  EEEEEEEEEEEEEEE
TTTTTTTTTTTTTTT  RRRRRRRRRRRR        AAAAAAAAA        CCCCCCCCCCCC  EEEEEEEEEEEEEEE
      TTT        RRR         RRR  AAA         AAA  CCC              EEE
      TTT        RRR         RRR  AAA         AAA  CCC              EEE
      TTT        RRR         RRR  AAA         AAA  CCC              EEE
      TTT        RRR         RRR  AAA         AAA  CCC              EEE
      TTT        RRR         RRR  AAA         AAA  CCC              EEE
      TTT        RRR         RRR  AAA         AAA  CCC              EEE
      TTT        RRRRRRRRRRRR     AAA         AAA  CCC              EEEEEEEEEEEE
      TTT        RRRRRRRRRRRR     AAA         AAA  CCC              EEEEEEEEEEEE
      TTT        RRRRRRRRRRRR     AAA         AAA  CCC              EEEEEEEEEEEE
      TTT        RRR   RRR        AAAAAAAAAAAAAAA  CCC              EEE
      TTT        RRR   RRR        AAAAAAAAAAAAAAA  CCC              EEE
      TTT        RRR   RRR        AAAAAAAAAAAAAAA  CCC              EEE
      TTT        RRR      RRR     AAA         AAA  CCC              EEE
      TTT        RRR      RRR     AAA         AAA  CCC              EEE
      TTT        RRR      RRR     AAA         AAA  CCC              EEE
      TTT        RRR         RRR  AAA         AAA     CCCCCCCCCCCC  EEEEEEEEEEEEEEE
      TTT        RRR         RRR  AAA         AAA     CCCCCCCCCCCC  EEEEEEEEEEEEEEE
      TTT        RRR         RRR  AAA         AAA     CCCCCCCCCCCC  EEEEEEEEEEEEEEE
```

```
**FILE**ID**TBKDPC
```

```
TTTTTTTTTT  BBBBBBBB    KK      KK  DDDDDDDD    PPPPPPPP      CCCCCCCC
TTTTTTTTTT  BBBBBBBB    KK      KK  DDDDDDDD    PPPPPPPP      CCCCCCCC
    TT      BB      BB  KK      KK  DD      DD  PP      PP  CC
    TT      BB      BB  KK      KK  DD      DD  PP      PP  CC
    TT      BB      BB  KK    KK    DD      DD  PP      PP  CC
    TT      BB      BB  KK    KK    DD      DD  PP      PP  CC
    TT      BBBBBBBB    KKKKKK      DD      DD  PPPPPPPP    CC
    TT      BBBBBBBB    KKKKKK      DD      DD  PPPPPPPP    CC
    TT      BB      BB  KK    KK    DD      DD  PP          CC
    TT      BB      BB  KK    KK    DD      DD  PP          CC
    TT      BB      BB  KK      KK  DD      DD  PP          CC              ....
    TT      BB      BB  KK      KK  DD      DD  PP          CC              ....
    TT      BBBBBBBB    KK      KK  DDDDDDDD    PP            CCCCCCCC      ....
    TT      BBBBBBBB    KK      KK  DDDDDDDD    PP            CCCCCCCC      ....


LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
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
```

TBK
V04

```
0 MODULE TBKDPC ( IDENT = 'V04-000' ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY:
1 !       TRACEBACK
1 !
1 ! ABSTRACT:
1 !       analyzes PC correlation tables for DEBUG and for symbolic
1 !       traceback.
1 !
1 ! ENVIRONMENT:  VAX/VMS, user mode, interrupts disabled.
1 !
1 ! AUTHOR:       Carol Peters, CREATION DATE:    16 September 1977
1 !
1 ! Version       13
1 !
1 ! MODIFIED BY:
1 !               Dale Roedger, 15 June 1978: Version 13
1 !               Sid Maxwell     09-Dec-81
1 !
1 !       15-Aug-83       PS      Did general clean up to use updated files
1 !                               from DEBUG.
1 !          Jan-84       RT      Changed TBK$PC_TO_LINE so that it only
1 !                               reports a match if the pc/line tables
1 !                               indicate that the line is "open" (i.e.,
1 !                               "TERM" records now close the line and
1 !                               prevent a match.) This fixes a problem
1 !                               we were seeing with RPG programs (They
1 !                               have code not associated with lines).
1 !--
```

```
! TABLE OF CONTENTS:
!

FORWARD ROUTINE
        TBK$PC_TO_LINE,                 ! matches a PC to a line number
        PROC_PC_CMD,                    ! processes a string of PC correlation commands
        GET_NEXT_DPC;                   ! gets the next PC correlation record


! REQUIRE FILES:
!
REQUIRE 'SRC$:TBKPROLOG.REQ';


! MACROS:
!
MACRO
        first_dpc_datum = 2, 0, 32, 0%,         ! passes count and type
        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:
!

! The body of a PC/LINE Table Record is interpreted as a sequence of commands
! each of which supplies some information about line/statement numbers in the
! context of the preceding commands.  The value is taken from DSTRECRDS.REQ.


LITERAL
        line_open       = 1,
        line_closed     = 2;

!
! OWN STORAGE:
!
OWN
        dst_entry       : REF dst$record,
        dpc_entry       : REF BLOCK [, BYTE],
        start_pc,
        current_line,
        current_stmt,
        current_incr,
        current_pc,
        current_stmt_mode,
        prev_line,
        prev_stmt,
        prev_incr,
        prev_pc,
        prev_stmt_mode,
1          current_mark,
1          prev_mark;
1
1 !
1 ! EXTERNAL REFERENCES:
1 !
1 EXTERNAL
1          tbk$module_dst : REF dst$record;
1
1 EXTERNAL ROUTINE
1          TBK$fake_MSG,
1          TBK$FAO_OUT : NOVALUE,
1          tbk$get_dst_rec,                        ! gets a DST record from a DST pointer.
1          tbk$get_nxt_dst,                        ! gets next DST record in sequence
1          tbk$POSITON_DST;                        ! Set up the DST 'next' sequence.
```

```
1 GLOBAL ROUTINE tbk$pc_to_line (match_pc_ptr, routine_address, excep_type,
1                               line_no_ptr  stmt_no_ptr) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !       This routine matches an absolute PC address to a line number
1 !       in a FORTRAN routine.  MATCH_PC is the given PC,
1 !       and the location pointed to by LINE_NO_PTR
1 !       is written as a result of delta-PC table analysis.
1 !
1 !       Each PC correlation record that exists for a single routine
1 !       is sequentially analyzed until the desired PC is seen.
1 !
1 !       If a match cannot be made because and end of routine record or
1 !       an invalid record is recognized, then this routine returns
1 !       FALSE.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       match_pc_ptr    - a pointer to the PC to be matched.
1 !       routine_address - DST of record for enclosing routine.
1 !       excep_type      - the type of exception, where
1 !                               zero, means irrelevant;
1 !                               one, means trap type exception,
1 !                               two, means fault or abort type exception.
1 !       line_no_ptr     - a copy-back pointer for the line number.
1 !       stmt_no_ptr     - a copy-back pointer for the statement number.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       The DST is already positioned to the record AFTER
1 !       the ROUTINE record we want to look at line numbers for.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       the routine get_nxt_dst is set up to next return the record after
1 !       the end of routine record or the record after the PC correlation
1 !       record that matched the given parameters.
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       true, if success; false, if any error or if match cannot
1 !       be made.
1 !
1 ! SIDE EFFECTS:
1 !
1 !       The DST is positioned for a GET_NXT_DST sequence.
1 !
1 !--
1
2       BEGIN
2
2       LOCAL   match_pc,
2               low_routine,
2               real_value;
2
```

```
! treat traps as faults by debumping PC
!
IF      .excep_type EQL trap_exc
THEN    match_pc = .match_pc_ptr - 1
ELSE    match_pc = .match_pc_ptr;

IF tbk$positon_dst(.tbk$module_dst) EQL 0
THEN
        RETURN FALSE;
dst_entry = .tbk$module_dst;
low_routine = -1;
REPEAT
        BEGIN
        dst_entry = tbk$get_nxt_dst(dst_entry);
        IF .dst_entry EQL 0
        THEN
                RETURN FALSE;
        IF .dst_entry[dst$b_type] EQL dst$k_modend
        THEN
                EXITLOOP;
        IF .dst_entry[dst$b_type] EQL dst$k_rtnbeg
        THEN
                BEGIN
                IF .dst_entry[dst$l_value] LSSA .low_routine
                THEN
                        low_routine = .dst_entry[dst$l_value];
                END;
END;


IF tbk$positon_dst(.tbk$module_dst) EQL 0
THEN
        RETURN FALSE;
IF get_next_dpc(dst_entry) EQL 0
THEN
        RETURN FALSE;
dpc_entry = dst_entry[dst$b_vflags];


!++
! Initialize state variables.
!--
current_line = 0;
current_stmt = 1;
current_incr = 1;
current_stmt_mode = FALSE;
current_pc = start_pc = .low_routine;
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 true, otherwise return false.
!--
```

```
;   245          0514  2          REPEAT
;   246          0515  3                  BEGIN
;   247          0516  3
;   248          0517  3                  prev_line = .current_line;
;   249          0518  3                  prev_stmt = .current_stmt;
;   250          0519  3                  prev_incr = .current_incr;
;   251          0520  3                  prev_stmt_mode = .current_stmt_mode;
;   252          0521  3                  prev_pc = .current_pc;
;   253          0522  3                  prev_mark = .current_mark;
;   254          0523  3
;   255          0524  3
;   256          0525  3                  IF NOT proc_pc_cmd ( )
;   257          0526  3                  THEN
;   258          0527  3                      RETURN FALSE;
;   259          0528  3
;   260          0529  3
;   261          0530  3                  ! Report a match to a line if:
;   262          0531  3                  ! - The PC is within the range given by the previous
;   263          0532  3                  !   PC and the current PC, and
;   264          0533  3                  ! - The line is marked as being OPEN.
;   265          0534  3                  !
;   266          0535  4                  IF      ((.prev_pc LEQA .match_pc) AND
;   267          0536  4                           (.match_pc LSSA .current_pc) AND
;   268          0537  4                           (.prev_mark EQL line_open))
;   269          0538  5                  THEN    BEGIN   .stmt_no_ptr =  (IF     .prev_stmt EQL 1
;   270          0539  5                                                  THEN   0
;   271          0540  4                                                  ELSE   .prev_stmt);
;   272          0541  4                                  .line_no_ptr = .prev_line;
;   273          0542  4                                  RETURN  TRUE
;   274          0543  3                          END;
;   275          0544  3
;   276          0545  3                  !++
;   277          0546  3                  ! Found nothing this round; continue trying.
;   278          0547  3                  !--
;   279          0548  3
;   280          0549  3                  END
;   281          0550  1          END;


                                                                  .TITLE  TBKDPC
                                                                  .IDENT  \V04-000\

                                                                  .PSECT  TBK$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 PREV_LINE:
                                                                       .BLKB   4
                                                00024 PREV_STMT:
                                                                       .BLKB   4
                                                00028 PREV_INCR:
                                                                       .BLKB   4
                                                0002C PREV_PC:.BLKB    4
                                                00030 PREV_STMT_MODE:
                                                                       .BLKB   4
                                                00034 CURRENT_MARK:
                                                                       .BLKB   4
                                                00038 PREV_MARK:
                                                                       .BLKB   4

                                                                       .EXTRN  TBK$MODULE_DST, TBK$FAKE_MSG
                                                                       .EXTRN  TBK$FAO_OUT, TBK$GET_DST_REC
                                                                       .EXTRN  TBK$GET_NXT_DST
                                                                       .EXTRN  TBK$POSITON_DST

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

                                            007C 00000                 .ENTRY  TBK$PC_TO_LINE, Save R2,R3,R4,R5,R6              ; 0400
                      56 00000000G    00    9E 00002                   MOVAB   TBK$POSITON_DST, R6                              ;
                      55 00000000G    00    9E 00009                   MOVAB   TBK$MODULE_DST, R5                               ;
                      54     0000'    CF    9E 00010                   MOVAB   DST_ENTRY, R4                                    ;
                      01        0C    AC    D1 00015                   CMPL    EXCEP_TYPE, #1                                   ; 0459
                                      07    12 00019                   BNEQ    1$                                               ;
    53          04    AC              01    C3 0001B                   SUBL3   #1, MATCH_PC_PTR, MATCH_PC                       ; 0460
                                      04    11 00020                   BRB     2$                                               ;
                      53          04  AC    D0 00022 1$:               MOVL    MATCH_PC_PTR, MATCH_PC                           ; 0461
                                      65    DD 00026 2$:               PUSHL   TBK$MODULE_DST                                   ; 0463
                      66              01    FB 00028                   CALLS   #1, TBK$POSITON_DST                              ;
                                      50    D5 0002B                   TSTL    R0                                               ;
                                      7D    13 0002D                   BEQL    6$                                               ;
                      64              65    D0 0002F                   MOVL    TBK$MODULE_DST, DST_ENTRY                        ; 0466
                      52              01    CE 00032                   MNEGL   #1, LOW_ROUTINE                                  ; 0467
                                      54    DD 00035 3$:               PUSHL   R4                                               ; 0470
          00000000G   00              01    FB 00037                   CALLS   #1, TBK$GET_NXT_DST                              ;
                      64              50    D0 0003E                   MOVL    R0, DST_ENTRY                                    ;
                                      69    13 00041                   BEQL    6$                                               ; 0471
                BD    8F        01    A0    91 00043                   CMPB    1(R0), #189                                      ; 0474
                                      13    13 00048                   BEQL    4$                                               ;
                BE    8F        01    A0    91 0004A                   CMPB    1(R0), #190                                      ; 0477
                                      E4    12 0004F                   BNEQ    3$                                               ;
                      52        03    A0    D1 00051                   CMPL    3(R0), LOW_ROUTINE                               ; 0480
                                      DE    1E 00055                   BGEQU   3$                                               ;
                      52        03    A0    D0 00057                   MOVL    3(R0), LOW_ROUTINE                               ; 0482
                                      D8    11 0005B                   BRB     3$                                               ; 0467
                                      65    DD 0005D 4$:               PUSHL   TBK$MODULE_DST                                   ; 0487
                      66              01    FB 0005F                   CALLS   #1, TBK$POSITON_DST                              ;
                                      50    D5 00062                   TSTL    R0                                               ;
                                      46    13 00064                   BEQL    6$                                               ;
                                      54    DD 00066                   PUSHL   R4                                               ; 0490
               0000V  CF              01    FB 00068                   CALLS   #1, GET_NEXT_DPC                                 ;
```

```
                                          50  D5 0006D           TSTL    R0                                          ;
                                          3B  13 0006F           BEQL    6$                                          ;
               04   A4           64       02  C1 00071           ADDL3   #2, DST_ENTRY, DPC_ENTRY                    ; 0493
                                     0C   A4  D4 00076           CLRL    CURRENT_LINE                                ; 0499
                          10   A4         01  D0 00079           MOVL    #1, CURRENT_STMT                            ; 0500
                          14   A4         01  D0 0007D           MOVL    #1, CURRENT_INCR                            ; 0501
                                     1C   A4  D4 00081           CLRL    CURRENT_STMT_MODE                           ; 0502
                          08   A4         52  D0 00084           MOVL    LOW_ROUTINE, START_PC                       ; 0503
                          18   A4         52  D0 00088           MOVL    LOW_ROUTINE, CURRENT_PC                     ;
                          34   A4         02  D0 0008C           MOVL    #2, CURRENT_MARK                            ; 0504
                          20   A4    0C   A4  7D 00090 5$:       MOVQ    CURRENT_LINE, PREV_LINE                     ; 0517
                          30   A4    1C   A4  D0 00095           MOVL    CURRENT_STMT_MODE, PREV_STMT_MODE           ; 0520
                          28   A4    14   A4  7D 0009A           MOVQ    CURRENT_INCR, PREV_INCR                     ; 0519
                          38   A4    34   A4  D0 0009F           MOVL    CURRENT_MARK, PREV_MARK                     ; 0522
                       0000V   CF         00  FB 000A4           CALLS   #0, PROC_PC_CMD                             ; 0525
                               03         50  E8 000A9           BLBS    R0, 7$                                      ;
                                          50  D4 000AC 6$:       CLRL    R0                                          ; 0527
                                              04 000AE           RET                                                 ;
                               53    2C   A4  D1 000AF 7$:       CMPL    PREV_PC, MATCH_PC                           ; 0535
                                          DB  1A 000B3           BGTRU   5$                                          ;
                          18   A4         53  D1 000B5           CMPL    MATCH_PC, CURRENT_PC                        ; 0536
                                          D5  1E 000B9           BGEQU   5$                                          ;
                               01    38   A4  D1 000BB           CMPL    PREV_MARK, #1                               ; 0537
                                          CF  12 000BF           BNEQ    5$                                          ;
                               01    24   A4  D1 000C1           CMPL    PREV_STMT, #1                               ; 0538
                                          04  12 000C5           BNEQ    8$                                          ;
                                          50  D4 000C7           CLRL    R0                                          ;
                                          04  11 000C9           BRB     9$                                          ;
                               50    24   A4  D0 000CB 8$:       MOVL    PREV_STMT, R0                               ; 0540
                          14   BC         50  D0 000CF 9$:       MOVL    R0, @STMT_NO_PTR                            ; 0538
                          10   BC    20   A4  D0 000D3           MOVL    PREV_LINE, @LINE_NO_PTR                     ; 0541
                               50         01  D0 000D8           MOVL    #1, R0                                      ; 0542
                                              04 000DB           RET                                                 ; 0550

; Routine Size:  220 bytes,    Routine Base:  TBK$CODE + 0000
```

```
1 ROUTINE PROC_PC_CMD =
1 !++
1 ! Functional description:
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                       IF NOT get_next_dpc(dst_entry)
4                       THEN
4                           RETURN FALSE
4
4                       ELSE
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
```

```
3       ! processed, control returns from this routine to its
3       ! caller.
3       !
3       CASE .dpc_entry [current_byte] FROM 1 TO dst$k_pccor_high OF
3               SET
3
3               ! Read the next two bytes as an unsigned word
3               ! representing a delta-PC value.  Update the next_pc
3               ! and update the dpc_entry address.
3               !
3               [dst$k_delta_pc_w]:
4                       BEGIN
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_mark = line_open;
4                       current_pc = .current_pc +
4                                       .dpc_entry [next_uns_word];
4                       dpc_entry = dpc_entry [add_three_bytes];
4                       RETURN TRUE;
3                       END;
3
3
3               ! Read the next four bytes as an unsigned longword
3               ! representing a delta-PC value.  Update the next_pc
3               ! and update the dpc_entry address.
3               !
3               [dst$k_delta_pc_l]:
4                       BEGIN
4                       IF .current_stmt_mode
4                       THEN
4                               current_stmt = .current_stmt + 1
4                       ELSE
4                               current_line = .current_line +
4                                                       .current_incr;
4
4                       current_mark = line_open;
4                       current_pc = .current_pc +
4                                       .dpc_entry [next_uns_long];
4                       dpc_entry = dpc_entry [add_five_bytes];
4                       RETURN TRUE;
3                       END;
3
3
3               ! Increase the current line number by the value
3               ! contained in the next unsigned byte.
3               !
3               [dst$k_incr_linum]:
4                       BEGIN
4                       current_line = .current_line + .dpc_entry [next_uns_byte];
4                       IF .current_stmt_mode THEN current_stmt = 1;
4                       dpc_entry = dpc_entry [add_two_bytes];
```

```
                END;


        ! Increase the current line number by the value
        ! contained in the next unsigned word.
        !
        [dst$k_incr_linum_w]:
                BEGIN
                IF .current_stmt_mode THEN current_stmt = 1;
                current_line = .current_line + .dpc_entry [next_uns_word];
                dpc_entry = dpc_entry [add_three_bytes];
                END;


        ! Increase the current line number by the value
        ! contained in the next unsigned longword.
        !
        [dst$k_incr_linum_l]:
                BEGIN
                IF .current_stmt_mode THEN current_stmt = 1;
                current_line = .current_line + .dpc_entry [next_uns_long];
                dpc_entry = dpc_entry [add_five_bytes];
                END;


        ! Change the line increment from its present value to
        ! the value contained in the next unsigned byte.
        !
        [dst$k_set_linum_incr]:
                BEGIN
                IF .current_stmt_mode THEN current_stmt = 1;
                current_incr = .dpc_entry [next_uns_byte];
                dpc_entry = dpc_entry [add_two_bytes];
                END;


        ! Change the line increment from its present value to
        ! the value contained in the next word.
        !
        [dst$k_set_linum_incr_w]:
                BEGIN
                IF .current_stmt_mode THEN current_stmt = 1;
                current_incr = .dpc_entry [next_uns_word];
                dpc_entry = dpc_entry [add_three_bytes];
                END;


        ! Revert to a line increment of value 1.
        !
        [dst$k_reset_linum_incr]:
                BEGIN
                IF .current_stmt_mode THEN current_stmt = 1;
                current_incr = 1;
                dpc_entry = dpc_entry [add_one_byte];
                END;

        [dst$k_beg_stmt_mode]:
```

```
;  454          0722  4                               BEGIN
;  455          0723  4                               IF .current_mark NEQ line_open
;  456          0724  4                               THEN
;  457          0725  5                                       BEGIN
;  458          0726  5                                       TBK$FAKE_MSG(TBK$_INVDSTREC,0);
;  459          0727  5                                       RETURN FALSE;
;  460          0728  4                                       END;
;  461          0729  4
;  462          0730  4                               current_stmt = 1;
;  463          0731  4                               current_stmt_mode = TRUE;
;  464          0732  4                               dpc_entry = dpc_entry[add_one_byte];
;  465          0733  3                               END;
;  466          0734  3
;  467          0735  3                       [dst$k_end_stmt_mode]:
;  468          0736  4                               BEGIN
;  469          0737  4                               current_stmt = 1;
;  470          0738  4                               current_stmt_mode = FALSE;
;  471          0739  4                               dpc_entry = dpc_entry[add_one_byte];
;  472          0740  3                               END;
;  473          0741  3
;  474          0742  3                       [dst$k_set_linum_b]:
;  475          0743  4                               BEGIN
;  476          0744  4                               IF .current_mark NEQ line_closed
;  477          0745  4                               THEN
;  478          0746  5                                       BEGIN
;  479          0747  5                                       TBK$FAKE_MSG(TBK$_INVDSTREC,0);
;  480          0748  5                                       RETURN FALSE;
;  481          0749  4                                       END;
;  482          0750  4
;  483          0751  4                               current_line = .dpc_entry[next_uns_byte];
;  484          0752  4                               dpc_entry = dpc_entry[add_two_bytes];
;  485          0753  3                               END;
;  486          0754  3
;  487          0755  3                       [dst$k_set_linum]:
;  488          0756  4                               BEGIN
;  489          0757  4                               IF .current_mark NEQ line_closed
;  490          0758  4                               THEN
;  491          0759  5                                       BEGIN
;  492          0760  5                                       TBK$FAKE_MSG(TBK$_INVDSTREC,0);
;  493          0761  5                                       RETURN FALSE;
;  494          0762  4                                       END;
;  495          0763  4
;  496          0764  4                               current_line = .dpc_entry[next_uns_word];
;  497          0765  4                               dpc_entry = dpc_entry[add_three_bytes];
;  498          0766  3                               END;
;  499          0767  3
;  500          0768  3                       [dst$k_set_linum_l]:
;  501          0769  4                               BEGIN
;  502          0770  4                               IF .current_mark NEQ line_closed
;  503          0771  4                               THEN
;  504          0772  5                                       BEGIN
;  505          0773  5                                       TBK$FAKE_MSG(TBK$_INVDSTREC,0);
;  506          0774  5                                       RETURN FALSE;
;  507          0775  4                                       END;
;  508          0776  4
;  509          0777  4                               current_line = .dpc_entry[next_uns_long];
;  510          0778  4                               dpc_entry = dpc_entry[add_five_bytes];
;   511         0779  3                          END;
;   512         0780  3
;   513         0781  3                  [dst$k_set_stmtnum]:
;   514         0782  4                          BEGIN
;   515         0783  4                          current_stmt = .dpc_entry[next_uns_word];
;   516         0784  4                          dpc_entry = dpc_entry[add_three_bytes];
;   517         0785  3                          END;
;   518         0786  3
;   519         0787  3                  [dst$k_set_pc]:
;   520         0788  4                          BEGIN
;   521         0789  4                          IF .current_mark NEQ line_closed
;   522         0790  4                          THEN
;   523         0791  5                                  BEGIN
;   524         0792  5                                  TBK$FAKE_MSG(TBK$_INVDSTREC,0);
;   525         0793  5                                  RETURN FALSE;
;   526         0794  4                                  END;
;   527         0795  4
;   528         0796  4                          current_pc = .start_pc +
;   529         0797  4                                          .dpc_entry[next_uns_byte];
;   530         0798  4                          dpc_entry = dpc_entry[add_two_bytes];
;   531         0799  3                          END;
;   532         0800  3
;   533         0801  3                  [dst$k_set_pc_w]:
;   534         0802  4                          BEGIN
;   535         0803  4                          IF .current_mark NEQ line_closed
;   536         0804  4                          THEN
;   537         0805  5                                  BEGIN
;   538         0806  5                                  TBK$FAKE_MSG(TBK$_INVDSTREC,0);
;   539         0807  5                                  RETURN FALSE;
;   540         0808  4                                  END;
;   541         0809  4
;   542         0810  4                          current_pc = .start_pc +
;   543         0811  4                                          .dpc_entry[next_uns_word];
;   544         0812  4                          dpc_entry = dpc_entry[add_three_bytes];
;   545         0813  3                          END;
;   546         0814  3
;   547         0815  3                  [dst$k_set_pc_l]:
;   548         0816  4                          BEGIN
;   549         0817  4                          IF .current_mark NEQ line_closed
;   550         0818  4                          THEN
;   551         0819  5                                  BEGIN
;   552         0820  5                                  TBK$FAKE_MSG(TBK$_INVDSTREC,0);
;   553         0821  5                                  RETURN FALSE;
;   554         0822  4                                  END;
;   555         0823  4
;   556         0824  4                          current_pc = .start_pc +
;   557         0825  4                                          .dpc_entry[next_uns_long];
;   558         0826  4                          dpc_entry = dpc_entry[add_five_bytes];
;   559         0827  3                          END;
;   560         0828  3
;   561         0829  3
;   562         0830  3                  ! Set the current PC value to an absolute address.
;   563         0831  3                  !
;   564         0832  3                  [DST$K_SET_ABS_PC]:
;   565         0833  4                          BEGIN
;   566         0834  4                          IF .CURRENT_MARK NEQ LINE_CLOSED
;   567         0835  4                          THEN
```

```
5                         BEGIN
5                         TBK$FAKE_MSG(TBK$_INVDSTREC,0);
5                         RETURN FALSE;
4                         END;
4
4                 CURRENT_PC = .DPC_ENTRY[NEXT_UNS_LONG];
4                 DPC_ENTRY = DPC_ENTRY[ADD_FIVE_BYTES];
3                 END;
3
3         [dst$k_term]:
4                 BEGIN
4                 current_pc = .current_pc +
4                                 .dpc_entry[next_uns_byte];
4                 current_mark = line_closed;
4                 dpc_entry = dpc_entry[add_two_bytes];
4                 RETURN TRUE;
3                 END;
3
3         [dst$k_term_w]:
4                 BEGIN
4                 current_pc = .current_pc +
4                                 .dpc_entry[next_uns_word];
4                 current_mark = line_closed;
4                 dpc_entry = dpc_entry[add_three_bytes];
4                 RETURN TRUE;
3                 END;
3
3         [dst$k_term_l]:
4                 BEGIN
4                 current_pc = .current_pc +
4                                 .dpc_entry[next_uns_long];
4                 current_mark = line_closed;
4                 dpc_entry = dpc_entry[add_five_bytes];
4                 RETURN TRUE;
3                 END;
3
3
3         ! This is a standard delta_PC command if the value is
3         ! less than or equal to zero. Otherwise it is an error.
3         ! If okay, set next_pc value, update the dpc_entry,
3         ! and return with success.
3         !
3         [OUTRANGE]:
4                 BEGIN
4                 IF .dpc_entry [current_byte] LSS
4                                         dst$k_delta_pc_low
4                 OR .dpc_entry[current_byte] GTR
4                                         dst$k_delta_pc_high
4                 THEN
5                         BEGIN
5                         TBK$FAKE_MSG(TBK$_INVDSTREC,0);
5                         RETURN FALSE;
4                         END;
4
4                 IF .current_stmt_mode
4                 THEN
4                         current_stmt = .current_stmt + 1
```

```
;  625        0893  4                             ELSE
;  626        0894  4                                 current_line = .current_line +
;  627        0895  4                                                 .current_incr;
;  628        0896  4
;  629        0897  4                             current_pc = .current_pc -
;  630        0898  4                                     .dpc_entry [current_byte];
;  631        0899  4                             current_mark = line_open;
;  632        0900  4                             dpc_entry = dpc_entry [add_one_byte];
;  633        0901  4                             RETURN TRUE;
;  634        0902  3                             END;
;  635        0903  3
;  636        0904  3                         TES;
;  637        0905  2                     END;
;  638        0906  2
;  639        0907  2                 RETURN 0;
;  640        0908  1                 END;



                                         000C 00000 PROC_PC_CMD:
                                                           .WORD   Save R2,R3                          ; 0551
                          53      0000'  CF   9E 00002     MOVAB   DPC_ENTRY, R3                       ;
                          50        FC   B3   9A 00007 1$: MOVZBL  @DST_ENTRY, R0                      ; 0594
                          50        FC   A3   C0 0000B     ADDL2   DST_ENTRY, R0                       ;
                          50             63   D1 0000F     CMPL    DPC_ENTRY, R0                       ; 0593
                                         13   15 00012     BLEQ    3$                                  ;
                                    FC   A3   9F 00014     PUSHAB  DST_ENTRY                           ; 0597
                       0000V  CF         01   FB 00017     CALLS   #1, GET_NEXT_DPC                    ;
                              03         50   E8 0001C     BLBS    R0, 2$                              ;
                                       01E0   31 0001F     BRW     56$                                 ;
               63         FC  A3         02   C1 00022 2$: ADDL3   #2, DST_ENTRY, DPC_ENTRY            ; 0602
                              52         63   D0 00027 3$: MOVL    DPC_ENTRY, R2                       ; 0611
               14             01         62   8F 0002A     CASEB   (R2), #1, #20                       ;
 00BC          009B         0089       004F      0002E 4$: .WORD   8$-4$,-                             ;
 00FB          00E8         00DA       00CB      00036             16$-4$,-                            ;
 016D          0155         013D       0116      0003E             17$-4$,-                            ;
 017F          01B5         01A4       0136      00046             21$-4$,-                            ;
 0126          0106         00AD       006F      0004E             23$-4$,-                            ;
                                       01C4      00056             25$-4$,-                            ;
                                                                   27$-4$,-                            ;
                                                                   29$-4$,-                            ;
                                                                   33$-4$,-                            ;
                                                                   39$-4$,-                            ;
                                                                   41$-4$,-                            ;
                                                                   44$-4$,-                            ;
                                                                   37$-4$,-                            ;
                                                                   51$-4$,-                            ;
                                                                   52$-4$,-                            ;
                                                                   46$-4$,-                            ;
                                                                   13$-4$,-                            ;
                                                                   19$-4$,-                            ;
                                                                   31$-4$,-                            ;
                                                                   35$-4$,-                            ;
                                                                   53$-4$                              ;
                                         62   95 00058     TSTB    (R2)                                ; 0882
```

```
                         03  15 0005A           BLEQ    5$                                          ;
                       0154  31 0005C           BRW     47$                                         ;
          05      18     A3  E9 0005F  5$:      BLBC    CURRENT_STMT_MODE, 6$                       ; 0890
                  0C     A3  D6 00063           INCL    CURRENT_STMT                                ; 0892
                         05  11 00066           BRB     7$                                          ;
    08    A3      10     A3  C0 00068  6$:      ADDL2   CURRENT_INCR, CURRENT_LINE                  ; 0895
          50      00     B3  98 0006D  7$:      CVTBL   @DPC_ENTRY, R0                              ; 0898
    14    A3             50  C2 00071           SUBL2   R0, CURRENT_PC                              ;
    30    A3             01  D0 00075           MOVL    #1, CURRENT_MARK                            ; 0899
                         63  D6 00079           INCL    DPC_ENTRY                                   ; 0900
                         1D  11 0007B           BRB     12$                                         ; 0901
          05      18     A3  E9 0007D  8$:      BLBC    CURRENT_STMT_MODE, 9$                       ; 0621
                  0C     A3  D6 00081           INCL    CURRENT_STMT                                ; 0623
                         05  11 00084           BRB     10$                                         ;
    08    A3      10     A3  C0 00086  9$:      ADDL2   CURRENT_INCR, CURRENT_LINE                  ; 0626
    30    A3             01  D0 0008B  10$:     MOVL    #1, CURRENT_MARK                            ; 0628
          50      01     A2  3C 0008F           MOVZWL  1(R2), R0                                   ; 0630
    14    A3             50  C0 00093           ADDL2   R0, CURRENT_PC                              ;
          63             03  C0 00097  11$:     ADDL2   #3, DPC_ENTRY                               ; 0631
                       0161  31 0009A  12$:     BRW     55$                                         ; 0632
          05      18     A3  E9 0009D  13$:     BLBC    CURRENT_STMT_MODE, 14$                      ; 0642
                  0C     A3  D6 000A1           INCL    CURRENT_STMT                                ; 0644
                         05  11 000A4           BRB     15$                                         ;
    08    A3      10     A3  C0 000A6  14$:     ADDL2   CURRENT_INCR, CURRENT_LINE                  ; 0647
    30    A3             01  D0 000AB  15$:     MOVL    #1, CURRENT_MARK                            ; 0649
    14    A3      01     A2  C0 000AF           ADDL2   1(R2), CURRENT_PC                           ; 0651
                       0144  31 000B4           BRW     54$                                         ; 0652
          50      01     A2  9A 000B7  16$:     MOVZBL  1(R2), R0                                   ; 0662
    08    A3             50  C0 000BB           ADDL2   R0, CURRENT_LINE                            ;
          7F      18     A3  E9 000BF           BLBC    CURRENT_STMT_MODE, 32$                      ; 0663
    0C    A3             01  D0 000C3           MOVL    #1, CURRENT_STMT                            ;
                         79  11 000C7           BRB     32$                                         ; 0664
          04      18     A3  E9 000C9  17$:     BLBC    CURRENT_STMT_MODE, 18$                      ; 0673
    0C    A3             01  D0 000CD           MOVL    #1, CURRENT_STMT                            ;
          50      01     A2  3C 000D1  18$:     MOVZWL  1(R2), R0                                   ; 0674
    08    A3             50  C0 000D5           ADDL2   R0, CURRENT_LINE                            ;
                         77  11 000D9           BRB     34$                                         ; 0675
          04      18     A3  E9 000DB  19$:     BLBC    CURRENT_STMT_MODE, 20$                      ; 0684
    0C    A3             01  D0 000DF           MOVL    #1, CURRENT_STMT                            ;
    08    A3      01     A2  C0 000E3  20$:     ADDL2   1(R2), CURRENT_LINE                         ; 0685
                         78  11 000E8           BRB     36$                                         ; 0686
          04      18     A3  E9 000EA  21$:     BLBC    CURRENT_STMT_MODE, 22$                      ; 0695
    0C    A3             01  D0 000EE           MOVL    #1, CURRENT_STMT                            ;
    10    A3      01     A2  9A 000F2  22$:     MOVZBL  1(R2), CURRENT_INCR                         ; 0696
                         49  11 000F7           BRB     32$                                         ; 0697
          04      18     A3  E9 000F9  23$:     BLBC    CURRENT_STMT_MODE, 24$                      ; 0706
    0C    A3             01  D0 000FD           MOVL    #1, CURRENT_STMT                            ;
    10    A3      01     A2  3C 00101  24$:     MOVZWL  1(R2), CURRENT_INCR                         ; 0707
                         61  11 00106           BRB     38$                                         ; 0708
          04      18     A3  E9 00108  25$:     BLBC    CURRENT_STMT_MODE, 26$                      ; 0716
    0C    A3             01  D0 0010C           MOVL    #1, CURRENT_STMT                            ;
    10    A3             01  D0 00110  26$:     MOVL    #1, CURRENT_INCR                            ; 0717
                         1A  11 00114           BRB     30$                                         ; 0718
          01      30     A3  D1 00116  27$:     CMPL    CURRENT_MARK, #1                            ; 0723
                         03  13 0011A           BEQL    28$                                         ;
                       0080  31 0011C           BRW     45$                                         ;
    0C    A3             01  D0 0011F  28$:     MOVL    #1, CURRENT_STMT                            ; 0730
```

```
                    18   A3            01  D0 00123       MOVL    #1, CURRENT_STMT_MODE                     ; 0731
                                       07  11 00127       BRB     30$                                       ; 0732
                    0C   A3            01  D0 00129 29$:  MOVL    #1, CURRENT_STMT                          ; 0737
                                 18    A3  D4 0012D       CLRL    CURRENT_STMT_MODE                         ; 0738
                                       63  D6 00130 30$:  INCL    DPC_ENTRY                                 ; 0739
                                       65  11 00132       BRB     43$                                       ; 0611
                         02      30    A3  D1 00134 31$:  CMPL    CURRENT_MARK, #2                          ; 0744
                                       79  12 00138       BNEQ    47$                                       ;
                         50            63  D0 0013A       MOVL    DPC_ENTRY, R0                             ; 0751
                    08   A3      01    A0  9A 0013D       MOVZBL  1(R0), CURRENT_LINE                       ;
                                       3A  11 00142 32$:  BRB     40$                                       ; 0752
                         02      30    A3  D1 00144 33$:  CMPL    CURRENT_MARK, #2                          ; 0757
                                       69  12 00148       BNEQ    47$                                       ;
                         50            63  D0 0014A       MOVL    DPC_ENTRY, R0                             ; 0764
                    08   A3      01    A0  3C 0014D       MOVZWL  1(R0), CURRENT_LINE                       ;
                                       42  11 00152 34$:  BRB     42$                                       ; 0765
                         02      30    A3  D1 00154 35$:  CMPL    CURRENT_MARK, #2                          ; 0770
                                       59  12 00158       BNEQ    47$                                       ;
                         50            63  D0 0015A       MOVL    DPC_ENTRY, R0                             ; 0777
                    08   A3      01    A0  D0 0015D       MOVL    1(R0), CURRENT_LINE                       ;
                                       68  11 00162 36$:  BRB     49$                                       ; 0778
                    0C   A3      01    A2  3C 00164 37$:  MOVZWL  1(R2), CURRENT_STMT                       ; 0783
                                       2B  11 00169 38$:  BRB     42$                                       ; 0784
                         02      30    A3  D1 0016B 39$:  CMPL    CURRENT_MARK, #2                          ; 0789
                                       42  12 0016F       BNEQ    47$                                       ;
                         50            63  D0 00171       MOVL    DPC_ENTRY, R0                             ; 0797
                         51      01    A0  9A 00174       MOVZBL  1(R0), R1                                 ;
                    14   A3      04  B341  9E 00178       MOVAB   @START_PC[R1], CURRENT_PC                 ;
                         63            02  C0 0017E 40$:  ADDL2   #2, DPC_ENTRY                             ; 0798
                                       4C  11 00181       BRB     50$                                       ; 0611
                         02      30    A3  D1 00183 41$:  CMPL    CURRENT_MARK, #2                          ; 0803
                                       2A  12 00187       BNEQ    47$                                       ;
                         50            63  D0 00189       MOVL    DPC_ENTRY, R0                             ; 0811
                         51      01    A0  3C 0018C       MOVZWL  1(R0), R1                                 ;
                    14   A3      04  B341  9E 00190       MOVAB   @START_PC[R1], CURRENT_PC                 ;
                         63            03  C0 00196 42$:  ADDL2   #3, DPC_ENTRY                             ; 0812
                                       34  11 00199 43$:  BRB     50$                                       ; 0611
                         02      30    A3  D1 0019B 44$:  CMPL    CURRENT_MARK, #2                          ; 0817
                                       12  12 0019F 45$:  BNEQ    47$                                       ;
                         50            63  D0 001A1       MOVL    DPC_ENTRY, R0                             ; 0825
  14   A3           04   A3      01    A0  C1 001A4       ADDL3   1(R0), START_PC, CURRENT_PC               ;
                                       1F  11 001AB       BRB     49$                                       ; 0826
                         02      30    A3  D1 001AD 46$:  CMPL    CURRENT_MARK, #2                          ; 0834
                                       11  13 001B1       BEQL    48$                                       ;
                                       7E  D4 001B3 47$:  CLRL    -(SP)                                     ; 0837
                           00098332    8F  DD 001B5       PUSHL   #623410                                   ;
                0000000G  00           02  FB 001BB       CALLS   #2, TBK$FAKE_MSG                          ;
                                       3E  11 001C2       BRB     56$                                       ; 0838
                         50            63  D0 001C4 48$:  MOVL    DPC_ENTRY, R0                             ; 0841
                    14   A3      01    A0  D0 001C7       MOVL    1(R0), CURRENT_PC                         ;
                         63            05  C0 001CC 49$:  ADDL2   #5, DPC_ENTRY                             ; 0842
                                     FE35  31 001CF 50$:  BRW     1$                                        ; 0611
                         50      01    A2  9A 001D2 51$:  MOVZBL  1(R2), R0                                 ; 0848
                    14   A3            50  C0 001D6       ADDL2   R0, CURRENT_PC                            ;
                    30   A3            02  D0 001DA       MOVL    #2, CURRENT_MARK                          ; 0849
                         63            02  C0 001DE       ADDL2   #2, DPC_ENTRY                             ; 0850
                                       1B  11 001E1       BRB     55$                                       ; 0851
```

```
                 50        01    A2  3C 001E3 52$:     MOVZWL  1(R2), R0                               ; 0857
           14    A3              50  C0 001E7          ADDL2   R0, CURRENT_PC                          ;
           30    A3              02  D0 001EB          MOVL    #2, CURRENT_MARK                        ; 0858
                               FEA5  31 001EF          BRW     11$                                     ; 0859
           14    A3        01    A2  C0 001F2 53$:     ADDL2   1(R2), CURRENT_PC                       ; 0866
           30    A3              02  D0 001F7          MOVL    #2, CURRENT_MARK                        ; 0867
                 63              05  C0 001FB 54$:     ADDL2   #5, DPC_ENTRY                           ; 0868
                 50              01  D0 001FE 55$:     MOVL    #1, R0                                  ; 0869
                                     04 00201          RET                                             ;
                                 50  D4 00202 56$:     CLRL    R0                                      ; 0908
                                     04 00204          RET                                             ;

; Routine Size:  517 bytes,    Routine Base:  TBK$CODE + 00DC
```

```
1 ROUTINE get_next_dpc (dst_rec_ptr) =     ! gets next PC correlation record
1
1 !++
1 ! Functional description:
1 !       Reads DST records until either no more exist, a module end
1 !       record is seen, or another PC correlation record is seen. In
1 !       the first two cases, a FALSE return is taken. In the third
1 !       case, the address of the new record and a success return is
1 !       taken.
1 !
1 ! Inputs:
1 !       dst_rec_ptr     - pointer for new DST PC correlation record
1 !
1 !
1 ! Implicit inputs:
1 !       the routine tbk$get_nxt_dst is set up to return
1 !       each DST record sequentially, and the last record
1 !       that it returned was a PC correlation record.
1 !
1 ! Implicit outputs:
1 !       tbk$get_nxt_dst is now set up to return the next record after
1 !       the returned record or the next record after the record that
1 !       caused this routine to fail.
1 !
1 ! Routine value:
1 !       true or false
1 !
1 ! Side effects:
1 !       none
1 !--
1
2       BEGIN
2
2       BIND
2               dst_entry       = .dst_rec_ptr : REF dst$record;
2
2       LOCAL
2               dst_rec_id;
2
2       REPEAT
3               BEGIN
3               dst_entry = tbk$get_nxt_dst (dst_rec_id);
3               IF .dst_entry EQL 0
3               THEN RETURN FALSE;
3               IF .dst_entry [dst$b_type] EQL dst$k_modend
3               THEN RETURN FALSE;
3               IF .dst_entry [dst$b_type] EQL dst$k_line_num
3               OR .dst_entry [dst$b_type] EQL dst$k_line_num_rel_r11
3               THEN RETURN TRUE;
2               END;
2       RETURN FALSE;
1       END;
```

```
                                        0000 00000 GET_NEXT_DPC:
                                                         .WORD   Save nothing                              ; 0909
                     5E           04  C2 00002           SUBL2   #4, SP                                    ;
                                  5E  DD 00005 1$:       PUSHL   SP                                        ; 0950
         00000000G   00           01  FB 00007           CALLS   #1, TBK$GET_NXT_DST                       ;
                04   BC           50  D0 0000E           MOVL    R0, @DST_REC_PTR                          ;
                     50      04   BC  D0 00012           MOVL    @DST_REC_PTR, R0                          ; 0951
                                  19  13 00016           BEQL    3$                                        ;
                BD   8F      01   A0  91 00018           CMPB    1(R0), #189                               ; 0953
                                  12  13 0001D           BEQL    3$                                        ;
                B9   8F      01   A0  91 0001F           CMPB    1(R0), #185                               ; 0955
                                  07  13 00024           BEQL    2$                                        ;
                B6   8F      01   A0  91 00026           CMPB    1(R0), #182                               ; 0956
                                  D8  12 0002B           BNEQ    1$                                        ;
                     50           01  D0 0002D 2$:       MOVL    #1, R0                                    ; 0957
                                      04 00030           RET                                               ;
                                  50  D4 00031 3$:       CLRL    R0                                        ; 0960
                                      04 00033           RET                                               ;

; Routine Size:  52 bytes,    Routine Base:  TBK$CODE + 02E1
;  695          0961  1 END
;  696          0962  0 ELUDOM




;                                        PSECT SUMMARY
;
;        Name                       Bytes                           Attributes
;
;   TBK$OWN                            60  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   TBK$CODE                          789  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  .EL,  CON,  PIC,ALIGN(0)




;                               Library Statistics
;
;                                              -------- Symbols --------       Pages       Processing
;        File                                  Total    Loaded   Percent       Mapped      Time
;
;   _$255$DUA28:[SYSLIB]LIB.L32;1              18619         3         0        1000         00:01.7
;   _$255$DUA28:[TRACE.OBJ]TBKLIB.L32;1          157         4         2          14         00:00.2
;   _$255$DUA28:[TRACE.OBJ]STRUCDEF.L32;1         32         0         0           7         00:00.1
;   _$255$DUA28:[TRACE.OBJ]TBKDST.L32;1          414       131        31          30         00:00.3




;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:TBKDPC/OBJ=OBJ$:TBKDPC MSRC$:TBKDPC/UPDATE=(ENH$:TBK^PC)

; Size:          789 code + 60 data bytes
; Run Time:         00:22.0
; Elapsed Time:     01:14.4
; Lines/CPU Min:     2618
; Lexemes/CPU-Min: 20537
; Memory Used:  232 pages
; Compilation Complete
```

0401 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

TBKLIB
LIS

TBKBAS
LIS

TBKDST
LIS

TBKINT
LIS

TBKDPC
LIS