

[illegible]

TTTTTTTTT	BBBBBBBB	KK	KK	SSSSSSSS	YY	YY	MM	MM	
TTTTTTTTT	BBBBBBBB	KK	KK	SSSSSSSS	YY	YY	MM	MM	
TT	BB	KK	KK	SS	YY	YY	MMMM	MMMM	
TT	BB	KK	KK	SS	YY	YY	MMMM	MMMM	
TT	BB	KK	KK	SS	YY	YY	MM	MM	
TT	BB	KK	KK	SS	YY	YY	MM	MM	
TT	BBBBBBBB	KKKKKK	SSSSSS		YY	YY	MM	MM	
TT	BBBBBBBB	KKKKKK	SSSSSS		YY	YY	MM	MM	
TT	BB	KK	SS		YY	YY	MM	MM	
TT	BB	KK	SS		YY	YY	MM	MM	
TT	BB	KK	SS		YY	YY	MM	MM	
TT	BB	KK	SS		YY	YY	MM	MM	
TT	BBBBBBBB	KK	SSSSSSSS		YY	YY	MM	MM	
TT	BBBBBBBB	KK	SSSSSSSS		YY	YY	MM	MM	

....

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

```
1 0001 0 MODULE TBKSYM ( IDENT = 'V04-000' ) =
2 0002 1 BEGIN
3 0003 1
4 0004 1
5 0005 1 *****
6 0006 1 *
7 0007 1 *   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
8 0008 1 *   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
9 0009 1 *   ALL RIGHTS RESERVED.
10 0010 1 *
11 0011 1 *   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
12 0012 1 *   ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
13 0013 1 *   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
14 0014 1 *   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
15 0015 1 *   OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
16 0016 1 *   TRANSFERRED.
17 0017 1 *
18 0018 1 *   THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
19 0019 1 *   AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
20 0020 1 *   CORPORATION.
21 0021 1 *
22 0022 1 *   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
23 0023 1 *   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
24 0024 1 *
25 0025 1 *****
26 0026 1
27 0027 1
28 0028 1
29 0029 1 ++
30 0030 1 FACILITY:
31 0031 1   TRACEBACK
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1   This module contains all routines used by
35 0035 1   traceback which actually look at the DST.
36 0036 1   These routines interface to reality via
37 0037 1   those in TBKINT.B32
38 0038 1
39 0039 1 Version      1.0
40 0040 1
41 0041 1 ENVIRONMENT:
42 0042 1   This module runs on VAX under VAX/VMS, user mode, non-AST level.
43 0043 1
44 0044 1 Author:
45 0045 1   Kevin Pammett, Creation Date: 18-jan-78.
46 0046 1
47 0047 1 MODIFIED BY:
48 0048 1   Dale Roedger, 09 November 1978
49 0049 1   Ping Sager,   30 August   1983
50 0050 1
51 0051 1 Revision History:
52 0052 1
53 0053 1   02      24-feb-78      KGP      -EXC_TYPE has to be an OWN so that
54 0054 1   03      28-FEB-78      KGP      -we only initialize it once.
55 0055 1   04      2-mar-78      KGP      -SYMBOLIZE is now truly NOVALUE and
56 0056 1   05      2-mar-78      KGP      -it works OK if the DST doesn't exist.
57 0057 1   06      2-mar-78      KGP      -Beginning exception_type for
```

TBKSVM
V04-000

F 5

16-Sep-1984 02:18:08
14-Sep-1984 13:20:26

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[TRACE.SRC]TBKSVM.B32;1

Page 2
(1)

58	0058	1			
59	0059	1			
60	0060	1			
61	0061	1			
62	0062	1			
63	0063	1	05	8-MAR-78	KGP
64	0064	1	06	10-mar-78	KGP
65	0065	1			
66	0066	1			
67	0067	1	07	14-mar-78	KGP
68	0068	1			
69	0069	1	08	26-APR-78	DAR
70	0070	1			
71	0071	1	09	15-JUN-78	DAR
72	0072	1	1.01	09-NOV-78	DAR
73	0073	1			
74	0074	1	1.02	3-NOV-79	JBD
75	0075	1	1.03	14-jan-80	JBD
76	0076	1	1.04	28-Jan-80	JBD
77	0077	1			
78	0078	1	4.0	30-Aug-83	PS
79	0079	1			
80	0080	1	--		

FTN PC correlation is now decided
in BAS (is no longer local to DPC)
-We reject a symbolization for which
the relative PC == absolute PC
i.e. symbols whose value is 0.
-NEW REQUIRE FILE NAMES
-SYMBOLIZE now knows about end-of-RTN
entries and uses this to know about
the length of ROUTINES.
-Call to PC_TO_LINE now has
one less parameter.
Modified require and library directives
for native build.
Changed all DBG\$ symbols to TBK\$.
Added new DST types EPT and R11,
and modified output for COBOL.
Added statement number support.
Added nested routine support.
Fixed some nesting problems. Was corrupting
the best routine.
Prefer routine to psect be the best value,
if there is routine begin

```

82      0081 1  ! Table of contents:
83      0082 1
84      0083 1  FORWARD ROUTINE
85      0084 1      TBK$SYMBOLIZE : NOVALUE;
86      0085 1
87      0086 1
88      0087 1  ! INCLUDE FILES:
89      0088 1
90      0089 1  REQUIRE 'SRC$:TBKPROLOG.REQ';
91      0361 1
92      0362 1  EXTERNAL ROUTINE
93      0363 1      TBK$FAO_OUT: NOVALUE;
94      0364 1
95      0365 1
96      0366 1  ! MACROS:
97      0367 1
98      0368 1
99      0369 1
100     0370 1  ! EQUATED SYMBOLS:
101     0371 1
102     0372 1  LITERAL
103     0373 1      TBK_ANY          = 0,      ! Turn on if any diagnostics are on.
104     0374 1      TBK_SYM1        = 0,      ! Specific output in TBK$SYMBOLIZE
105     0375 1      TBK_SYM2        = 0,      ! List DSTs in TBK$SYMBOLIZE
106     0376 1      TBK_SYM3        = 0,      ! How to symbolize.
107     0377 1  LITERAL
108     0378 1      MAX_NEST_DEPTH  = 100;    ! Maximum number of nested routines to
109     0379 1      ! reliably traceback.
110     0380 1
111     L 0381 1  %IF TBK_ANY
112     U 0382 1  %THEN
113     U 0383 1  FORWARD ROUTINE
114     U 0384 1      PR_CS : NOVALUE;
115     U 0385 1      ! Routine to print diagnostics during
116     0386 1      ! debug of traceback
117     0387 1  %FI
118     0388 1
119     0389 1  ! OWN STORAGE:
120     0390 1
121     0391 1
122     0392 1
123     0393 1
124     0394 1  ! EXTERNAL REFERENCES:
125     0395 1
126     0396 1  EXTERNAL ROUTINE
127     0397 1      tbk$pc_to_line,      ! FORTRAN PC-to-line_number correlation.
128     0398 1      tbk$get_dst_rec,     ! Make a certain DST record available.
129     0399 1      tbk$get_nxt_dst,     ! Make the next DST record available.
130     0400 1      tbk$positon_dst;     ! Make a certain DST record available
131     0401 1      ! and set up for tbk$get_nxt_dst
132     0402 1
133     0403 1
134     0404 1  EXTERNAL
135     0405 1      TBK$GL_EXC_TYPE,      ! Initial FAULT/TRAP type for PC
136     0406 1      ! correlation.
137     0407 1      TBK$MODULE_CS : CS_POINTER,
138     0408 1      TBK$ROUTINE_CS : CS_POINTER,
```

TBKSVM
V04-000

H 5
16-Sep-1984 02:18:08
14-Sep-1984 13:20:26

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[TRACE.SRC]TBKSVM.B32;1 Page 4 (2)

: 139
: 140
: 141
: 142
: 143

0409 1
0410 1
0411 1
0412 1
0413 1

TBK\$GL_STMT,
TBK\$GL_LINE,
TBK\$REC_PC,

TBK\$MODULE_DST : REF DST\$RECORD;

```
145 0414 1 GLOBAL ROUTINE tbk$symbolize(value) : NOVALUE =
146 0415 1
147 0416 1 !++
148 0417 1 Functional Specification:
149 0418 1
150 0419 1 Given a (supposed) PC value, look thru the
151 0420 1 entire DST (of TBT records only) and find the best
152 0421 1 possible symbolization for it.
153 0422 1
154 0423 1 Routine Value:
155 0424 1 NOVALUE
156 0425 1
157 0426 1 Side Effects:
158 0427 1
159 0428 1 The entire DST is scanned.
160 0429 1 GLOBAL variables are used for communication with
161 0430 1 other traceback routines.
162 0431 1 --
163 0432 2 BEGIN
164 0433 2 LOCAL
165 0434 2 RTN_RCD, ! Type of last routine record
166 0435 2 CURRENT_RTN : REF DST$RECORD,
167 0436 2 IN_MODULE : REF DST$RECORD,
168 0437 2 BEST_VALUE,
169 0438 2 BEST_DST : REF DST$RECORD,
170 0439 2 BEST_DST_TYPE,
171 0440 2 BEST_MODULE : REF DST$RECORD,
172 0441 2 NT_COUNT,
173 0442 2 DST_REC_ID,
174 0443 2 DST_REC_RD : REF DST$RECORD,
175 0444 2 RTN_NEST_VECTOR : VECTOR[ MAX_NEST_DEPTH ],
176 0445 2 RTN_NESTING;
177 0446 2
178 L 0447 2 XIF TBK_SYM3
179 U 0448 2 XTHEN
180 0449 2 $FAO TT_OUT('symbolize !XL as ',..value);
181 0450 2 XFI
182 0451 2 ! Assume that no symbolization can be found.
183 0452 2 ! Also, explicitly remove all previous
184 0453 2 ! symbolizations so that what is left set after
185 0454 2 ! this one definately refers to this symbolization.
186 0455 2
187 0456 2 BEST_MODULE = 0;
188 0457 2 BEST_VALUE = 0;
189 0458 2 BEST_DST_TYPE = 0;
190 0459 2 CURRENT_RTN = 0;
191 0460 2 TBK$MODULE_CS = 0;
192 0461 2 TBK$ROUTINE_CS = 0;
193 0462 2 TBK$GL_STMT = TBK$GL_LINE = 0;
194 0463 2 TBK$REC_PC = .VALUE;
195 0464 2 TBK$MODULE_DST = 0;
196 0465 2
197 0466 2 ! Position the DST to the beginning.
198 0467 2
199 0468 2 IF( NOT TBK$POSITON_DST(0) )
200 0469 2 THEN
201 0470 2 RETURN;
```

```
202      0471 2
203      0472
204      0473
205      0474
206      0475
207      0476
208      0477
209      0478
210      0479
211      0480 %IF TBK_SYM2
212      0481 %THEN
213      0482
214      0483
215      0484
216      0485
217      0486
218      0487
219      0488
220      0489
221      0490 $FAO TT_OUT( 'DST Rec Id=!XL, is at !XL, for !UD bytes.'
222      0491      .DST_REC_ID, .DST_RECRD, .DST_RECRD[dst$b_length] );
223      0492
224      0493 ! Dump the record in bytes.
225      0494
226      0495 INCR I FROM 0 TO .DST_RECRD[dst$b_length]
227      0496 DO
228      0497     $FAO TT_OUT('!XB ',.DST_RECRD[ .I, 0, 8, 0 ] );
229      0498
230      0499 %FI
231      0500
232      0501 CASE .DST_RECRD[dst$b_type] FROM dst$k_lowest TO dst$k_highest OF
233      0502
234      0503 SET
235      0504
236      0505 [dst$k_modbeg]:      ! Module Begin Record.
237      0506
238      0507     BEGIN
239      0508     IN_MODULE = .DST_RECRD;
240      0509     END;
241      0510
242      0511 [dst$k_modend]:      ! Module End Record.
243      0512
244      0513     BEGIN
245      0514     IN_MODULE = 0;
246      0515     END;
247      0516
248      0517 [dst$k_rtnbeg]:      ! Routine DSTs.
249      0518
250      0519     BEGIN
251      0520     LOCAL
252      0521         RBEGIN; ! Address where routine begins.
253      0522
254      0523     RTN_RCD = .DST_RECRD[dst$b_type];
255      0524     CURRENT_RTN = .DST_RECRD;
256      0525     RBEGIN = .DST_RECRD[dst$[ value];
257      0526     IF .RTN_NESTING LSS MAX_NEST_DEPTH
258      0527     THEN
```

```
259      0528      5      BEGIN      ! Push current routine.
260      0529      5      %IF TBK_SYM1 %THEN
261      0530      5      $FAO TT_OUT('Pushing routine !XL', .current_rtn);
262      0531      5      $FAO TT_OUT('      index: !SL', .rtn_nesting);
263      0532      5      %FI
264      0533      5      RTN_NEST_VECTOR[.RTN_NESTING] = .CURRENT_RTN;
265      0534      5      RTN_NESTING = .RTN_NESTING + 1;
266      0535      5
267      0536      5
268      0537      5      ! In macro there is no routine end record, so, we need to record
269      0538      5      ! the best value at this point as well.
270      0539      5      :
271      0540      5      IF .RBEGIN LEQA .VALUE
272      0541      5      THEN
273      0542      6      BEGIN
274      0543      6      IF (.BEST_VALUE EQL 0) OR
275      0544      6      ((.VALUE - .BEST_VALUE) GEQ
276      0545      7      (.VALUE - .RBEGIN))
277      0546      6      THEN
278      0547      7      BEGIN
279      0548      7      BEST_MODULE = .IN_MODULE;
280      0549      7      BEST_VALUE = .RBEGIN;
281      0550      7      BEST_DST = .CURRENT_RTN;
282      0551      7      BEST_DST_TYPE = .DST_REC'D[DST$B_TYPE];
283      0552      6      END;
284      0553      5      END;
285      0554      5
286      0555      4      END;
287      0556      4
288      0557      4      %IF TBK_SYM1
289      0558      4      %THEN
290      0559      4      $FAO TT_OUT('routine begins: !XL', .RBEGIN);
291      0560      4      $FAO TT_OUT('best value: !XL', .BEST_VALUE);
292      0561      4      $FAO TT_OUT('best dst: !XL', .BEST_DST);
293      0562      4      $FAO TT_OUT('best dst type: !XL', .BEST_DST_TYPE);
294      0563      4      %FI
295      0564      4      END;
296      0565      4
297      0566      4      [dst$k_lblorlit,      ! label or literal
298      0567      4      dst$k_label,      ! labels.
299      0568      4      dst$k_entry,      ! Entry point records.
300      0569      4      dst$k_line_num,      ! delta-PC tables.
301      0570      4      dst$k_line_num_rel_r11,      ! Thread / line correlation tables.
302      0571      4      dst$k_blif[d]:      ! FIELD records.
303      0572      4
304      0573      4      ;      ! These records are not used in TRACEBACK.
305      0574      4
306      0575      4      [dst$k_rtnend]:      ! End of Routine
307      0576      4      BEGIN
308      0577      4      ! When we hit an end-of-routine record, we check the
309      0578      4      ! start and ending addresses of the routine at the
310      0579      4      ! top of the stack. If the PC is within these addresses,
311      0580      4      ! then this is the routine we want.
312      0581      4      LOCAL
313      0582      4      RBEGIN;      ! Address where routine begins
314      0583      4
315      0584      4      ! Note that the current routine is always the top of
```

```

316      0585 4
317      0586 4
318      0587 4
319      U 0588 4 %IF TBK_SYM1 %THEN
320      U 0589 4
321      U 0590 4
322      U 0591 4
323      U 0592 4
324      U 0593 4
325      U 0594 4
326      0595 4 %FI
327      0596 4
328      0597 4
329      0598 4
330      0599 4
331      0600 4
332      0601 4
333      0602 5
334      0603 5
335      0604 5
336      0605 5
337      L 0606 5 %IF TBK_SYM1
338      U 0607 5 %THEN
339      U 0608 5
340      U 0609 5
341      U 0610 5
342      U 0611 5
343      0612 5 %FI
344      0613 5
345      0614 4
346      0615 4
347      0616 4
348      0617 4
349      0618 4
350      0619 4
351      0620 4
352      0621 4
353      0622 5
354      0623 5
355      0624 5
356      0625 5
357      0626 5
358      0627 5
359      0628 5
360      0629 5
361      0630 5
362      U 0631 5 %IF TBK_SYM1 %THEN
363      U 0632 5
364      U 0633 5
365      0634 5 %FI
366      0635 4
367      0636 4
368      0637 4
369      0638 4
370      0639 4
371      0640 4
372      0641 4

! the stack.
CURRENT_RTN = .RTN_NEST_VECTOR[.RTN_NESTING - 1];
RBEGIN = .CURRENT_RTN[dst$l_value];

IF( .CURRENT_RTN NEQ 0 )
THEN
    $FAO TT_OUT('routine ends at !XL',
                .dst_recrd[dst$l_value] +
                .current_rtn[dst$l_value]);

! Now see if this routine contains the PC (.VALUE)
IF .value LEQA .RBEGIN + .dst_recrd[dst$l_value]
AND .RBEGIN LEQA .VALUE
THEN
    BEGIN ! This routine contains the PC. Mark it.
        best_module = .IN_MODULE;
        best_value = .RBEGIN;
        best_dst = .CURRENT_RTN;

$FAO TT_OUT('routine begins: !XL', .RBEGIN);
$FAO TT_OUT('best value: !XL', .BEST_VALUE);
$FAO TT_OUT('best dst: !XL', .BEST_DST);
$FAO TT_OUT('best dst type: !XL', .BEST_DST_TYPE);

    EXITLOOP;
END;

! If we got this far, then the PC is not within the
! current routine. So we POP the routine from the
! stack.
IF .RTN_NESTING GTR 0
THEN
    BEGIN ! Pop the last routine
        RTN_NESTING = .RTN_NESTING - 1;
        RTN_NEST_VECTOR[.RTN_NESTING] = 0;

        CURRENT_RTN = .RTN_NEST_VECTOR[.RTN_NESTING];
        BEST_MODULE = 0;
        BEST_VALUE = 0;
        BEST_DST = 0;
        BEST_DST_TYPE = 0;

        $FAO TT_OUT('Popped to routine !XL', .CURRENT_RTN);
        $FAO TT_OUT('index !SL', .RTN_NESTING);

    END;

RTN_RCD = .DST_RECRD[dst$b_type];
END;

[dst$k_psect]: ! Psect DSTs.
BEGIN
```

```

373      0642  4
374      0643  4
375      0644  4
376      0645  4
377      0646  4
378      0647  4
379      0648  4
380      0649  4
381      0650  4
382      0651  5
383      0652  5
384      0653  4
385      0654  4
386      0655  4
387      0656  4
388      0657  4 %IF TBK_SYM1
389      0658  4 %THEN
390      0659  4
391      0660  4
392      0661  4 %FI
393      0662  6
394      0663  5
395      0664  6
396      0665  5
397      0666  6
398      0667  5
399      0668  4
400      0669  4
401      0670  4
402      0671  4
403      0672  4
404      0673  5
405      0674  4
406      0675  5
407      0676  5
408      0677  5
409      0678  5
410      0679  5
411      0680  5
412      0681  5
413      0682  4
414      0683  4 %IF TBK_SYM1 %THEN
415      0684  4
416      0685  4
417      0686  4 %FI
418      0687  3
419      0688  3
420      0689  3
421      0690  4
422      0691  4
423      0692  3
424      0693  3
425      0694  3
426      0695  3
427      0696  4
428      0697  4
429      0698  4

LOCAL
    PBEGIN,
    PEND;
BIND
    PSECT_LENGTH =
        ! Pick up the field length, which
        ! is after the NAME so must be
        ! dynamically located.
        (.DST_REC RD[dst$b_name] ! The symbol-name count,
        + DST_REC RD[dst$b_name] ! plus its address,
        + 1) : LONG;
        ! addresses the LENGTH.
PBEGIN = .DST_REC RD[dst$l_value];
PEND = .DST_REC RD[dst$l_value] + .PSECT_LENGTH + 1;

$FAO TT_OUT('PSECT begins: !XL, ends !XL',
    .PBEGIN,.PEND);

IF( (.PBEGIN LEQA .VALUE)
    AND
    (.PEND GEQA .VALUE)
    AND
    (.pbegin GTRA 0 )
    )
THEN
    ! Adopt the new one only if it is
    ! better than a previous match.
    IF( .PBEGIN GEQA .BEST_VALUE ) AND
        ( .BEST_DST_TYPE NEQ DST$K_RTNBEG )
    THEN
        BEGIN
            ! This P-sect is better so adopt it.
            BEST_MODULE = .IN_MODULE;
            BEST_VALUE = .PBEGIN;
            BEST_DST = .DST_REC RD;
            END;
        $FAO TT_OUT('best value !XL', .best_value);
        $FAO TT_OUT('best dst !XL', .best_dst);
    END;

[INRANGE] :
    BEGIN
    0;
    END;
    ! Other types of records get ignored

[OUTRANGE]:
    BEGIN
    ! The only reason for not making the "SRM types"
```

```

430      0699      4      ! part of the above CASE is because of the huge
431      0700      4      ! case table which gets generated otherwise.
432      0701      4
433      0702      3      IF( .DST_RECIRD[dst$b_type] EQL DSC$K_DTYPE_Z )
434      0703      4      THEN
435      0704      5          BEGIN
436      0705      5              ! BLISS type ZERO records.
437      0706      5              ! Whatever symbol this is, it contributes
438      0707      5              ! a name, for sure, and either a literal
439      0708      5              ! or a static. We assume the worst!
440      0709      5
441      0710      5              NT_COUNT = .NT_COUNT +1;
442      0711      5          END
443      0712      5      ELSE
444      0713      4      IF( .DST_RECIRD[dst$b_type] LEQ dsc$k_dtype_highest)
445      0714      5          AND
446      0715      4          ( .DST_RECIRD[dst$b_type] GEQ dsc$k_dtype_lowest)
447      0716      5      THEN
448      0717      4          BEGIN
449      0718      5              ! These types are candidates for
450      0719      5              ! the LVT and NT tables only.
451      0720      5
452      0721      5              NT_COUNT = .NT_COUNT +1;
453      0722      5          END;
454      0723      4      END;
455      0724      4      TES;
456      0725      4
457      0726      3      ! Go back and process the next DST record.
458      0727      3
459      0728      3      END;
460      0729      3
461      0730      3
462      0731      2      END;
463      0732      2
464      0733      2      %IF TBK_SYM3
465      0734      2      %THEN
466      0735      2          IF( .BEST_MODULE NEQ 0 )
467      0736      2          THEN
468      0737      2              $FAO_TT_OUT(' module !AC, symbol !AC, + !XL',
469      0738      2                  best_module[dst$b_name],
470      0739      2                  best_dst[dst$b_name],
471      0740      2                  .value - .best_value)
472      0741      2          ELSE
473      0742      2              $FAO_TT_OUT('*****');
474      0743      2          %FI
475      0744      2          ! Pass back info via the TBK globals.
476      0745      2
477      0746      2          IF( .BEST_MODULE NEQ 0 )
478      0747      2          THEN
479      0748      2              BEGIN
480      0749      2                  TBK$MODULE_DST = .BEST_MODULE;
481      0750      2                  +
482      0751      2                  ! Try to translate this PC into its
483      0752      2                  ! corresponding %line number.
484      0753      2                  -
485      0754      2                  IF NOT TBK$PC_TO_LINE (.VALUE, .BEST_DST, .TBK$GL_EXC_TYPE, TBK$GL_LINE, TBK$GL_STMT)
486      0755      2                  THEN

```

TBKSVM
V04-000

B 6

16-Sep-1984 02:18:08

14-Sep-1984 13:20:26

VAX-11 Bliss-32 V4.0-742

DISK\$VMSMASTER:[TRACE.SRC]TBKSVM.B32;1

Page 11

(3)

```

: 487      0756      3      TBK$GL_STMT = tbk$gl_line = 0;      ! line number later on.
: 488      0757
: 489      0758      TBK$MODULE_CS = BEST_MODULE[dst$b_name];
: 490      0759
: 491      0760      TBK$ROUTINE_CS = BEST_DST[dst$b_name];
: 492      0761      TBK$REL_PC = .VALUE -- .BEST_VALUE;
: 493      0762      END;
: 494      0763
: 495      0764      RETURN;      ! The values are returned via globals.
: 496      0765      1 END;
: INFO#250      L1:0711
: Referenced LOCAL symbol NT_COUNT is probably not initialized
: INFO#250      L1:0548
: Referenced LOCAL symbol IN_MODULE is probably not initialized
: INFO#250      L1:0603
: Referenced LOCAL symbol IN_MODULE is probably not initialized
: INFO#250      L1:0679
: Referenced LOCAL symbol IN_MODULE is probably not initialized
```

```

                                OFFC 00000
5E      FE64      CE 9E 00002
                        59 7C 00007
                        57 D4 00009
                        00000000G 00 D4 0000B
                        00000000G 00 D4 00011
                        00000000G 00 D4 00017
                        00000000G 00 D4 0001D
56      04      AC D0 00023
00000000G 00 56 D0 00027
                        00000000G 00 D4 0002E
                        7E 7C 00034
00000000G 00 01 FB 00036
                        01 50 E8 0003D
                        04 00040
                        52 D4 00041 1$:
10 AE42 D4 00043
OC      AE 9F 00047 2$:
00000000G 00 01 FB 0004A
                        50 D5 00051
                        03 12 00053
                        012D 31 00055
```

```

.TITLE TBKSVM
.IDENT \V04-000\

.EXTRN TBK$FAO_OUT, TBK$PC_TO_LINE
.EXTRN TBK$GET_DST_REC
.EXTRN TBK$GET_NXT_DST
.EXTRN TBK$POSITON_DST
.EXTRN TBK$GL_EXC_TYPE
.EXTRN TBK$MODULE_CS, TBK$ROUTINE_CS
.EXTRN TBK$GL_STMT, TBK$GL_LINE
.EXTRN TBK$REL_PC, TBK$MODULE_DST

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

.ENTRY TBK$SYMBOLIZE, Save R2, R3, R4, R5, R6, R7, R8, - 0414
      R9, R10, R11
      MOVAB -412(SP), SP
      CLRQ BEST_VALUE 0457
      CLRL CURRENT RTN 0459
      CLRL TBK$MODULE_CS 0460
      CLRL TBK$ROUTINE_CS 0461
      CLRL TBK$GL_LINE 0462
      CLRL TBK$GL_STMT
      MOVL VALUE, R6 0463
      MOVL R6, TBK$REL_PC
      CLRL TBK$MODULE_DST 0464
      CLRQ -(SP) 0468
      CALLS #1, TBK$POSITON_DST
      BLBS R0, 1$
      RET
      CLRL RTN_NESTING 0472
      CLRL RTN_NEST_VECTOR[RTN_NESTING] 0473
      PUSHAB DST_REC_ID 0475
      CALLS #1, TBK$GET_NXT_DST
      TSTL DST_REC_ID
      BNEQ 3$
      BRW 17$
```

[illegible]

		67	18	000DC	BGEQ	13\$			
	10	AE42	57	D0	000DE	MOVL	CURRENT_RTN, RTN_NEST_VECTOR[RTN_NESTING]	0533	
			52	D6	000E3	INCL	RTN_NESTING	0534	
		56	55	D1	000E5	CMPL	RBEGIN, R6	0540	
			76	1A	000E8	BGTRU	15\$		
			59	D5	000EA	TSTL	BEST_VALUE	0543	
			0D	13	000EC	BEQL	9\$		
54		56	59	C3	000EE	SUBL3	BEST_VALUE, R6, R4	0544	
51		56	55	C3	000F2	SUBL3	RBEGIN, R6, R1	0545	
		51	54	D1	000F6	CMPL	R4, R1		
			4A	19	000F9	BLSS	13\$		
		5A	58	D0	000FB	9\$: MOVL	IN MODULE, BEST_MODULE	0548	
		59	55	D0	000FE	MOVL	RBEGIN, BEST_VALUE	0549	
		5B	57	D0	00101	MOVL	CURRENT_RTN, BEST_DST	0550	
		6E	53	D0	00104	MOVL	R3, BEST_DST_TYPE	0551	
			79	11	00107	BRB	16\$	0501	
		57	OC	AE42	D0	00109	10\$: MOVL	RTN_NEST_VECTOR-4[RTN_NESTING], CURRENT_RTN	0586
		55	03	A7	D0	0010E	MOVL	3(CURRENT_RTN), RBEGIN	0587
51		55	03	A0	C1	00112	ADDL3	3(DST_REC'D), RBEGIN, R1	0599
		51		56	D1	00117	CMPL	R6, RT	
				10	1A	0011A	BGTRU	11\$	
		56		55	D1	0011C	CMPL	RBEGIN, R6	0600
				0B	1A	0011F	BGTRU	11\$	
		5A		58	D0	00121	MOVL	IN MODULE, BEST_MODULE	0603
		59		55	D0	00124	MOVL	RBEGIN, BEST_VALUE	0604
		5B		57	D0	00127	MOVL	CURRENT_RTN, BEST_DST	0605
				59	11	0012A	BRB	17\$	0602
				52	D5	0012C	11\$: TSTL	RTN_NESTING	0620
				11	15	0012E	BLEQ	12\$	
				52	D7	00130	DECL	RTN_NESTING	0623
			10	AE42	D4	00132	CLRL	RTN_NEST_VECTOR[RTN_NESTING]	0624
		57	10	AE42	D0	00136	MOVL	RTN_NEST_VECTOR[RTN_NESTING], CURRENT_RTN	0626
				59	7C	0013B	CLRQ	BEST_VALUE	0628
				5B	D4	0013D	CLRL	BEST_DST	0629
				6E	D4	0013F	CLRL	BEST_DST_TYPE	0630
	08	AE		53	D0	00141	12\$: MOVL	R3, RTN_RCD	0637
				3B	11	00145	13\$: BRB	16\$	0501
		53		07	A0	9E	14\$: MOVAB	7(DST_REC'D), R3	0651
		51			63	9A	MOVZBL	(R3), R1	
		55		03	A0	D0	MOVL	3(DST_REC'D), PBEGIN	0655
				01	A341	9F	PUSHAB	1(R3)[R1]	0656
51		03			9E	C1	ADDL3	a(SP)+, 3(DST_REC'D), R1	
		A0		51	D6	0015B	INCL	PEND	
		56		55	D1	0015D	CMPL	PBEGIN, R6	0662
				20	1A	00160	15\$: BGTRU	16\$	
		56		51	D1	00162	CMPL	PEND, R6	0664
				1B	1F	00165	BLSSU	16\$	
				55	D5	00167	TSTL	PBEGIN	0666
				17	13	00169	BEQL	16\$	
		59		55	D1	0016B	CMPL	PBEGIN, BEST_VALUE	0672
				12	1F	0016E	BLSSU	16\$	
000000BE		8F		6E	D1	00170	CMPL	BEST_DST_TYPE, #190	0673
				09	13	00177	BEQL	16\$	
		5A		58	D0	00179	MOVL	IN MODULE, BEST_MODULE	0679
		59		55	D0	0017C	MOVL	PBEGIN, BEST_VALUE	0680
		5B		50	D0	0017F	MOVL	DST_REC'D, BEST_DST	0681
				FEC2	31	00182	16\$: BRW	2\$	0501

TBKSVM
V04-000

E 6
16-Sep-1984 02:18:08
14-Sep-1984 13:20:26

VAX-11 BLISS-32 V4.0-742
DISK\$VMSMASTER:[TRACE.SRC]TBKSVM.B32;1
Page 14
(3)

00000000G	00		5A	D5	00185	17\$:	TSTL	BEST_MODULE	:	0746
			4B	13	00187		BEQL	19\$	:	
		00000000G	5A	D0	00189		MOVL	BEST_MODULE, TBK\$MODULE_DST	:	0749
		00000000G	00	9F	00190		PUSHAB	TBK\$GL_STMT	:	0754
		00000000G	00	9F	00196		PUSHAB	TBK\$GL_LINE	:	
		0840	00	DD	0019C		PUSHL	TBK\$GL_EXC_TYPE	:	
			8F	BB	001A2		PUSHR	#^M<R6,R11\$	:	
00000000G	00		05	FB	001A6		CALLS	#5, TBK\$PC_TO_LINE	:	
	0C		50	E8	001AD		BLBS	R0, 18\$	:	
		00000000G	00	D4	001B0		CLRL	TBK\$GL_LINE	:	0756
		00000000G	00	D4	001B6		CLRL	TBK\$GL_STMT	:	
00000000G	00	07	AA	9E	001BC	18\$:	MOVAB	7(R10), TBK\$MODULE_CS	:	0758
00000000G	00	07	AB	9E	001C4		MOVAB	7(R11), TBK\$ROUTINE_CS	:	0760
00000000G	00		59	C3	001CC		SUBL3	BEST_VALUE, R6, TBK\$REL_PC	:	0761
	56		04	001D4	19\$:		RET		:	0765

; Routine Size: 469 bytes, Routine Base: TBK\$CODE + 0000

```
: 498      L 0766 1 %IF TBK_ANY
: 499      U 0767 1 %THEN
: 500      U 0768 1      ! This routine is only used by DEBUGging output routines.
: 501      U 0769 1
: 502      U 0770 1 ROUTINE PR_CS( ADDR ) : NOVALUE =
: 503      U 0771 1
: 504      U 0772 1
: 505      U 0773 1      !++
: 506      U 0774 1      | Functional Description:
: 507      U 0775 1      | Print out a counted string in an
: 508      U 0776 1      | unambiguous way for debugging purposes.
: 509      U 0777 1      | --
: 510      U 0778 1
: 511      U 0779 1 BEGIN
: 512      U 0780 1     MAP
: 513      U 0781 1         ADDR : REF VECTOR[.BYTE];
: 514      U 0782 1
: 515      U 0783 1     ! Don't get fooled!
: 516      U 0784 1
: 517      U 0785 1     IF( .ADDR EQL 0 )
: 518      U 0786 1     THEN
: 519      U 0787 1         $FAO_TT_OUT( '**** PR_CS AT 0 **** ' )
: 520      U 0788 1     ELSE
: 521      U 0789 1         $FAO_TT_OUT( 'Name(!UB.): "!AC". ' , .ADDR[0], ADDR[0] );
: 522      U 0790 1     END;
: 523      U 0791 1
: 524      U 0792 1 %FI
```

TBKSVM
V04-000

6 6
16-Sep-1984 02:18:08
14-Sep-1984 13:20:26

VAX-11 BLISS-32 V4.0-742
DISK\$VMSMASTER:[TRACE.SRC]TBKSVM.B32;1 Page 16 (5)

: 526 0793 1 END
: 527 0794 0 ELUDOM

!End of module

PSECT SUMMARY

Name Bytes Attributes
TBK\$CODE 469 NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(0)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
-\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	4	0	1000	00:02.0
-\$255\$DUA28:[TRACE.OBJ]TBKLIB.L32;1	157	1	0	14	00:00.3
-\$255\$DUA28:[TRACE.OBJ]STRUCDEF.L32;1	32	0	0	7	00:00.1
-\$255\$DUA28:[TRACE.OBJ]TBKDST.L32;1	414	118	28	30	00:00.3

: Information: 4
: Warnings: 0
: Errors: 0

COMMAND QUALIFIERS

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

: Size: 469 code + 0 data bytes
: Run Time: 00:17.4
: Elapsed Time: 00:54.9
: Lines/CPU Min: 2745
: Lexemes/CPU-Min: 20987
: Memory Used: 188 pages
: Compilation Complete

0402 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

