


```

BBB888888      AAAA      SSSSSSSS      IIIII      NN      NN      IIIII      DDDDDDD      FFFFFFFFFF      SSSSSSSS
BBB888888      AAAA      SSSSSSSS      IIIII      NN      NN      IIIII      DDDDDDD      FFFFFFFFFF      SSSSSSSS
BB      BB      AA      AA      SS      II      NN      NN      II      DD      DD      FF      SS
BB      BB      AA      AA      SS      II      NN      NN      II      DD      DD      FF      SS
BB      BB      AA      AA      SS      II      NNNN      NN      II      DD      DD      FF      SS
BB      BB      AA      AA      SS      II      NNNN      NN      II      DD      DD      FF      SS
BBB888888      AA      AA      SSSSSS      II      NN      NN      II      DD      DD      FFFFFFFF      SSSSSS
BBB888888      AA      AA      SSSSSS      II      NN      NN      II      DD      DD      FFFFFFFF      SSSSSS
BB      BB      AAAAAAAAAA      SS      II      NN      NNNN      II      DD      DD      FF      SS
BB      BB      AAAAAAAAAA      SS      II      NN      NNNN      II      DD      DD      FF      SS
BB      BB      AA      AA      SS      II      NN      NN      II      DD      DD      FF      SS
BB      BB      AA      AA      SS      II      NN      NN      II      DD      DD      FF      SS
BBB888888      AA      AA      SSSSSSSS      IIIII      NN      NN      IIIII      DDDDDDD      FF      SSSSSSSS
BBB888888      AA      AA      SSSSSSSS      IIIII      NN      NN      IIIII      DDDDDDD      FF      SSSSSSSS

```

....
....
....
....

```

LL      IIIII      SSSSSSSS
LL      IIIII      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      IIIII      SSSSSSSS
LLLLLLLLLL      IIIII      SSSSSSSS

```



```

1 0001 0 MODULE BASSINIT_DFS (
2 0002 0
3 0003 0 IDENT = '1-005'
4 0004 1 ) =
5 0005 1 BEGIN
6 0006 1
7 0007 1 *****
8 0008 1 *
9 0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
10 0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1
30 0030 1
31 0031 1 ++
32 0032 1 FACILITY: BASIC-PLUS-2 Frame Support
33 0033 1
34 0034 1 ABSTRACT:
35 0035 1
36 0036 1 These routines set up and tear down frames for BASIC-PLUS-2.
37 0037 1 Frames are used for main routines, external functions,
38 0038 1 external subroutines, internal functions (both DEFs and DEF*s)
39 0039 1 internal subroutines (GOSUBs) and condition handlers.
40 0040 1
41 0041 1 ENVIRONMENT: VAX-11 user mode
42 0042 1
43 0043 1 AUTHOR: John Sauter, CREATION DATE: 10-Oct-78
44 0044 1
45 0045 1 MODIFIED BY:
46 0046 1
47 0047 1 : VERSION
48 0048 1 1-001 - Original. Just a skeleton.
49 0049 1 1-002 - Change LIB$$ and OTS$$ to STR$. This routine is still not
50 0050 1 implemented. JBS 21-MAY-1979
51 0051 1 1-003 - Finally, code this routine, based on BASSINIT_DEF.
52 0052 1 JBS 03-AUG-1979
53 0053 1 1-004 - Remove BASSK WROMATPAC, not used. JBS 19-SEP-1979
54 0054 1 1-005 - Fix a comment. JBS 07-NOV-1979
55 0055 1 --
56 0056 1
57 0057 1 !

```

BASSINIT_DFS
1-005

¹
16-Sep-1984 00:36:03
²
14-Sep-1984 11:55:07

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASINIDFS.B32;1

Page (1) 2

: 58

0058 1 !<BLF/PAGE>


```
60 0059 1 |
61 0060 1 | SWITCHES:
62 0061 1 |
63 0062 1 |
64 0063 1 | SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
65 0064 1 |
66 0065 1 |
67 0066 1 | LINKAGES:
68 0067 1 |
69 0068 1 |
70 0069 1 | LINKAGE
71 0070 1 |     BASSINIT_LINK = JSB (REGISTER = 0, REGISTER = 1, REGISTER = 2) : !
72 0071 1 |     GLOBAL (BSF$A_MAJOR_STG = 11, BSF$A_MINOR_STG = 10, BSF$A_TEMP_STG = 9) !
73 0072 1 |     NOPRESERVE (8, 7, 6, 5, 4, 3, 2, 1, 0);
74 0073 1 |
75 0074 1 |
76 0075 1 | TABLE OF CONTENTS:
77 0076 1 |
78 0077 1 |
79 0078 1 | FORWARD ROUTINE
80 0079 1 |     BASSINIT_DFS_R8 : NOVALUE BASSINIT_LINK; ! start DEF*
81 0080 1 |
82 0081 1 |
83 0082 1 | INCLUDE FILES:
84 0083 1 |
85 0084 1 |
86 0085 1 | REQUIRE 'RTLIN:RTLPSECT'; ! macros for defining psects
87 0180 1 |
88 0181 1 | REQUIRE 'RTLIN:BASFRAME'; ! Define frame structure
89 0384 1 |
90 0385 1 | REQUIRE 'RTLIN:BASINARG'; ! Define argument list
91 0469 1 |
92 0470 1 | LIBRARY 'RTLSTARLE'; ! System symbols
93 0471 1 |
94 0472 1 |
95 0473 1 | MACROS:
96 0474 1 |
97 0475 1 |     NONE
98 0476 1 |
99 0477 1 | EQUATED SYMBOLS:
100 0478 1 |
101 0479 1 |     NONE
102 0480 1 |
103 0481 1 | PSECTS:
104 0482 1 |
105 0483 1 | DECLARE_PSECTS (BAS); ! declare psects for BASS facility
106 0484 1 |
107 0485 1 | OWN STORAGE:
108 0486 1 |
109 0487 1 |     NONE
110 0488 1 |
111 0489 1 | EXTERNAL REFERENCES:
112 0490 1 |
113 0491 1 |
114 0492 1 | EXTERNAL ROUTINE
115 0493 1 |     BASS$SIGNAL : NOVALUE, ! signals error
116 0494 1 |     STR$FREE1_DX, ! deallocates one string
```



```

: 117      0495 1      BASSHANDLER;
: 118      0496 1
: 119      0497 1      !+
: 120      0498 1      ! The following are the error codes used in this module.
: 121      0499 1      !-
: 122      0500 1
: 123      0501 1      EXTERNAL LITERAL
: 124      0502 1      BASSK_TOOFEWARG : UNSIGNED (8),
: 125      0503 1      BASSK_TOOMANARG : UNSIGNED (8),
: 126      0504 1      BASSK_SCAFACINT : UNSIGNED (8),
: 127      0505 1      BASSK_PROLOSSOR : UNSIGNED (8),
: 128      0506 1      BASSK_ARGDONMAT : UNSIGNED (8),
: 129      0507 1      BASSK_NOTIMP : UNSIGNED (8);
: 130      0508 1

```

! handles signals

! Too few arguments
! Too many arguments
! Scale factor interlock
! Program lost, sorry
! Arguments don't match
! Not implemented

```
132 0509 1 GLOBAL ROUTINE BASSINIT_DFS_R8 (
133 0510 1     ARGLIST,
134 0511 1     DATA_RELOC
135 0512 1 ) : NOVALUE BASSINIT_LINK =
136 0513 1
137 0514 1 ++
138 0515 1 FUNCTIONAL DESCRIPTION:
139 0516 1
140 0517 1     Set up a frame for a BASIC-PLUS-2 DEF*. The frame is allocated
141 0518 1     on the stack, and R10 and R9 are set up to point to it.
142 0519 1     The argument tells how to do the allocation.
143 0520 1
144 0521 1 FORMAL PARAMETERS:
145 0522 1
146 0523 1     ARGLIST.rl.v List of information needed to set up the
147 0524 1     frame. See BASIC-PLUS-2/VAX Description
148 0525 1     of Generated Code for details.
149 0526 1     DATA_RELOC.ra.v Address of the major procedure's contribution
150 0527 1     to the data PSECT. This is needed so that the
151 0528 1     argument list can be PIC.
152 0529 1
153 0530 1 IMPLICIT INPUTS:
154 0531 1
155 0532 1     NONE
156 0533 1
157 0534 1 IMPLICIT OUTPUTS:
158 0535 1
159 0536 1     The values of R10 and R9, which point to the automatic
160 0537 1     storage and the temporary storage, respectively.
161 0538 1
162 0539 1 ROUTINE VALUE:
163 0540 1
164 0541 1     NONE
165 0542 1
166 0543 1 COMPLETION CODES:
167 0544 1
168 0545 1     NONE
169 0546 1
170 0547 1 SIDE EFFECTS:
171 0548 1
172 0549 1     Leaves lots of things on the stack for use by the compiled
173 0550 1     BASIC-PLUS-2 code. These things will be removed by
174 0551 1     BASSEND_DFS_R8.
175 0552 1
176 0553 1 --
177 0554 1
178 0555 2 BEGIN
179 0556 2
180 0557 2 EXTERNAL REGISTER
181 0558 2     BSF$A_MAJOR_STG : REF BLOCK [0, BYTE] FIELD (BSF$MAJOR_FRAME),
182 0559 2     BSF$A_MINOR_STG : REF BLOCK [0, BYTE] FIELD (BSF$MINOR_FRAME),
183 0560 2     BSF$A_TEMP_STG;
184 0561 2
185 0562 2 BUILTIN
186 0563 2     AP,
187 0564 2     FP,
188 0565 2     SP;
```



```
189 0566 2
190 0567 2 MAP
191 0568 2     ARGLIST : REF BLOCK [0, BYTE] FIELD (BASSINIT_ARGS), ! arg list
192 0569 2     AP : REF VECTOR; ! caller's arg list
193 0570 2
194 0571 2 +
195 0572 2 | Define local variables as registers. We cannot have any stack locals
196 0573 2 | since we manipulate the stack pointer in this routine.
197 0574 2 |
198 0575 2 -
199 0576 2 REGISTER
200 0577 2     RETURN_ADDRESS, ! address to return to
201 0578 2     FMP : REF BLOCK [0, BYTE] FIELD (BSF$FCD), ! pointer to FCD
202 0579 2     ARRAY_DESC : REF BLOCK [0, BYTE], ! pointer to build array descriptors
203 0580 2     ARRAY_INDEX; ! index for array modification
204 0581 2
205 0582 2 +
206 0583 2 | Save return address because we are going to fool with the stack
207 0584 2 |
208 0585 2 | RETURN_ADDRESS = ..SP;
209 0586 2 +
210 0587 2 | Make sure we are passed an argument list we understand.
211 0588 2 |
212 0589 2 -
213 0590 2     IF (.ARGLIST [BASS$B_IN_V_FCD] NEQ BASS$K_IN_V_FCD) THEN BASS$$SIGNAL (BASS$K_NOTIMP);
214 0591 2
215 0592 2 +
216 0593 2 | Allocate frame control data.
217 0594 2 |
218 0595 2 | FMP = .FMP;
219 0596 2 | SP = .FMP - BSF$K_LENFCDDFS + %UPVAL;
220 0597 2 +
221 0598 2 | LOAD Rn (R10)
222 0599 2 |
223 0600 2 | BSF$A_MINOR_STG = .SP - 127;
224 0601 2 +
225 0602 2 | Initialize parts of the frame control data.
226 0603 2 |
227 0604 2 | FMP [BSF$A_MARK] = 0;
228 0605 2 | FMP [BSF$A_BASE_R11] = .BSF$A_MAJOR_STG;
229 0606 2 | FMP [BSF$A_BASE_R10] = .BSF$A_MINOR_STG;
230 0607 2 | FMP [BSF$B_LEN_FCD] = BSF$K_LENFCDDFS;
231 0608 2 | FMP [BSF$B_PROC_CODE] = .ARGLIST [BASS$B_IN_PROC_C];
232 0609 2 | FMP [BSF$W_FCD_FLAGS] = .ARGLIST [BASS$W_IN_FLAGS];
233 0610 2 | FMP [BSF$A_PROC_ID] = .ARGLIST [BASS$L_IN_PROC_I] + .DATA_RELOC;
234 0611 2 | FMP [BSF$A_INIT_ARG] = .ARGLIST;
235 0612 2 | FMP [BSF$L_INIT_REL] = .DATA_RELOC;
236 0613 2 +
237 0614 2 | Allocate numeric scalars. They are all initialized to zero.
238 0615 2 |
239 0616 2 -
240 0617 2     INCR COUNTER FROM 1 TO .ARGLIST [BASS$L_IN_LEN_SC] DO
241 0618 2     BEGIN
242 0619 2     SP = .SP - %UPVAL;
243 0620 2     .SP = 0;
244 0621 2     END;
245 0622 2
```



```
246 0623 2 !+
247 0624 2 !- Copy formals.
248 0625 2 !-
249 0626 2 !-
250 0627 2 !-   DECR COUNTER FROM MIN (.ARGLIST [BAS$B_IN_NO_FML], ((.AP [0]) AND 255)) TO 1 DO
251 0628 2 !-   BEGIN
252 0629 2 !-   SP = .SP - %UPVAL;
253 0630 2 !-   .SP = .AP [.COUNTER];
254 0631 2 !-   END;
255 0632 2 !-
256 0633 2 !+
257 0634 2 !- Allocate and initialize descriptors.
258 0635 2 !-
259 0636 2 !-   SP = .SP - .ARGLIST [BAS$L_IN_LEN_DT];
260 0637 2 !-
261 0638 2 !+ Set ARRAY_DESC to point to the space allocated.
262 0639 2 !-
263 0640 2 !-   ARRAY_DESC = .SP;
264 0641 2 !+
265 0642 2 !- Load the space from the template and then modify it based
266 0643 2 !- on the modification table.
267 0644 2 !-
268 0645 2 !-
269 0646 2 !-   INCR COUNTER FROM 0 TO ((.ARGLIST [BAS$L_IN_LEN_DT]^2) - 1) DO
270 0647 2 !-   BEGIN
271 0648 2 !-   ARRAY_DESC [.COUNTER*%UPVAL, 0, %BPVAL, 0] = .((.ARGLIST [BAS$L_IN_DT_TMT]) + !
272 0649 2 !-   .DATA_RELOC + (.COUNTER*%UPVAL));
273 0650 2 !-   END;
274 0651 2 !-
275 0652 2 !+
276 0653 2 !- Now modify the descriptors. These are usually array descriptors.
277 0654 2 !-
278 0655 2 !-
279 0656 2 !-   INCR COUNTER FROM 0 TO (.ARGLIST [BAS$L_IN_LEN_DM] - 1) DO
280 0657 2 !-   BEGIN
281 0658 2 !-   ARRAY_INDEX = .((.ARGLIST [BAS$L_IN_DT_MOD]) + .DATA_RELOC + (.COUNTER*%UPVAL));
282 0659 2 !-   BSF$A_MINOR STG [.ARRAY_INDEX, 0, %BPVAL, 0] !
283 0660 2 !-   = .BSF$A_MINOR_STG [.ARRAY_INDEX, 0, %BPVAL, 0] + .BSF$A_MINOR_STG;
284 0661 2 !-   END;
285 0662 2 !-
286 0663 2 !+
287 0664 2 !- Allocate dynamic string descriptors.
288 0665 2 !-
289 0666 2 !-
290 0667 2 !-   INCR COUNTER FROM 1 TO .ARGLIST [BAS$W_IN_NO_DST] DO
291 0668 2 !-   BEGIN
292 0669 2 !-   SP = .SP - %UPVAL;
293 0670 2 !-   .SP = 0; ! Pointer 0 implies not allocated.
294 0671 2 !-   SP = .SP - %UPVAL;
295 0672 2 !-   BLOCK [.SP, DSC$B_CLASS; 0, BYTE] = DSC$K_CLASS_D; ! dynamic
296 0673 2 !-   BLOCK [.SP, DSC$B_DTYPE; 0, BYTE] = DSC$K_DTYPE_T; ! text
297 0674 2 !-   BLOCK [.SP, DSC$W_LENGTH; 0, BYTE] = 0; ! length = 0
298 0675 2 !-   END;
299 0676 2 !-
300 0677 2 !-   FMP [BSF$A_STR_DESC] = .SP;
301 0678 2 !+
302 0679 2 !- Allocate fixed string templates.
```



```

303 0680 2 :-
304 0681 2
305 0682 2 INCR COUNTER FROM 1 TO .ARGLIST [BASSW_IN_NO_FST] DO
306 0683 2 BEGIN
307 0684 2 SP = .SP - %UPVAL;
308 0685 2 .SP = 0; ! Pointer 0 implies not allocated.
309 0686 2 SP = .SP - %UPVAL;
310 0687 2 BLOCK [.SP, DSC$B_CLASS; 0, BYTE] = DSC$K_CLASS_S; ! fixed
311 0688 2 BLOCK [.SP, DSC$B_DTYPE; 0, BYTE] = DSC$K_DTYPE_T; ! text
312 0689 2 BLOCK [.SP, DSC$W_LENGTH; 0, BYTE] = 0; ! length = 0
313 0690 2 END;
314 0691 2
315 0692 2 !+
316 0693 2 Allocate numeric array elements. They are all initialized to zero.
317 0694 2 :-
318 0695 2
319 0696 2 INCR COUNTER FROM 1 TO (.ARGLIST [BASSL_IN_LEN_NA]^2) DO
320 0697 2 BEGIN
321 0698 2 SP = .SP - %UPVAL;
322 0699 2 .SP = 0;
323 0700 2 END;
324 0701 2
325 0702 2 !+
326 0703 2 Allocate temporary cells.
327 0704 2 :-
328 0705 2
329 0706 2 IF ((.ARGLIST [BASSL_IN_NO_TST] NEQ 0) OR (.ARGLIST [BASSL_IN_NO_NMT] NEQ 0))
330 0707 2 THEN
331 0708 2 BEGIN
332 0709 2 !+
333 0710 2 We must set up R9. First allocate string temporaries.
334 0711 2 :-
335 0712 2
336 0713 2 INCR COUNTER FROM 1 TO .ARGLIST [BASSL_IN_NO_TST] DO
337 0714 2 BEGIN
338 0715 2 SP = .SP - %UPVAL;
339 0716 2 .SP = 0; ! Pointer 0 implies not allocated.
340 0717 2 SP = .SP - %UPVAL;
341 0718 2 BLOCK [.SP, DSC$B_CLASS; 0, BYTE] = DSC$K_CLASS_D; ! dynamic
342 0719 2 BLOCK [.SP, DSC$B_DTYPE; 0, BYTE] = DSC$K_DTYPE_T; ! text
343 0720 2 BLOCK [.SP, DSC$W_LENGTH; 0, BYTE] = 0; ! length = 0
344 0721 2 END;
345 0722 2
346 0723 2 !+
347 0724 2 Point R9 to the last string descriptor allocated.
348 0725 2 :-
349 0726 2 BSF$A_TEMP_STG = .SP;
350 0727 2 !+
351 0728 2 Now allocate numeric temporaries.
352 0729 2 :-
353 0730 2 SP = .SP - .ARGLIST [BASSL_IN_NO_NMT];
354 0731 2 END;
355 0732 2
356 0733 2 !+
357 0734 2 Store R9 in the stack frame for setting up I/O lists.
358 0735 2 :-
359 0736 2 FMP [BSF$A_BASE_R9] = .BSF$A_TEMP_STG;
```



```
360 0737 2 !+
361 0738 2 ! Complete frame.
362 0739 2 !-
363 0740 2 FMP [BSF$A_BASE_SP] = .SP;
364 0741 2 FMP [BSF$A_HANDLER] = BAS$HANDLER;
365 0742 2 !+
366 0743 2 ! First consistency checks.
367 0744 2 !-
368 0745 2
369 0746 2 IF (((.AP [0]) AND 255) NEQ .ARGLIST [BAS$B_IN_NO_FML])
370 0747 2 THEN
371 0748 2 !+
372 0749 2 ! The number of arguments is incorrect.
373 0750 2 !-
374 0751 2 BEGIN
375 0752 2
376 0753 2 IF (((.AP [0]) AND 255) GTRU .ARGLIST [BAS$B_IN_NO_FML])
377 0754 2 THEN
378 0755 2 BAS$$SIGNAL (BAS$K_TOOMANARG)
379 0756 2 ELSE
380 0757 2 BAS$$SIGNAL (BAS$K_TOOFEWARG);
381 0758 2
382 0759 2 END;
383 0760 2
384 0761 2 IF (((.FMP [BSF$W_FCD_FLAGS]) AND (BSF$M_FCD_RSTR)) NEQ 0)
385 0762 2 THEN
386 0763 2 BEGIN
387 0764 2
388 0765 2 LOCAL
389 0766 2 STR_DESC_ADDR : REF BLOCK [8, BYTE];
390 0767 2
391 0768 2 !+
392 0769 2 ! This procedure has been marked by the compiler as returning a
393 0770 2 ! string result. Be sure that there is at least one formal, and
394 0771 2 ! that it is a dynamic string descriptor. If so, null its value.
395 0772 2 !-
396 0773 2
397 0774 2 IF (.ARGLIST [BAS$B_IN_NO_FML] LSSU 1) THEN BAS$$SIGNAL (BAS$K_TOOFEWARG);
398 0775 2
399 0776 2 STR_DESC_ADDR = AP [1];
400 0777 2 STR_DESC_ADDR = ..STR_DESC_ADDR;
401 0778 2
402 0779 2 IF ((.STR_DESC_ADDR [DSC$B_CLASS] NEQU DSC$K_CLASS_D) OR !
403 0780 2 (.STR_DESC_ADDR [DSC$B_DTYPE] NEQU DSC$K_DTYPE_T))
404 0781 2 THEN
405 0782 2 BAS$$SIGNAL (BAS$K_ARGDONMAT);
406 0783 2
407 0784 2 !+
408 0785 2 ! Null the string. This insures that, if the procedure does not reference
409 0786 2 ! the string, the function will have the value of the null string.
410 0787 2 !-
411 0788 2 STR$FREE1_DX (.STR_DESC_ADDR);
412 0789 2 END;
413 0790 2
414 0791 2 !+
415 0792 2 ! Put the return address back on the stack so we can return to the
416 0793 2 ! caller.
```


BASSINIT_DFS
1-005

D 3
16-Sep-1984 00:36:03
14-Sep-1984 11:55:07

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASINIDFS.B32;1

Page 10
(3)

```
: 417      0794 2  !-
: 418      0795 2
: 419      0796 2  SP = .SP - %UPVAL;
: 420      0797 2  .SP = .RETURN_ADDRESS;
: 421      0798 1  RETURN;
                        END;
```

! of BASSINIT_DFS_R8

.TITLE BASSINIT_DFS
.IDENT \1-005\

.EXTRN BASS\$SIGNAL, STR\$FREE1_DX
.EXTRN BASS\$HANDLER, BASS\$K_TOOFEWARG
.EXTRN BASS\$K_TOOMANARG
.EXTRN BASS\$K_SCAFACINT
.EXTRN BASS\$K_PROLOSSOR
.EXTRN BASS\$K_ARGDONMAT
.EXTRN BASS\$K_NOTIMP

.PSECT _BASS\$CODE, NOWRT, SHR, PIC, 2

57	51	D0	00000	BASSINIT_DFS_R8::	MOV	R1, R7	0509
54	50	D0	00003		MOV	R0, R4	
52	6E	D0	00006		MOV	(SP), RETURN_ADDRESS	0585
01	04	A4	91 00009		CMP	4(ARGLIST), #1	0590
		0B	13 0000D		BEQ	1\$	
7E	00G	8F	9A 0000F		MOVZBL	#BASS\$K_NOTIMP, -(SP)	
00		01	FB 00013		CALLS	#1, BASS\$SIGNAL	
53		5D	D0 0001A	1\$:	MOV	FP, FMP	0595
5E	DB	A3	9E 0001D		MOVAB	-4(R3), SP	0596
5A	81	AE	9E 00021		MOVAB	-127(SP), BSF\$A_MINOR_STG	0600
	FC	A3	D4 00025		CLRL	-4(FMP)	0604
F0	A3	5A	7D 00028		MOVQ	BSF\$A_MINOR_STG, -16(FMP)	0606
E4	A3	2C	90 0002C		MOV	#44, -28(FMP)	0607
E5	A3	05	A4 90 00030		MOV	5(ARGLIST), -27(FMP)	0608
E6	A3	06	A4 B0 00035		MOV	6(ARGLIST), -26(FMP)	0609
E8	A3	08	B447 9E 0003A		MOVAB	28(ARGLIST)[DATA_RELOC], -24(FMP)	0610
D8	A3	54	D0 00040		MOV	ARGLIST, -40(FMP)	0611
DC	A3	57	D0 00044		MOV	DATA_RELOC, -36(FMP)	0612
		50	D4 00048		CLRL	COUNTER	0617
		05	11 0004A		BRB	3\$	
5E		04	C2 0004C	2\$:	SUBL2	#4, SP	0619
		6E	D4 0004F		CLRL	(SP)	0620
F6		50	A4 F3 00051	3\$:	AOBLEQ	16(ARGLIST), COUNTER, 2\$	0617
		50	A4 9A 00056		MOVZBL	20(ARGLIST), R0	0627
		50	6C 91 0005A		CMP	(AP), R0	
		03	1E 0005D		BGEQ	4\$	
		50	6C 9A 0005F		MOVZBL	(AP), R0	
		50	D6 00062	4\$:	INCL	COUNTER	
		07	11 00064		BRB	6\$	
5E		04	C2 00066	5\$:	SUBL2	#4, SP	0629
6E		6C40	D0 00069		MOV	(AP)[COUNTER], (SP)	0630
F3		50	F5 0006D	6\$:	SOBGR	COUNTER, 5\$	0627
5E	18	A4	C2 00070		SUBL2	24(ARGLIST), SP	0636
50		5E	D0 00074		MOV	SP, ARRAY_DESC	0640
56	18	A4	8F 78 00077		ASHL	#-2, 24(ARGLIST), R6	0646
		51	01 CE 0007D		MNEGL	#1, COUNTER	0648

55	57	1C	0A	11	00080	BRB	8\$		
	6041		A4	C1	00082	ADDL3	28(ARGLIST), DATA RELOC, R5		0649
F2	51		6541	D0	00087	MOVL	(R5)[COUNTER], (ARRAY_DESC)[COUNTER]		0648
	51		56	F2	0008C	AOBLSS	R6, COUNTER, 7\$		0646
			01	CE	00090	MNEGL	#1, COUNTER		0658
			0F	11	00093	BRB	10\$		
55	57	24	A4	C1	00095	ADDL3	36(ARGLIST), DATA RELOC, R5		
	50		6541	D0	0009A	MOVL	(R5)[COUNTER], ARRAY INDEX		
			604A	9F	0009E	PUSHAB	(ARRAY_INDEX)[BSFSA_MINOR_STG]		0660
	9E		5A	C0	000A1	ADDL2	BSFSA_MINOR_STG, @ (SP)+		
EC	51	20	A4	F2	000A4	AOBLSS	32(ARGLIST), COUNTER, 9\$		0656
	51	28	A4	3C	000A9	MOVZWL	40(ARGLIST), R1		0667
			50	D4	000AD	CLRL	COUNTER		
			0F	11	000AF	BRB	12\$		
	5E		04	C2	000B1	SUBL2	#4, SP		0669
			6E	D4	000B4	CLRL	(SP)		0670
	5E		04	C2	000B6	SUBL2	#4, SP		0671
	6E	020E0000	8F	D0	000B9	MOVL	#34471936, (SP)		0674
ED	50		51	F3	000C0	AOBLEQ	R1, COUNTER, 11\$		0667
	A3		5E	D0	000C4	MOVL	SP, -32(FMP)		0677
	51	2A	A4	3C	000C8	MOVZWL	42(ARGLIST), R1		0682
			50	D4	000CC	CLRL	COUNTER		
			0F	11	000CE	BRB	14\$		
	5E		04	C2	000D0	SUBL2	#4, SP		0684
			6E	D4	000D3	CLRL	(SP)		0685
	5E		04	C2	000D5	SUBL2	#4, SP		0686
	6E	010E0000	8F	D0	000D8	MOVL	#17694720, (SP)		0689
ED	50		51	F3	000DF	AOBLEQ	R1, COUNTER, 13\$		0682
50	A4	FE	8F	78	000E3	ASHL	#-2, 44(ARGLIST), R0		0696
			51	D4	000E9	CLRL	COUNTER		
			05	11	000EB	BRB	16\$		
	5E		04	C2	000ED	SUBL2	#4, SP		0698
			6E	D4	000F0	CLRL	(SP)		0699
F7	51		50	F3	000F2	AOBLEQ	R0, COUNTER, 15\$		0696
		30	A4	D5	000F6	TSTL	48(ARGLIST)		0706
			05	12	000F9	BNEQ	17\$		
		34	A4	D5	000FB	TSTL	52(ARGLIST)		
			1F	13	000FE	BEQL	20\$		
			50	D4	00100	CLRL	COUNTER		0713
			0F	11	00102	BRB	19\$		
	5E		04	C2	00104	SUBL2	#4, SP		0715
			6E	D4	00107	CLRL	(SP)		0716
	5E		04	C2	00109	SUBL2	#4, SP		0717
	6E	020E0000	8F	D0	0010C	MOVL	#34471936, (SP)		0720
EC	50	30	A4	F3	00113	AOBLEQ	48(ARGLIST), COUNTER, 18\$		0713
	59		5E	D0	00118	MOVL	SP, BSFSA_TEMP_STG		0726
	5E	34	A4	C2	0011B	SUBL2	52(ARGLIST), SP		0730
	A3		59	D0	0011F	MOVL	BSFSA_TEMP_STG, -20(FMP)		0736
	A3		5E	D0	00123	MOVL	SP, -8(FMP)		0740
	63	00000000G	00	9E	00127	MOVAB	BASSHANDLER, (FMP)		0741
	14		6C	91	0012E	CMPB	(AP), 20(ARGLIST)		0746
			13	13	00132	BEQL	23\$		
			06	1B	00134	BLEQU	21\$		0753
	7E	00G	8F	9A	00136	MOVZBL	#BASSK_TOOMANARG, -(SP)		0755
			04	11	0013A	BRB	22\$		
	7E	00G	8F	9A	0013C	MOVZBL	#BASSK_TOOFEWARG, -(SP)		0757
00000000G	00		01	FB	00140	CALLS	#1, BASS\$SIGNAL		

37	E6	A3		0D	E1	00147	23\$:	BBC	#13, -26(FMP), 27\$	: 0761
			14	A4	95	0014C		TSTB	20(ARGLIST)	: 0774
				0B	12	0014F		BNEQ	24\$	
		7E	00G	8F	9A	00151		MOVZBL	#BASSK_TOOFEWARG, -(SP)	
00000000G		00		01	FB	00155		CALLS	#1, BASS\$SIGNAL	
		53		04	AC	9E 0015C	24\$:	MOVAB	4(AP), STR_DESC_ADDR	: 0776
		53		63	D0	00160		MOVL	(STR_DESC_ADDR), STR_DESC_ADDR	: 0777
		02		03	A3	91 00163		CMPB	3(STR_DESC_ADDR), #2	: 0779
				06	12	00167		BNEQ	25\$	
		0E		02	A3	91 00169		CMPB	2(STR_DESC_ADDR), #14	: 0780
				0B	13	0016D		BEQ	26\$	
00000000G		7E	00G	8F	9A	0016F	25\$:	MOVZBL	#BASSK_ARGDONMAT, -(SP)	: 0782
		00		01	FB	00173		CALLS	#1, BASS\$SIGNAL	
00000000G		00		53	DD	0017A	26\$:	PUSHL	STR_DESC_ADDR	: 0788
		5E		01	FB	0017C		CALLS	#1, STR\$FREE1_DX	
		6E		04	C2	00183	27\$:	SUBL2	#4, SP	: 0795
				52	D0	00186		MOVL	RETURN_ADDRESS, (SP)	: 0796
				05	00189			RSB		: 0798

; Routine Size: 394 bytes, Routine Base: _BASS\$CODE + 0000

```

: 422      0799 1
: 423      0800 1 END
: 424      0801 1
: 425      0802 0 ELUDOM

```

PSECT SUMMARY

Name	Bytes	Attributes
_BASS\$CODE	394	NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	6	0	581	00:01.1

COMMAND QUALIFIERS

```

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

```


BASSINIT_DFS
1-005

G 3
16-Sep-1984 00:36:03
14-Sep-1984 11:55:07

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASINIDFS.B32;1

Page 13
(3)

:)
: Size: 394 code + 0 data bytes
: Run Time: 00:12.4
: Elapsed Time: 00:24.9
: Lines/CPU Min: 3886
: Lexemes/CPU-Min: 20699
: Memory Used: 170 pages
: Compilation Complete

0024 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY