

```

PPPPPPPPPPPP      AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTT      LLL
PPPPPPPPPPPPPP    AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTT      LLL
PPPPPPPPPPPPPP    AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTT      LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPP              PPP  AAA          AAA  SSS              RRR              RRR              TTT              LLL
PPPPPPPPPPPPPP    AAA          AAA          SSSSSSSSSS      RRRRRRRRRRRR      TTT              LLL
PPPPPPPPPPPPPP    AAA          AAA          SSSSSSSSSS      RRRRRRRRRRRR      TTT              LLL
PPPPPPPPPPPPPP    AAA          AAA          SSSSSSSSSS      RRRRRRRRRRRR      TTT              LLL
PPP              AAAAAAAAAAAAAAAAAA          SSS              RRR              RRR              TTT              LLL
PPP              AAAAAAAAAAAAAAAAAA          SSS              RRR              RRR              TTT              LLL
PPP              AAAAAAAAAAAAAAAAAA          SSS              RRR              RRR              TTT              LLL
PPP              AAA          AAA          SSS              RRR              RRR              TTT              LLL
PPP              AAA          AAA          SSS              RRR              RRR              TTT              LLL
PPP              AAA          AAA          SSS              RRR              RRR              TTT              LLL
PPP              AAA          AAA          SSS              RRR              RRR              TTT              LLL
PPP              AAA          AAA          SSSSSSSSSSSS      RRR              RRR              TTT              LLLLLLLLLLLLLLLLLL
PPP              AAA          AAA          SSSSSSSSSSSS      RRR              RRR              TTT              LLLLLLLLLLLLLLLLLL
PPP              AAA          AAA          SSSSSSSSSSSS      RRR              RRR              TTT              LLLLLLLLLLLLLLLLLL

```

2

Sym

PAS

PAS
PASPAS
PAS

PAS

PAS
PAS

PAS
PAS

PAS

PAS
PASPAS
PAS

PAS

PAS
PAS

PAS

PAS
PASPAS
PAS

PAS

PAS
PASPAS
PAS

PAS

PAS
PAS

PAS
PAS

PAS

PAS

PAS

PAS

PAC

PAS

PAS

PAS
PAS

PAS

PAS

PAS

PAS

PAS
PASPAS
PAS

PAS

PAS
PAS

PAS

```
PPPPPPPP      AAAAAA      SSSSSSSS  WW      WW  RRRRRRRR  IIIIII  RRRRRRRR  FFFFFFFF  DDDDDDDD
PPPPPPPP      AAAAAA      SSSSSSSS  WW      WW  RRRRRRRR  IIIIII  RRRRRRRR  FFFFFFFF  DDDDDDDD
PP      PP  AA      AA  SS      WW      WW  RR      RR  II      RR      FF      DD      DD
PP      PP  AA      AA  SS      WW      WW  RR      RR  II      RR      FF      DD      DD
PP      PP  AA      AA  SS      WW      WW  RR      RR  II      RR      FF      DD      DD
PP      PP  AA      AA  SS      WW      WW  RR      RR  II      RR      FF      DD      DD
PPPPPPPP      AA      AA  SSSSSS  WW      WW  RRRRRRRR  II      RRRRRRRR  FFFFFFFF  DD      DD
PPPPPPPP      AA      AA  SSSSSS  WW      WW  RRRRRRRR  II      RRRRRRRR  FFFFFFFF  DD      DD
PP      AA      AA  SS      WW      WW  RR      RR  II      RR      FF      DD      DD
PP      AA      AA  SS      WWW  WWW  RR      RR  II      RR      FF      DD      DD
PP      AA      AA  SS      WWW  WWW  RR      RR  II      RR      FF      DD      DD
PP      AA      AA  SSSSSSSS  WW      WW  RR      RR  IIIIII  RR      RR      FF      DDDDDDDD
PP      AA      AA  SSSSSSSS  WW      WW  RR      RR  IIIIII  RR      RR      FF      DDDDDDDD
```

```
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 PASSWRITE_REALF D.( %TITLE 'Write a D_floating in F format'
2 0002 0 IDENT = '1-003' ! File: PASWRIRFD.B32 Edit: DG1003
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
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 FACILITY: Pascal Language Support
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1
35 0035 1 This module contains a procedure which writes a D_floating in
36 0036 1 fixed-point notation to a textfile.
37 0037 1
38 0038 1 ENVIRONMENT: User mode - AST reentrant
39 0039 1
40 0040 1 AUTHOR: Steven B. Lionel, CREATION DATE: 1-April-1981
41 0041 1
42 0042 1 MODIFIED BY:
43 0043 1
44 0044 1 1-001 - Original. SBL 1-April-1981
45 0045 1 1-002 - Make total-width a longword. SBL 30-June-1982
46 0046 1 1-003 - Allocate space for field width fraction specification. DG 19-Oct-1983
47 0047 1 --
48 0048 1
```

PASSWRITE_REALF Write a D floating in F format
1-003 Declarations

B 7
16-Sep-1984 02:23:54
14-Sep-1984 12:52:06

VAX-11 Bliss-32 V4.0-742
[PASRTL.SRC]PASWRIRFD.B32;1

Page 2
(2)

```

50      0049 1 %SBTTL 'Declarations'
51      0050 1
52      0051 1 PROLOGUE DEFINITIONS:
53      0052 1
54      0053 1
55      0054 1 REQUIRE 'RTLIN:PASPROLOG';           ! Externals, linkages, PSECTs, structures
56      0118 1
57      0119 1
58      0120 1 TABLE OF CONTENTS:
59      0121 1
60      0122 1
61      0123 1 FORWARD ROUTINE
62      0124 1 PASSWRITE_REALF_D: NOVALUE,         ! Write to textfile
63      0125 1 PASSWRITEV_REALF_D: NOVALUE;         ! Write to string
64      0126 1
65      0127 1
66      0128 1 MACROS:
67      0129 1
68      0130 1 NONE
69      0131 1
70      0132 1 EQUATED SYMBOLS:
71      0133 1
72      0134 1 NONE
73      0135 1
74      0136 1 FIELDS:
75      0137 1
76      0138 1 NONE
77      0139 1
78      0140 1 OWN STORAGE:
79      0141 1
80      0142 1 NONE
81      0143 1
```

```

83      0144 1 %SBTTL 'PASSWRITE_REALF D - Write D_floating in F format to textfile'
84      0145 1 GLOBAL ROUTINE PASSWRITE_REALF_D (
85      0146 1     PFV: REF $PASSPFV_FILE_VARIABLE,      ! File variable
86      0147 1     VALUE_0,VALUE_1,                      ! Value to write
87      0148 1     TOTAL_WIDTH: SIGNED,                  ! Total field width
88      0149 1     FRAC_DIGITS: SIGNED,                  ! Digits in fraction
89      0150 1     ERROR                                  ! Error unwind address
90      0151 1     ): NOVALUE =
91      0152 1
92      0153 1 ++
93      0154 1 FUNCTIONAL DESCRIPTION:
94      0155 1
95      0156 1     This procedure writes a D_floating value in fixed-point notation
96      0157 1     to the specified textfile.
97      0158 1
98      0159 1 CALLING SEQUENCE:
99      0160 1
100     0161 1     CALL PASSWRITE_REALF_D (PFV.mr.r, VALUE.rd.v , TOTAL_WIDTH.rl.v
101     0162 1     , FRAC_DIGITS.rl.v [, ERROR.jā.r])
102     0163 1
103     0164 1 FORMAL PARAMETERS:
104     0165 1
105     0166 1     PFV          - The Pascal File Variable (PFV) passed by reference.
106     0167 1     The structure of the PFV is defined in PASPFV.REQ.
107     0168 1
108     0169 1     VALUE        - The D_floating value to write, by immediate value.
109     0170 1     Note that this takes up two argument list positions.
110     0171 1
111     0172 1     TOTAL_WIDTH   - Total field width.
112     0173 1
113     0174 1     FRAC_DIGITS   - Number of digits in fraction.
114     0175 1
115     0176 1     ERROR        - Optional. Address to unwind to if an error occurs.
116     0177 1
117     0178 1 IMPLICIT INPUTS:
118     0179 1
119     0180 1     NONE
120     0181 1
121     0182 1 IMPLICIT OUTPUTS:
122     0183 1
123     0184 1     NONE
124     0185 1
125     0186 1 ROUTINE VALUE:
126     0187 1
127     0188 1     NONE
128     0189 1
129     0190 1 SIDE EFFECTS:
130     0191 1
131     0192 1     If the file is the standard file OUTPUT, it is implicitly opened.
132     0193 1
133     0194 1 SIGNALLED ERRORS:
134     0195 1
135     0196 1     LINTOOLON - Line too long
136     0197 1     NEGWIDDIG - negative Width or Digits specification is not allowed
137     0198 1
138     0199 1 --
139     0200 1

```

```

: 140      0201 2      BEGIN
: 141      0202 2
: 142      0203 2      BUILTIN
: 143      0204 2      ACTUALCOUNT;
: 144      0205 2
: 145      0206 2      LOCAL
: 146      0207 2      FCB: REF $PASSFCB CONTROL_BLOCK,      ! File control block
: 147      0208 2      FIELD_WIDTH: SIGNED,                  ! Minimum/actual width
: 148      0209 2      REMAINING_WIDTH,                      ! Maximum width
: 149      0210 2      PFV_ADDR: VOLATILE,                   ! Enable argument
: 150      0211 2      UNWIND_ACT: VOLATILE,                 ! Enable argument
: 151      0212 2      ERROR_ADDR: VOLATILE;                 ! Enable argument
: 152      0213 2
: 153      0214 2      ENABLE
: 154      0215 2      PASS$IO_HANDLER (PFV_ADDR, UNWIND_ACT, ERROR_ADDR);      ! Enable error handler
: 155      0216 2
: 156      0217 2      !+
: 157      0218 2      ! Get ERROR parameter, if present.
: 158      0219 2      !-
: 159      0220 2
: 160      0221 2      IF ACTUALCOUNT () GEQU 6
: 161      0222 2      THEN
: 162      0223 2      ERROR_ADDR = .ERROR;                  ! Set unwind address
: 163      0224 2
: 164      0225 2      PFV_ADDR = PFV [PFV$R_PFV];            ! Set PFV address
: 165      0226 2
: 166      0227 2      !+
: 167      0228 2      ! Validate PFV and get PFV.
: 168      0229 2      !-
: 169      0230 2
: 170      0231 2      PASS$VALIDATE_PFV (PFV [PFV$R_PFV]; FCB);
: 171      0232 2
: 172      0233 2      !+
: 173      0234 2      ! Set unwind action to unlock file.
: 174      0235 2      !-
: 175      0236 2
: 176      0237 2      UNWIND_ACT = PASS$K_UNWIND_UNLOCK;
: 177      0238 2
: 178      0239 2      !+
: 179      0240 2      ! Do common initialization.
: 180      0241 2      !-
: 181      0242 2
: 182      0243 2      PASS$INIT_WRITE (PFV [PFV$R_PFV], FCB [FCB$R_FCB]; FCB);
: 183      0244 2
: 184      0245 2      !+
: 185      0246 2      ! Get field width and remaining width. Check for invalid width or digits.
: 186      0247 2      !-
: 187      0248 2
: 188      0249 2      FIELD_WIDTH = .TOTAL_WIDTH;
: 189      0250 2      IF (.FIELD_WIDTH LSS 0) OR (.FRAC_DIGITS LSS 0)
: 190      0251 2      THEN
: 191      0252 2      $PASS$IO_ERROR (PASS$NEGWIDDIG, 0);
: 192      0253 2      REMAINING_WIDTH = .FCB [FCB$A_RECORD_END] - .FCB [FCB$A_RECORD_CUR];
: 193      0254 2
: 194      0255 2      !+
: 195      0256 2      ! Convert to text and put in record buffer. If an error occurs, signal it.
: 196      0257 2      !-

```

PASSWRITE_REALF Write a D_floating in F format
1-003

PASSWRITE_REALF_D - Write D_floating in F forma

E 7
16-Sep-1984 02:23:54
14-Sep-1984 12:52:06

VAX-11 Bliss-32 V4.0-742
[PASRTL.SRC]PASWRIRFD.B32;1

Page 5
(3)

```

197      0258 2
198      0259 2
199      0260 2
200      0261 2
201      0262 2
202      0263 2
203      0264 2
204      0265 2
205      0266 2
206      0267 2
207      0268 2
208      0269 2
209      0270 2
210      0271 2
211      0272 2
212      0273 2
213      0274 2
214      0275 2
215      0276 2
216      0277 2
217      0278 2
218      0279 2
219      0280 2
220      0281 2
221      0282 1

IF NOT PASS$CVT_D_T (VALUE 0,
                     .FCB [FCB$A_RECORD_CUR],
                     FIELD_WIDTH,
                     .REMAINING_WIDTH,
                     .FRAC_DIGITS)
THEN
    $PASS$IO_ERROR (PASS$_LINTOOLON,1,(.FIELD_WIDTH-.REMAINING_WIDTH));
    +
    Advance buffer pointer.
    -
    FCB [FCB$A_RECORD_CUR] = .FCB [FCB$A_RECORD_CUR] + .FIELD_WIDTH;
    +
    Call WRITE epilogue routine to move the last character written to the
    user's buffer and to unlock the file variable.
    -
    PASS$END_WRITE (PFV [PFV$R_PFV], FCB [FCB$R_FCB]);
    RETURN;
    END;
```

! End of routine PASSWRITE_REALF_D

.TITLE PASSWRITE_REALF_D Write a D_floating in F forma

.IDENT \1-003\

.EXTRN PASSWRITE_REALF_D
.EXTRN PASSWRITE_REALF_D
.EXTRN PASS\$IO_HANDLER
.EXTRN PASS\$VACIDATE_PFV
.EXTRN PASS\$INIT_WRITE
.EXTRN PASS\$SIGNAL, PASS\$K_NEGWIDDIG
.EXTRN PASS\$CVT_D_T, PASS\$K_LINTOOLON
.EXTRN PASS\$END_WRITE

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

.ENTRY PASSWRITE_REALF_D, Save R2,R3,R4,R5,R6,R7,- ; 0145

			01FC 00000			
58	00000000G	00	9E 00002	MOVAB	PASS\$SIGNAL, R8	
5E		10	C2 00009	SUBL2	#15, SP	
	04	AE	7C 0000C	CLRQ	ERROR_ADDR	0201
	0C	AE	D4 0000F	CLRL	PFV_ADDR	
6D	006F	CF	DE 00012	MOVAL	5\$, (FP)	
06		6C	91 00017	CMPB	(AP), #6	0221
		05	1F 0001A	BLSSU	1\$	
04	AE	18	AC D0 0001C	MOVL	ERROR, ERROR_ADDR	0223
	56	04	AC D0 00021	MOVL	PFV, R6	0225
0C	AE	56	D0 00025	MOVL	R6, PFV_ADDR	
	00000000G	00	16 00029	JSB	PASS\$VACIDATE_PFV	0231
08	AE	01	D0 0002F	MOVL	#1, UNWIND_ACT	0237
	00000000G	00	16 00033	JSB	PASS\$INIT_WRITE	0243

```

; 222      0283 1
; 223      0284 1 !<BLF/PAGE>

```

```
225 0285 1 %SBTTL 'PASSWRITEV_REALF_D - Write F floating in F format to string'
226 0286 1 GLOBAL ROUTINE PASSWRITEV_REALF_D (
227 0287 1     MAX_LENGTH: WORD,                                ! Maximum length of string
228 0288 1     STRING_LINE: REF VECTOR [, WORD],                ! String to write to
229 0289 1     VALUE0,VALUE1,                                    ! Value to write
230 0290 1     TOTAL_WIDTH: SIGNED,                              ! Total field width
231 0291 1     FRAC_DIGITS: SIGNED,                             ! Fraction digits
232 0292 1     ERROR                                             ! Error unwind address
233 0293 1 ) : NOVALUE =
234 0294 1
235 0295 1 ++
236 0296 1 FUNCTIONAL DESCRIPTION:
237 0297 1
238 0298 1     This procedure writes a D_floating in fixed-point format
239 0299 1     to the specified string.
240 0300 1
241 0301 1 CALLING SEQUENCE:
242 0302 1
243 0303 1     CALL PASSWRITEV_REALF_D (MAX_LENGTH.rw.v, STRING_LINE.wvt.r,
244 0304 1     VALUE.rd.v, TOTAL_WIDTH.rl.v, FRAC_DIGITS.rl.v [, ERROR.j.r])
245 0305 1
246 0306 1 FORMAL PARAMETERS:
247 0307 1
248 0308 1     MAX_LENGTH      - The maximum length of STRING_LINE.
249 0309 1
250 0310 1     STRING_LINE    - A varying string to which the output will be appended.
251 0311 1
252 0312 1     VALUE          - The value to write.
253 0313 1
254 0314 1     TOTAL_WIDTH    - The width of the field to write.
255 0315 1
256 0316 1     FRAC_DIGITS    - The number of digits in the fraction.
257 0317 1
258 0318 1     ERROR          - Optional. If specified, the address to unwind to
259 0319 1                   in case of an error.
260 0320 1
261 0321 1 IMPLICIT INPUTS:
262 0322 1
263 0323 1     NONE
264 0324 1
265 0325 1 IMPLICIT OUTPUTS:
266 0326 1
267 0327 1     NONE
268 0328 1
269 0329 1 ROUTINE VALUE:
270 0330 1
271 0331 1     NONE
272 0332 1
273 0333 1 SIDE EFFECTS:
274 0334 1
275 0335 1     NONE
276 0336 1
277 0337 1 SIGNALLED ERRORS:
278 0338 1
279 0339 1     See PASSWRITE_REALF_D
280 0340 1
281 0341 1 --
```

```

282 0342 1
283 0343 2 BEGIN
284 0344 3
285 0345 4 LOCAL
286 0346 5     PFV: $PASSPFV FILE VARIABLE,      ! Pascal File Variable
287 0347 6     ARG_LIST: VECTOR [8, LONG],    ! Argument list
288 0348 7     PFV_ADDR: VOLATILE,            ! Enable argument
289 0349 8     UNWIND_ACT: VOLATILE,          ! Enable argument
290 0350 9     ERROR_ADDR: VOLATILE;          ! Enable argument
291 0351 10
292 0352 11 BUILTIN
293 0353 12     ACTUALCOUNT;                  ! Count of arguments
294 0354 13
295 0355 14 ENABLE
296 0356 15     PASS$IO_HANDLER (PFV_ADDR, UNWIND_ACT, ERROR_ADDR); ! Enable error handler
297 0357 16
298 0358 17 !+
299 0359 18 ! Get ERROR parameter, if present.
300 0360 19 !-
301 0361 20
302 0362 21 IF ACTUALCOUNT () GEQU 6
303 0363 22 THEN
304 0364 23     ERROR_ADDR = .ERROR;              ! Set unwind address
305 0365 24
306 0366 25 PFV_ADDR = PFV [PFV$R_PFV];          ! Set PFV address
307 0367 26
308 0368 27 !+
309 0369 28 ! Set up ARG_LIST.
310 0370 29 !-
311 0371 30
312 0372 31 ARG_LIST [0] = 5;                          ! Five arguments
313 0373 32 ARG_LIST [1] = PFV [PFV$R_PFV];      ! PFV address
314 0374 33 ARG_LIST [2] = .VALUE0;              ! Value to write
315 0375 34 ARG_LIST [3] = .VALUE1;
316 0376 35 ARG_LIST [4] = .TOTAL_WIDTH;         ! Field width
317 0377 36 ARG_LIST [5] = .FRAC_DIGITS;        ! Fraction digits
318 0378 37
319 0379 38 !+
320 0380 39 ! Call PASS$DO_WRITEV to do the work, giving it the address of
321 0381 40 ! PASSWRITE_REALF_D to call.
322 0382 41 !-
323 0383 42
324 0384 43 PASS$DO_WRITEV (PFV [PFV$R_PFV], .MAX_LENGTH, STRING_LINE [0], ARG_LIST,
325 0385 44     PASSWRITE_REALF_D);
326 0386 45
327 0387 46 RETURN;
328 0388 47
329 0389 48 END;

```

! End of routine PASSWRITEV_REALF_D

.EXTRN PASS\$DO_WRITEV

SE 007C 00000
 30 C2 00002
 7E D4 00005
 04 AE 7C 00007

.ENTRY PASSWRITEV_REALF_D, Save R2,R3,R4,R5,R6
 SUBL2 #48, SP
 CLRL ERROR_ADDR
 CLRQ UNWIND_ACT

: 0286
 :
 : 0343
 :

PASSWRITE_REALF Write a D floating in F format
1-003

PASSWRITEV_REALF_D - Write F_floating in F form

1 7
16-Sep-1984 02:23:54
14-Sep-1984 12:52:06

VAX-11 Bliss-32 V4.0-742
[PASRTL.SRC]PASWRIRFD.B32;1

Page 9
(4)

	6D	003E	CF	DE	0000A	MOVAL	2\$, (FP)	
	06		6C	91	0000F	CMPB	(AP), #6	0362
			04	1F	00012	BLSSU	1\$	
08	6E	1C	AC	D0	00014	MOVL	ERROR, ERROR_ADDR	0364
0C	AE	24	AE	9E	00018	MOVAB	PFV, PFV_ADDR	0366
10	AE		05	D0	0001D	MOVL	#5, ARG_LIST	0372
14	AE	24	AE	9E	00021	MOVAB	PFV, ARG_LIST+4	0373
1C	AE	0C	AC	7D	00026	MOVQ	VALUE0, ARG_LIST+8	0374
	AE	14	AC	7D	0002B	MOVQ	TOTAL_WIDTH, ARG_LIST+16	0376
	55	FF24	CF	9E	00030	MOVAB	PASSWRITE_REALF_D, R5	0384
	54	0C	AE	9E	00035	MOVAB	ARG_LIST, R4	
	56	24	AE	9E	00039	MOVAB	PFV, R6	
	53	08	AC	D0	0003D	MOVL	STRING_LINE, R3	
	52	04	AC	3C	00041	MOVZWL	MAX_LENGTH, R2	
		00000000G	00	16	00045	JSB	PASS\$DO_WRITEV	
				04	0004B	RET		0389
				0000	0004C	.WORD	Save nothing	0343
	50	08	AC	D0	0004E	MOVL	8(AP), R0	
	50	04	A0	D0	00052	MOVL	4(R0), R0	
		CC	A0	9F	00056	PUSHAB	ERROR_ADDR	
		D0	A0	9F	00059	PUSHAB	UNWIND_ACT	
		D4	A0	9F	0005C	PUSHAB	PFV_ADDR	
			03	DD	0005F	PUSHL	#3	
			5E	DD	00061	PUSHL	SP	
	7E	04	AC	7D	00063	MOVQ	4(AP), -(SP)	
00000000G	00		03	FB	00067	CALLS	#3, PASS\$IO_HANDLER	
			04	0006E	RET			

; Routine Size: 111 bytes, Routine Base: _PASS\$CODE + 00A8

; 330 0390 1
; 331 0391 1 !<BLF/PAGE>

PASSWRITE_REALF

1-003

Write a D floating in F format

PASSWRITE_REALF_D - Write F floating in F form

J 7

16-Sep-1984 02:23:54

VAX-11 Bliss-32 V4.0-742

[PASRTL.SRC]PASWRIRFD.B32;1

Page 10

(5)

333

0392

1

END

334

0393

1

335

0394

0

ELUDOM

! End of module PASSWRITE_REALF_D

PSECT SUMMARY

Name

Bytes

Attributes

_PASS\$CODE

279

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	0	0	581	00:01.0
\$255\$DUA28:[PASRTL.OBJ]PASLIB.L32;1	427	97	22	33	00:00.4

COMMAND QUALIFIERS

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

Size: 279 code + 0 data bytes

Run Time: 00:07.4

Elapsed Time: 00:16.2

Lines/CPU Min: 3194

Lexemes/CPU-Min: 12956

Memory Used: 82 pages

Compilation Complete

0298 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

PASWIREH
LIS

PASWIREL
LIS

PASWIREG
LIS

PASWISTR
LIS

PATCH

PATDEF
MDL

PASWIREF
LIS

PATCH
MAP

SRMDEF
MDL

PASWIRFH
LIS

PASWIRAR
LIS

PASWIREL
LIS

PASWIRLS
LIS

PASWIRINT
LIS

PASWIRIOCT
LIS

PASWIRFG
LIS

BSTRUC
REQ

PASWIRFF
LIS

CHRKEY
REQ

PASWIRFD
LIS