

[illegible]

```

CCCCCCCCC 000000 MM MM AAAAAA SSSSSSSS SSSSSSSS IIIIII GGGGGGGG NN NN
CCCCCCCCC 000000 MM MM AAAAAA SSSSSSSS SSSSSSSS IIIIII GGGGGGGG NN NN
CC 00 00 MMMM MMMM AA AA SS SS SSSSSSSS IIII GGGGGGGG NN NN
CC 00 00 MMMM MMMM AA AA SS SS SSSSSSSS IIII GGGGGGGG NN NN
CC 00 00 MM MM MM AA AA SS SS SSSSSSSS IIII GGGGGGGG NN NN
CC 00 00 MM MM MM AA AA SS SS SSSSSSSS IIII GGGGGGGG NN NN
CC 00 00 MM MM MM AA AA SS SS SSSSSSSS IIII GGGGGGGG NN NN
CC 00 00 MM MM MM AA AA SS SS SSSSSSSS IIII GGGGGGGG NN NN
CC 00 00 MM MM MM AA AA SS SS SSSSSSSS IIII GGGGGGGG NN NN
CC 00 00 MM MM MM AA AA SS SS SSSSSSSS IIII GGGGGGGG NN NN
CC 00 00 MM MM MM AA AA SS SS SSSSSSSS IIII GGGGGGGG NN NN
CCCCCCCCC 000000 MM MM AA AA SSSSSSSS SSSSSSSS IIIIII GGGGGG NN NN
CCCCCCCCC 000000 MM MM AA AA SSSSSSSS SSSSSSSS IIIIII GGGGGG NN NN

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
LLLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLLL IIIIII SSSSSSSS

```



```
1 0001 0 MODULE COM$ASSIGN (%TITLE'Compatibility CALL ASSIGN routine'  
2 0002 0 IDENT = '1-007', ! File: COM$ASSIGN.B32 Edit: sbl1007  
3 0003 0 LINKAGE (FORTRAN) ! Call-by reference  
4 0004 0 ) =  
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 * ALL RIGHTS RESERVED. *  
12 0012 1 *  
13 0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *  
14 0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE *  
15 0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER *  
16 0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *  
17 0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY *  
18 0018 1 * TRANSFERRED. *  
19 0019 1 *  
20 0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE *  
21 0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT *  
22 0022 1 * CORPORATION. *  
23 0023 1 *  
24 0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *  
25 0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL. *  
26 0026 1 *  
27 0027 1 *  
28 0028 1 *****  
29 0029 1  
30 0030 1  
31 0031 1 ++  
32 0032 1 FACILITY: FORTRAN Compatibility Library  
33 0033 1  
34 0034 1 ABSTRACT:  
35 0035 1  
36 0036 1 Assign logical name to unit to be in effect until call CLOSE  
37 0037 1  
38 0038 1 ENVIRONMENT: User Mode - AST re-entrant  
39 0039 1  
40 0040 1 AUTHOR: Thomas N. Hastings, CREATION DATE: 8-Aug-1977  
41 0041 1  
42 0042 1 MODIFIED BY:  
43 0043 1  
44 0044 1 Thomas N. Hastings, 31-Oct-1977: VERSION 0  
45 0045 1 [Previous edit history removed. SBL 1-March-1983]  
46 0046 1 1-001 - Update version number and copyright notice. JBS 16-NOV-78  
47 0047 1 1-002 - Declare NULLPARAMETER for the new BLISS compiler. JBS 22-NOV-78  
48 0048 1 1-003 - Change REQUIRE file names from FOR... to OTS... JBS 07-DEC-78  
49 0049 1 1-004 - Change OTSOPN back to FOROPN and change some OPEN  
50 0050 1 prefixes to LUB. JBS 13-DEC-78  
51 0051 1 1-005 - Use 32-bit addresses for externals. JBS 27-JAN-1979  
52 0052 1 1-006 - Move LIBRARY RTLSTARLE so FOROPN can use its symbols.  
53 0053 1 JBS 10-APR-1979  
54 0054 1 1-007 - Remove upcasing/blank-stripping logic from OPEN FNAME, as it hasn't  
55 0055 1 been needed since VAX/VMS V2.0. Use prologue file. SBL 1-Mar-1983  
56 0056 1 --  
57 0057 1
```


[illegible]

```
100 0163 1 GLOBAL ROUTINE ASSIGN (
101 0164 1     LOGICAL UNIT,
102 0165 1     NAME_ARRAYORDSC,
103 0166 1     NAME_COUNT)
104 0167 1     : NOVALUE =
105 0168 1
106 0169 1 ++
107 0170 1 FUNCTIONAL DESCRIPTION:
108 0171 1
109 0172 1     Assign filename string to logical unit until CALL CLOSE.
110 0173 1     Leading, trailing and imbedded spaces are removed.
111 0174 1     Lowercase is converted to upper case.
112 0175 1
113 0176 1 FORMAL PARAMETERS:
114 0177 1
115 0178 1     LOGICAL UNIT.rw.r           Adr. of word containing logical unit
116 0179 1     [NAME_ARRAYORDSC.rt{raids} Adr. of array of characters or adr. of descriptor of file name
117 0180 1     [NAME_COUNT.rw.r]]       Optional adr. or word containing cont of no. of characters
118 0181 1                             If count is 0, or arg omitted, scan array for null.
119 0182 1
120 0183 1 IMPLICIT INPUTS:
121 0184 1
122 0185 1     OTSS$A_CUR_LUB               Adr. of current LUB/ISB/RAB
123 0186 1     LUB$A_FAB                 Adr. of FAB if assigned to LUB
124 0187 1
125 0188 1 IMPLICIT OUTPUTS:
126 0189 1
127 0190 1     LUB$A_FAB                   Set to adr. of allocated FAB
128 0191 1
129 0192 1 ROUTINE VALUE:
130 0193 1 COMPLETION CODES:
131 0194 1
132 0195 1     NONE
133 0196 1
134 0197 1 SIDE EFFECTS:
135 0198 1
136 0199 1     Allocate LUB/ISB/RAB and FAB and RMS name block if not already
137 0200 1     SIGNAL STOPS the following errors:
138 0201 1     FOR$_INSVIRMEM = INSUFFICIENT VIRTUAL MEMORY
139 0202 1     FOR$_UNIALROPE = UNIT ALREADY OPENED
140 0203 1     OTSS$_FATINTERR = FATAL INTERNAL ERROR IN RUN-TIME LIBRARY
141 0204 1 --
142 0205 1
143 0206 2 BEGIN
144 0207 2
145 0208 2 BUILTIN
146 0209 2     NULLPARAMETER;
147 0210 2
148 0211 2 GLOBAL REGISTER
149 0212 2     CCB = 11 : REF $FOR$CCB_DECL;
150 0213 2
151 0214 2 MAP
152 0215 2     LOGICAL_UNIT : REF VECTOR [1, WORD], ! Pass word by-reference
153 0216 2     NAME_COUNT : REF VECTOR [1, WORD]; ! Pass word by-reference
154 0217 2
155 0218 2 LOCAL
156 0219 2     FAB : REF BLOCK [FAB$K_BLN, BYTE], ! base ptr to FAB
```



```
157      0220 2      NAME_DSC_PTR : REF DSC$DESCRIPTOR,      ! Pointer to name descriptor - either points to callers
158      0221 2      ! descriptor or to LOCAL NAME_DSC if caller passes by reference.
159      0222 2      NAME_DSC : DSC$DESCRIPTOR,      ! String descriptor for file name
160      0223 2      L_UNWIND_ACTION : VOLATILE;      ! Unwind action for error handler
161      0224 2
162      0225 2      ENABLE
163      0226 2      FOR$$ERR_OPECLO (L_UNWIND_ACTION);      ! Enable error handler
164      0227 2
165      0228 2      !+
166      0229 2      ! Indicate no unwind action until LUB/ISB/RAB successfully pushed down
167      0230 2      !-
168      0231 2
169      0232 2      L_UNWIND_ACTION = FOR$K_UNWINDNOP;
170      0233 2
171      0234 2      !+
172      0235 2      ! Pushdown I/O system and allocate LUB/ISB/RAB for logical unit if not already allocated
173      0236 2      ! SIGNAL_STOP if any errors.
174      0237 2      ! If no errors, change error UNWIND action to pop LUB/ISB/RAB if an unwind should occur.
175      0238 2      ! On return, CCB points to the current control block
176      0239 2      !-
177      0240 2
178      0241 2      FOR$$CB_PUSH (.LOGICAL_UNIT [0], LUB$K_LUN_MIN);
179      0242 2      L_UNWIND_ACTION = FOR$K_UNWINDPOP;
180      0243 2
181      0244 2      !+
182      0245 2      ! If unit already opened, SIGNAL_STOP FOR$ _UNIALROPE (43 = 'UNIT ALREADY OPENED')
183      0246 2      !-
184      0247 2
185      0248 2      IF .CCB [LUB$V_OPENED] THEN $FOR$$SIGNAL_STO (FOR$K_UNIALROPE);
186      0249 2
187      0250 2      !+
188      0251 2      ! If filename present, build a descriptor for string
189      0252 2      !-
190      0253 2
191      0254 2      IF NOT NULLPARAMETER (2)
192      0255 2      THEN
193      0256 2      BEGIN
194      0257 2
195      0258 2      !+
196      0259 2      ! Assume NAMEARRAYORDSC is adr. of descriptor (caller using a string expression)
197      0260 2      ! Note: the detection of whether a descriptor or not is not completely fool proof.
198      0261 2      !-
199      0262 2
200      0263 2      NAME_DSC_PTR = .NAME_ARRAYORDSC;
201      0264 2
202      0265 2      IF .NAME_DSC_PTR [DSC$W_LENGTH] GTRU 255 OR .NAME_DSC_PTR [DSC$B_DTYPE] NEQU DSC$K_DTYPE_T OR
203      0266 2      .NAME_DSC_PTR [DSC$B_CLASS] NEQU DSC$K_CLASS_S
204      0267 2      THEN
205      0268 2
206      0269 2      !+
207      0270 2      ! NAME_ARRAY really is an array, not a descriptor.
208      0271 2      ! Set pointer to LOCAL string descriptor NAME_DSC which will be filled in.
209      0272 2      ! Set length from NAME_COUNT if present and not 0, else scan string for null.
210      0273 2      ! Set pointer to start of string.
211      0274 2      !-
212      0275 2
213      0276 2      BEGIN
```

```
214      0277 4      NAME_DSC_PTR = NAME_DSC;
215      0278 4      NAME_DSC [DSC$A_POINTER] = .NAME_ARRAYORDSC;
216      0279 5      NAME_DSC [DSC$W_LENGTH] = (IF NOT NULLPARAMETER (3) THEN IF .NAME_COUNT [0] NEQ 0 THEN
217      0280 6          .NAME_COUNT [0] ELSE MINU ((CH$FIND_CH (OPEN$K_STR_MAX, .NAME_ARRAYORDSC, 0) -
218      0281 6          .NAME_ARRAYORDSC), OPEN$K_STR_MAX) ELSE MINU ((CH$FIND_CH (OPEN$K_STR_MAX,
219      0282 4          .NAME_ARRAYORDSC, 0) - .NAME_ARRAYORDSC), OPEN$K_STR_MAX));
220      0283 3      END;
221      0284 3
222      0285 3
223      0286 3      !+ If FAB not already allocated (no previous CALL FDBSET or CALL ASSIGN) set one up.
224      0287 3      Clear FAB$B_FNS and FAB$L_FNA to indicate filename string not allocated.
225      0288 3
226      0289 3
227      0290 3      IF (FAB = .CCB [LUB$A_FAB]) EQL 0
228      0291 3      THEN
229      0292 4          BEGIN
230      0293 4              FAB = FOR$GET_VM (FAB$K_BLN);
231      0294 4              CH$FILL (0, FAB$K_BLN, .FAB);
232      0295 4              CCB [LUB$A_FAB] = .FAB;
233      0296 4              FAB [FAB$B_BID] = FAB$C_BID;
234      0297 4              FAB [FAB$B_BLN] = FAB$K_BLN;
235      0298 4          END
236      0299 3      ELSE
237      0300 3
238      0301 3      !+ FAB already setup and a filename string already present (previous CALL ASSIGN) deallocate it.
239      0302 3
240      0303 3
241      0304 3
242      0305 4          BEGIN
243      0306 4
244      0307 4              IF .FAB [FAB$L_FNA] NEQA 0
245      0308 4              THEN
246      0309 5                  BEGIN
247      0310 5                      FOR$FREE_VM (.FAB [FAB$B_FNS], .FAB [FAB$L_FNA]);
248      0311 5                      FAB [FAB$B_FNS] = 0;
249      0312 5                      FAB [FAB$L_FNA] = 0;
250      0313 4                  END;
251      0314 4
252      0315 3              END;
253      0316 3
254      0317 3      !+ Copy string from user to dynamically allocated area
255      0318 3      and set FAB to point to string. Strip out leading, embedded and trailing spaces.
256      0319 3      Convert lower case to upper case. OPEN_FNAME expects FAB to be already allocated
257      0320 3      with no filename allocated.
258      0321 3
259      0322 3
260      0323 3
261      0324 3      OPEN_FNAME (.NAME_DSC_PTR, .FAB);
262      0325 3      END;
263      0326 2
264      0327 2      !+ Pop LUB/ISB/RAB to previous if any.
265      0328 2
266      0329 2
267      0330 2
268      0331 2      FOR$CB_POP ();
269      0332 2      RETURN
270      0333 1      END;

! return to caller leaving FAB allocated.
```


				.TITLE	COM\$ASSIGN Compatibility CALL ASSIGN routine	
				.IDENT	\1-007\	
				.EXTRN	FOR\$\$SIGNAL_STO	
				.EXTRN	FOR\$\$GET_VM, FOR\$\$FREE_VM	
				.EXTRN	FOR\$\$CB_PUSH, FOR\$\$CB_POP	
				.EXTRN	FOR\$\$ERR_OPECLO	
				.PSECT	_FOR\$CODE, NOWRT, SHR, PIC, 2	
			08FC 00000	.ENTRY	ASSIGN, Save R2,R3,R4,R5,R6,R7,R11	: 0163
5E		08	C2 00002	SUBL2	#8, SP	: 0206
		7E	D4 00005	CLRL	L UNWIND_ACTION	: 0232
6D	00DC	CF	DE 00007	MOVAL	14\$, (FP)	: 0241
6E		01	D0 0000C	MOVL	#1, L_UNWIND_ACTION	: 0242
		50	D4 0000F	CLRL	R0	: 0248
52	04	BC	3C 00011	MOVZWL	@LOGICAL UNIT, R2	
	00000000G	00	16 00015	JSB	FOR\$\$CB_PUSH	
		6E	D4 0001B	CLRL	L_UNWIND_ACTION	
0A	FC	AB	E9 0001D	BLBC	-2(CCB), -1\$	
		22	DD 00021	PUSHL	#34	
00000000G		01	FB 00023	CALLS	#1, FOR\$\$SIGNAL_STO	
			04 0002A	RET		
02		6C	91 0002B 1\$:	CMPB	(AP), #2	: 0254
		03	1E 0002E	BGEQU	3\$	
		00AD	31 00030 2\$:	BRW	13\$	
		08	AC D5 00033 3\$:	TSTL	8(AP)	
		F8	13 00036	BEQL	2\$	
52	08	AC	D0 00038	MOVL	NAME_ARRAYORDSC, R2	: 0263
57		52	D0 0003C	MOVL	R2, NAME_DSC_PTR	
00FF		67	B1 0003F	CMPW	(NAME_DSC_PTR), #255	: 0265
		0C	1A 00044	BGTRU	4\$	
0E	02	A7	91 00046	CMPB	2(NAME_DSC_PTR), #14	
		06	12 0004A	BNEQ	4\$	
01	03	A7	91 0004C	CMPB	3(NAME_DSC_PTR), #1	: 0266
		45	13 00050	BEQL	10\$	
57	04	AE	9E 00052 4\$:	MOVAB	NAME_DSC, NAME_DSC_PTR	: 0277
08		52	D0 00056	MOVL	R2, NAME_DSC+4	: 0278
		6C	91 0005A	CMPB	(AP), #3	: 0279
		1A	1F 0005D	BLSSU	6\$	
		0C	AC D5 0005F	TSTL	12(AP)	
		15	13 00062	BEQL	6\$	
		0C	BC B5 00064	TSTW	@NAME_COUNT	
		06	13 00067	BEQL	5\$	
51	0C	BC	3C 00069	MOVZWL	@NAME_COUNT, R1	: 0280
		24	11 0006D	BRB	9\$	
62	0064	8F	00 3A 0006F 5\$:	LOCC	#0, #100, (R2)	
		0A	13 00075	BEQL	7\$	
		0A	11 00077	BRB	8\$	: 0281
62	0064	8F	00 3A 00079 6\$:	LOCC	#0, #100, (R2)	
		02	12 0007F	BNEQ	8\$	
		51	D4 00081 7\$:	CLRL	R1	
		52	C2 00083 8\$:	SUBL2	R2, R1	: 0282
00000064		51	D1 00086	CMPL	R1, #100	: 0281
		04	1B 0008D	BLEQU	9\$	

0050	8F	00	04	51	64	8F	9A	0008F	MOVZBL	#100, R1	:		
				AE		51	B0	00093	9\$:	MOVW	R1, NAME_DSC	:	0279
				56	E8	AB	D0	00097	10\$:	MOVL	-24(CCB), FAB	:	0290
				7E	50	21	12	0009B		BNEQ	11\$	:	
		00000000G		00		8F	9A	0009D		MOVZBL	#80, -(SP)	:	0293
				56		01	FB	000A1		CALLS	#1, FOR\$\$GET_VM	:	
				6E		50	D0	000A8		MOVL	R0, FAB	:	
						00	2C	000AB		MOVCS	#0, (SP), #0, #80, (FAB)	:	0294
						66		000B2				:	
			E8	AB		56	D0	000B3		MOVL	FAB, -24(CCB)	:	0295
				66	5003	8F	B0	000B7		MOVW	#20483, (FAB)	:	0296
						19	11	000BC		BRB	12\$	:	0290
						2C	A6	D5 000BE	11\$:	TSTL	44(FAB)	:	0307
						14	13	000C1		BEQL	12\$	:	
						2C	A6	DD 000C3		PUSHL	44(FAB)	:	0310
		00000000G		7E	34	A6	9A	000C6		MOVZBL	52(FAB), -(SP)	:	
				00		02	FB	000CA		CALLS	#2, FOR\$\$FREE_VM	:	
						34	A6	94 000D1		CLRB	52(FAB)	:	0311
						2C	A6	D4 000D4		CLRL	44(FAB)	:	0312
						56	DD	000D7	12\$:	PUSHL	FAB	:	0324
						57	DD	000D9		PUSHL	NAME_DSC_PTR	:	
		0000V	CF			02	FB	000DB		CALLS	#2, OPEN_FNAME	:	
					00000000G	00	16	000E0	13\$:	JSB	FOR\$\$CB_POP	:	0331
						04		000E6		RET		:	0333
						0000		000E7	14\$:	.WORD	Save nothing	:	0206
				50	08	AC	D0	000E9		MOVL	8(AP), R0	:	
				50	04	A0	D0	000ED		MOVL	4(R0), R0	:	
					F4	A0	9F	000F1		PUSHAB	L_UNWIND_ACTION	:	
						01	DD	000F4		PUSHL	#T	:	
						5E	DD	000F6		PUSHL	SP	:	
		00000000G		7E	04	AC	7D	000F8		MOVQ	4(AP), -(SP)	:	
				00		03	FB	000FC		CALLS	#3, FOR\$\$ERR_OPECLO	:	
						04		00103		RET		:	

; Routine Size: 260 bytes, Routine Base: _FOR\$CODE + 0000

; 271 0334 1

```
273 0335 1 ROUTINE OPEN_FNAME (
274 0336 1     FNAME_DSC,
275 0337 1     FAB)
276 0338 1     : NOVALUE =
277 0339 1
278 0340 1 ++
279 0341 1 FUNCTIONAL DESCRIPTION:
280 0342 1
281 0343 1     Copies filename to heap storage.
282 0344 1
283 0345 1     Historical note: In VAX/VMS V1, this routine upcased the string
284 0346 1     and removed all embedded blanks. Since V2, this has not been
285 0347 1     necessary.
286 0348 1
287 0349 1 FORMAL PARAMETERS:
288 0350 1
289 0351 1     FNAME_DSC.rt.ds    Adr. of filename descriptor
290 0352 1     FAB.mz.r        Adr. of FAB
291 0353 1
292 0354 1 IMPLICIT INPUTS:
293 0355 1
294 0356 1     NONE
295 0357 1
296 0358 1 IMPLICIT OUTPUTS:
297 0359 1
298 0360 1     FAB$B_FNS        Size of string allocated
299 0361 1     FAB$L_FNA        Adr. of string
300 0362 1
301 0363 1 ROUTINE VALUE:
302 0364 1 COMPLETION CODES:
303 0365 1
304 0366 1     NONE
305 0367 1
306 0368 1 SIDE EFFECTS:
307 0369 1
308 0370 1     Allocates storage for edited filename string.
309 0371 1 --
310 0372 1
311 0373 2 BEGIN
312 0374 2
313 0375 2 MAP
314 0376 2     FNAME_DSC : REF DSC$DESCRIPTOR,
315 0377 2     FAB : REF BLOCK [FAB$C_BLN, BYTE];
316 0378 2
317 0379 2 LOCAL
318 0380 2     LENGTH;                ! length of edited string
319 0381 2
320 0382 2 ++
321 0383 2     Count number of characters in string. Set length in FAB.
322 0384 2 --
323 0385 2
324 0386 2     LENGTH = .FNAME_DSC [DSC$W_LENGTH];
325 0387 2
326 0388 2     IF .LENGTH EQLU 0 THEN $FOR$$$SIGNAL_STO (FOR$K_FILNAMSPE);
327 0389 2
328 0390 2     IF .LENGTH GTRU 255 THEN $FOR$$$SIGNAL_STO (FOR$K_INVARGFOR);
329 0391 2
```



```

: 330      0392 2      !+
: 331      0393 2      !- Allocate # of non-blank characters and save address and length in FAB
: 332      0394 2
: 333      0395 2
: 334      0396 2      FAB [FAB$L_FNA] = FOR$$GET_VM (.LENGTH);
: 335      0397 2      FAB [FAB$B_FNS] = .LENGTH;
: 336      0398 2
: 337      0399 2      !+
: 338      0400 2      !- Copy string.
: 339      0401 2
: 340      0402 2
: 341      0403 2      CH$MOVE (.LENGTH, .FNAME_DSC [DSC$A_POINTER], .FAB [FAB$L_FNA]);
: 342      0404 2
: 343      0405 2      RETURN;
: 344      0406 1      END;

```

! End of OPEN_FNAME routine

```

                                003C 00000 OPEN_FNAME:
                                .WORD      Save R2,R3,R4,R5
                                53          04 AC D0 00002      MOVL      FNAME_DSC, R3
                                54          63 3C 00006      MOVZWL   (R3), LENGTH
                                04          04 12 00009      BNEQ      1$
                                2B          2B DD 0000B      PUSHL     #43
                                0B          0B 11 0000D      BRB       2$
                                000000FF 8F          54 D1 0000F 1$: CMPL      LENGTH, #255
                                0A          0A 1B 00016      BLEQU     3$
                                30          30 DD 00018      PUSHL     #48
                                00000000G 00          01 FB 0001A 2$: CALLS    #1, FOR$$SIGNAL_STO
                                52          04 00021      RET
                                08          08 AC D0 00022 3$: MOVL      FAB, R2
                                54          54 DD 00026      PUSHL     LENGTH
                                00000000G 00          01 FB 00028      CALLS    #1, FOR$$GET_VM
                                2C          50 D0 0002F      MOVL      R0, 44(R2)
                                34          54 90 00033      MOV      LENGTH, 52(R2)
                                2C B2      04 A2          54 28 00037      MOV      LENGTH, @4(R3), @44(R2)
                                04 B3          54 04 0003D      RET

```

; Routine Size: 62 bytes, Routine Base: _FOR\$CODE + 0104

```

: 345      0407 1 END
: 346      0408 1
: 347      0409 0 ELUDOM

```

PSECT SUMMARY

Name	Bytes	Attributes
_FOR\$CODE	322	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	13	0	581	00:01.0
-\$255\$DUA28:[FORRTL.OBJ]FORLIB.L32;1	711	188	26	52	00:00.5
-\$255\$DUA28:[FORRTL.OBJ]RTLLIB.L32;1	36	0	0	8	00:00.1

COMMAND QUALIFIERS

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

: Size: 322 code + 0 data bytes
: Run Time: 00:08.9
: Elapsed Time: 00:31.9
: Lines/CPU Min: 2754
: Lexemes/CPU-Min: 11872
: Memory Used: 123 pages
: Compilation Complete

0178

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY