


```

LL      IIIIII  BBBB BBBB  GGGGGGGG  EEEEEEEEE  TTTTTTTTTT  FFFFFFFFFF  000000  RRRRRRRR
LL      IIIIII  BBBB BBBB  GGGGGGGG  EEEEEEEEE  TTTTTTTTTT  FFFFFFFFFF  000000  RRRRRRRR
LL      II      BB      GG      EE      TT      FF      00      RR      RR
LL      II      BB      GG      EE      TT      FF      00      RR      RR
LL      II      BB      GG      EE      TT      FF      00      RR      RR
LL      II      BB      GG      EE      TT      FF      00      RR      RR
LL      II      BBBB BBBB  GG      EE      TT      FFFFFFFF  00      RRRRRRRR
LL      II      BBBB BBBB  GG      EE      TT      FFFFFFFF  00      RRRRRRRR
LL      II      BB      GG      GG      EE      TT      FF      00      RR      RR
LL      II      BB      GG      GG      EE      TT      FF      00      RR      RR
LL      II      BB      GG      GG      EE      TT      FF      00      RR      RR
LL      II      BB      GG      GG      EE      TT      FF      00      RR      RR
LLLLLLLL  IIIIII  BBBB BBBB  GGGGGG  EEEEEEEEE  TT      FF      00      RR      RR
LLLLLLLL  IIIIII  BBBB BBBB  GGGGGG  EEEEEEEEE  TT      FF      00      RR      RR

```

```

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
LLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLL  IIIIII  SSSSSSSS

```

```
1 0001 0 MODULE LIB$GET_FOREIGN ( ! Get a "foreign" command line
2 0002 0 IDENT = '1-011' ! File: LIBGETFOR.B32 Edit: RKR1011
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1
30 0030 1 **
31 0031 1 FACILITY: General Purpose Library
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1
35 0035 1 Contains a routine to return the command line from the
36 0036 1 last entered CLI foreign command. An optional prompt
37 0037 1 string may be supplied.
38 0038 1
39 0039 1 ENVIRONMENT: User mode - AST reentrant
40 0040 1
41 0041 1 AUTHOR: Benn Schreiber and Steven Lionel, CREATION DATE: 18-May-1979
42 0042 1
43 0043 1 EDIT HISTORY:
44 0044 1
45 0045 1 1-001 - Original. Adapted from Benn Schreiber's MACRO version. SBL 18-May-1979
46 0046 1 1-002 - Call LIB$GET_COMMAND instead of LIB$GET_INPUT. SBL 10-Jul-1979
47 0047 1 1-003 - Optionally, return the number of characters returned. Also, use
48 0048 1 the STR facility for string copies. JBS 18-SEP-1979
49 0049 1 1-004 - Back to SYSS$INPUT if no command line. SBL 28-Sept-1979
50 0050 1 1-005 - If invoked via MCR, strip off first token. SBL 8-Oct-1979
51 0051 1 1-006 - If no prompt supplied, don't call LIB$GET_INPUT. This was
52 0052 1 the original design, but edit 1-003 changed it. SBL 16-Oct-1979
53 0053 1 1-007 - If LIB$GET_INPUT returns an error, return it. SBL 15-Jan-1980
54 0054 1 1-008 - Use LIB$ routines instead of STR$. SBL 22-Jan-1980
55 0055 1 1-009 - Don't store OUTLEN until after the copy. SPR 11-40365 SBL 27-Aug-1981
56 0056 1 1-010 - Add optional parameter "force prompt" and upcase string from
57 0057 1 LIB$GET_INPUT. SBL 4-Nov-1981
```

LIB\$GET_FOREIGN
1-011

6 7
16-Sep-1984 00:59:05 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:38:58 [LIBRTL.SRC]LIBGETFOR.B32;1

Page 2
(1)

: 58 0058 1 ! 1-011 - Change to do copying via LIB\$SCOPY_DXDX6. RKR 18-NOV-1981
: 59 0059 1 ! <BLF/PAGE>

```

61      0060 1  !
62      0061 1  ! SWITCHES:
63      0062 1  !
64      0063 1  !
65      0064 1  SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
66      0065 1  !
67      0066 1  ! LINKAGES:
68      0067 1  !
69      0068 1  REQUIRE 'RTLIN:STRLNK';           ! Linkage to LIB$SCOPY_DXDX6
70      0253 1  !
71      0254 1  !
72      0255 1  ! TABLE OF CONTENTS:
73      0256 1  !
74      0257 1  !
75      0258 1  FORWARD ROUTINE
76      0259 1  LIB$GET_FOREIGN;           ! Get foreign command line
77      0260 1  !
78      0261 1  !
79      0262 1  ! INCLUDE FILES:
80      0263 1  !
81      0264 1  !
82      0265 1  LIBRARY 'RTLSTARLE';
83      0266 1  !
84      0267 1  REQUIRE 'RTLIN:RTLPSECT';
85      0362 1  !
86      0363 1  !
87      0364 1  ! MACROS:
88      0365 1  !
89      0366 1  NONE
90      0367 1  !
91      0368 1  EQUATED SYMBOLS:
92      0369 1  !
93      0370 1  NONE
94      0371 1  !
95      0372 1  PSECT DEFINITIONS:
96      0373 1  !
97      0374 1  DECLARE_PSECTS (LIB);
98      0375 1  !
99      0376 1  OWN STORAGE:
100     0377 1  !
101     0378 1  NONE
102     0379 1  !
103     0380 1  EXTERNAL REFERENCES:
104     0381 1  !
105     0382 1  !
106     0383 1  EXTERNAL ROUTINE
107     0384 1  SYSSCLI,           ! Get command line from CLI
108     0385 1  LIB$SCOPY_DXDX6 : STRING_JSB, ! Copy strings of any type to
109     0386 1  ! any type
110     0387 1  LIB$SFREE1 DD,      ! Free a dynamic string
111     0388 1  LIB$GET_INPUT;      ! Get string from SYSS$INPUT:
112     0389 1  !
113     0390 1  EXTERNAL LITERAL
114     0391 1  LIB$INPSTRTRU,      ! Input string truncated
115     0392 1  LIB$_STRTRU;        ! String truncated (success)
116     0393 1  !
117     0394 1  EXTERNAL
```

LIB\$GET_FOREIGN
1-011

17
16-Sep-1984 00:59:05
14-Sep-1984 12:38:58

VAX-11 Bliss-32 V4.0-742
[LIBRTL.SRC]LIBGETFOR.B32;1

Page 4
(2)

; 118 0395 1 LIB\$AB_UPCASE;

! Upcase translate table

```
120 0396 1 GLOBAL ROUTINE LIB$GET_FOREIGN (      | Get a "foreign" command line
121 0397 1      GET_STRING: REF BLOCK [, BYTE]      | Returned string
122 0398 1      USER_PROMPT: REF BLOCK [, BYTE],    | Optional prompt
123 0399 1      OUTLEN: REF VECTOR [, WORD],        | Number of bytes stored in GET_STRING
124 0400 1      FORCE_PROMPT: REF VECTOR [, LONG]    | Force call to LIB$GET_INPUT
125 0401 1      ) =
126 0402 1
127 0403 1 ++
128 0404 1 ++ FUNCTIONAL DESCRIPTION:
129 0405 1
130 0406 1 LIB$GET_FOREIGN obtains text from the command line that
131 0407 1 activated the current image. The user program can then
132 0408 1 interpret this text in any manner it desires. Using this
133 0409 1 facility, user programs can be made to look like commands
134 0410 1 which are intrinsic to the system.
135 0411 1
136 0412 1 The most useful way of using LIB$GET_FOREIGN, is if it is
137 0413 1 called from a program activated by a DCL "foreign command",
138 0414 1 hence the name. A foreign command allows any image to be run
139 0415 1 by means of its own command name.
140 0416 1
141 0417 1 To define a foreign command, use the following DCL command:
142 0418 1
143 0419 1 $ command_name ::= $filespec
144 0420 1
145 0421 1 where "command_name" is the name of the foreign command you
146 0422 1 wish to define and "filespec" is the fully qualified file specification
147 0423 1 of the executable image to be run when "command_name" is invoked.
148 0424 1
149 0425 1 For example:
150 0426 1
151 0427 1 $ VULCAN ::= $DB0:[SPOCK]VULCAN.EXE
152 0428 1
153 0429 1 The "$" is required and must immediately precede the file specification.
154 0430 1
155 0431 1 Assuming that the above command VULCAN was defined, the command
156 0432 1 line:
157 0433 1
158 0434 1 $ VULCAN /OUTPUT=GANYMEDE TITAN.DAT
159 0435 1
160 0436 1 would start running the image DB0:[SPOCK]VULCAN.EXE. If that
161 0437 1 program then calls LIB$GET_FOREIGN, it can obtain the remainder
162 0438 1 of the command line, which in the example would be the string
163 0439 1
164 0440 1 "/OUTPUT=GANYMEDE TITAN.DAT"
165 0441 1
166 0442 1 (without the quotes, of course).
167 0443 1
168 0444 1 The user program can then analyze this returned string in any manner
169 0445 1 it desires. No interpretation is done by DCL.
170 0446 1
171 0447 1 If the image resides in the SYS$SYSTEM: directory, it can be
172 0448 1 run using the MCR command. Normally, the first token in the
173 0449 1 returned text would be the name of the image being run, but
174 0450 1 LIB$GET_FOREIGN makes a special check and skips over that token.
175 0451 1
176 0452 1 The optional parameters "user_prompt" and "force_prompt" allow
```

LIB\$GET_INPUT to be called to prompt the user for the command line if no text was entered on the line that activated the program. To use this feature, specify a prompt string for "user_prompt"; if no text is found on the command line, LIB\$GET_INPUT will be called to prompt for the text. Text returned from LIB\$GET_INPUT is translated to upper case so as to minimize the differences between prompted-for text and command-line text.

Some applications may wish to call LIB\$GET_FOREIGN in a loop, having it retrieve the command line on the first invocation, but prompting with LIB\$GET_INPUT on the second and subsequent invocations. To accomplish this, specify the parameter "force_prompt" as an integer variable which is initialized before the loop to zero. LIB\$GET_FOREIGN will set this variable to 1 before returning. If "force_prompt" is 1 upon entry to LIB\$GET_FOREIGN, LIB\$GET_INPUT will be called directly. Note that "user_prompt" must also be specified for prompting to occur.

If no command line is present and if "user_prompt" is omitted, a zero-length string is returned.

CALLING SEQUENCE:

```
status.wlc.v = LIB$GET_FOREIGN (get_string.wt.dx [, [user_prompt.rt.dx]
                                [, [outlen.ww.r] [, "force_prompt.m(.r)"]])
```

FORMAL PARAMETERS:

get_string	- A string passed by descriptor into which is written the command line.
user_prompt	- An optional string passed by descriptor which is used for the prompt in the case that there is no information on the command line. If omitted, no prompting is done.
outlen	- The number of bytes stored in get_string, not counting blank padding.
force_prompt	- An optional longword, passed by reference. If zero, the command line will be returned if present. If one, LIB\$GET_INPUT will be called instead. "force_prompt" is set to 1 before returning.

IMPLICIT INPUTS:

NONE

IMPLICIT OUTPUTS:

NONE

COMPLETION CODES:

NONE

```
234 0510 1 | SSS NORMAL - Successful completion
235 0511 1 | LIB$INSVIRMEM - There was insufficient virtual memory to
236 0512 1 | copy the string to the dynamic class output
237 0513 1 | string.
238 0514 1 | LIB$INVSTRDES - A string descriptor had an invalid format.
239 0515 1 | LIB$INPSTRTRU - The command line was too big to fit in the
240 0516 1 | output string. It was truncated.
241 0517 1 | RMSS_XXX - RMS errors returned by LIB$GET_INPUT.
242 0518 1 |
243 0519 1 |
244 0520 1 | SIDE EFFECTS:
245 0521 1 |
246 0522 1 | NONE
247 0523 1 |
248 0524 1 | --
249 0525 1 |
250 0526 2 | BEGIN
251 0527 2 |
252 0528 2 | BUILTIN
253 0529 2 | NULLPARAMETER;
254 0530 2 |
255 0531 2 | LOCAL
256 0532 2 | DYNAM_LEN, ! Length of dynamic string
257 0533 2 | PROMPT_FLAG, ! Indicate whether or not to prompt
258 0534 2 | RET_STATUS, ! Return status
259 0535 2 | LGI_STATUS, ! Return status from LIB$GET_INPUT
260 0536 2 | CLI_REQ_BLOCK : BLOCK [CLISC_REQDESC, BYTE], ! CLI request block
261 0537 2 | CLI_STATUS; ! Return status from SYS$CLI
262 0538 2 |
263 0539 2 | +
264 0540 2 | Initialize CLI command request block.
265 0541 2 | -
266 0542 2 | CH$FILL (0, CLISC_REQDESC, CLI_REQ_BLOCK);
267 0543 2 | CLI_REQ_BLOCK [CLISB_RQTYPE] = CLISK_GETCMD; ! Request type = Get Command
268 0544 2 | DYNAM_LEN = 0; ! No dynamic string yet
269 0545 2 | +
270 0546 2 | Determine if force_prompt is present. If so and if 0, go ahead and call
271 0547 2 | SYS$CLI. If 1, don't call SYS$CLI but call LIB$GET_INPUT.
272 0548 2 | -
273 0549 2 |
274 0550 2 | IF NULLPARAMETER (4)
275 0551 2 | THEN
276 0552 2 | PROMPT_FLAG = 0
277 0553 2 | ELSE
278 0554 2 | BEGIN
279 0555 2 | PROMPT_FLAG = .FORCE_PROMPT [0];
280 0556 2 | FORCE_PROMPT [0] = 1;
281 0557 2 | END;
282 0558 2 |
283 0559 2 | +
284 0560 2 | If PROMPT_FLAG is 0, then call SYS$CLI.
285 0561 2 | -
286 0562 2 |
287 0563 2 | IF NOT .PROMPT_FLAG
288 0564 2 | THEN
289 0565 2 | BEGIN
290 0566 2 |
```

```
0567
0568
0569
0570
0571
0572
0573
0574
0575
0576
0577
0578
0579
0580
0581
0582
0583
0584
0585
0586
0587
0588
0589
0590
0591
0592
0593
0594
0595
0596
0597
0598
0599
0600
0601
0602
0603
0604
0605
0606
0607
0608
0609
0610
0611
0612
0613
0614
0615
0616
0617
0618
0619
0620
0621
0622
0623

+ Get the command line from the CLI, if any.
- CLI_STATUS = SYS$CLI (CLI_REQ_BLOCK);

+ If the image was invoked via MCR, skip over the first token which
- is the name of the image.

IF .CLI_STATUS AND .CLI_REQ_BLOCK [CLISB_RQSTAT] EQLU CLISK_VERB_MCR
THEN
  BEGIN
    LOCAL
      POINTER,          ! Local pointer
      LENGTH,           ! Local length
      ORIG_LENGTH,      ! Original length
      CLI_DSC : REF BLOCK [8, BYTE]; ! Pointer to CLI descriptor
    CLI_DSC = CLI_REQ_BLOCK [CLISQ_RQDESC];
    POINTER = CH$FIND_CH (.CLI_DSC [DSC$W_LENGTH], .CLI_DSC [DSC$A_POINTER], %C' ');
    IF CH$FAIL (.POINTER) ! Blank not found?
    THEN
      CLI_DSC [DSC$W_LENGTH] = 0
    ELSE
      BEGIN
        ORIG_LENGTH = .CLI_DSC [DSC$W_LENGTH];
        LENGTH = .ORIG_LENGTH - CH$DIFF (.POINTER, .CLI_DSC [DSC$A_POINTER]);
        POINTER = CH$FIND_NOT_CH (.LENGTH, .POINTER, %C' ');
        IF CH$FAIL (.POINTER)
        THEN
          CLI_DSC [DSC$W_LENGTH] = 0
        ELSE
          BEGIN
            LENGTH = .ORIG_LENGTH - CH$DIFF (.POINTER, .CLI_DSC [DSC$A_POINTER]);
            CLI_DSC [DSC$W_LENGTH] = .LENGTH;
            CLI_DSC [DSC$A_POINTER] = .POINTER;
          END;
        END;
      END;
    END;
  END
ELSE
  CLI_STATUS = 0;

+ Check for failure or null command line. If so, call
- LIB$GET_INPUT if user supplied a prompt else return 0 length string.

LGI_STATUS = 1;          ! Assume success
IF (NOT .CLI_STATUS) OR (.CLI_REQ_BLOCK [CLISW_RQSIZE] EQL 0)
THEN
  BEGIN
    LOCAL
      CLI_DSC : REF BLOCK [8, BYTE];
    CLI_DSC = CLI_REQ_BLOCK [CLISQ_RQDESC];          ! Descriptor address
    CLI_DSC [DSC$B_CLASS] = DSC$K_CLASS_D;          ! Dynamic
    CLI_DSC [DSC$B_DTYPE] = DSC$K_DTYPE_T;          ! Text
```

```

348      CLI_DSC [DSC$W_LENGTH] = 0;
349      CLI_DSC [DSC$A_POINTER] = 0;
350
351      ! 0-Length string
352
353      !+
354      ! If the user supplied a prompt, then call LIB$GET_INPUT
355      ! for the string.
356      !-
357      IF NOT NULLPARAMETER (2)
358      THEN
359          LGI_STATUS = LIB$GET_INPUT (.CLI_DSC, .USER_PROMPT);
360          DYNAM_LEN = .CLI_DSC [DSC$W_LENGTH];
361
362      !+
363      ! Upcase the string
364      !-
365      CH$TRANSLATE (LIB$AB_UPCASE, .CLI_DSC [DSC$W_LENGTH],
366                  .CLI_DSC [DSC$A_POINTER], %C',
367                  .CLI_DSC [DSC$W_LENGTH], .CLI_DSC [DSC$A_POINTER]);
368      END;
369
370      !+
371      ! Copy the string to the user's string with correct semantics.
372      !-
373      RET_STATUS = LIB$SCOPY DXDX6 (CLI_REQ_BLOCK [CLI$Q_RQDESC], .GET_STRING);
374      IF .RET_STATUS EQLU LIB$_STRTRU
375      THEN
376          RET_STATUS = LIB$_INPSTRTRU;
377
378      !+
379      ! If the user wants the length returned, return it.
380      !-
381      IF NOT NULLPARAMETER (3)
382      THEN
383          OUTLEN [0] = MINU (.CLI_REQ_BLOCK [CLI$W_RQSIZE], .GET_STRING [DSC$W_LENGTH]);
384
385      !+
386      ! Free our temporary string if we have one.
387      !-
388      IF .DYNAM_LEN GTR 0
389      THEN
390          LIB$SFREE1_DD (CLI_REQ_BLOCK [CLI$Q_RQDESC]);
391
392      !+
393      ! Return to our caller with the worst return status.
394      !-
395      IF .LGI_STATUS
396      THEN
397          RETURN .RET_STATUS
398      ELSE
399          RETURN .LGI_STATUS;
400
401      END;
402
403      ! End of routine LIB$GET_FOREIGN
404
```

				.TITLE	LIB\$GET_FOREIGN	
				.IDENT	\1-011\	
				.EXTRN	SYSSCLI, LIB\$SCOPY, DXDX6	
				.EXTRN	LIB\$FREE1 DD, LIB\$GET_INPUT	
				.EXTRN	LIB\$INPSTRTRU, LIB\$_STRTRU	
				.EXTRN	LIB\$AB_UPCASE	
				.PSECT	_LIB\$CODE, NOWRT, SHR, PIC, 2	
				.ENTRY	LIB\$GET_FOREIGN, Save R2,R3,R4,R5,R6,R7,R8	: 0396
1C	00	5E	01FC 00000	SUBL2	#28, SP	: 0542
		6E	00 2C 00005	MOVCS	#0, (SP), #0, #28, CLI_REQ_BLOCK	: 0543
		6E	01 90 0000B	MOVB	#1, CLI_REQ_BLOCK	: 0544
		04	58 D4 0000E	CLRL	DYNAM_LEN	: 0550
			6C 91 00010	CMPB	(AP), #4	: 0552
			05 1F 00013	BLSSU	1\$	: 0555
			10 AC D5 00015	TSTL	16(AP)	: 0556
			04 12 00018	BNEQ	2\$	: 0563
			50 D4 0001A 1\$:	CLRL	PROMPT_FLAG	: 0570
			08 11 0001C	BRB	3\$	: 0577
		50	10 BC D0 0001E 2\$:	MOVL	@FORCE_PROMPT, PROMPT_FLAG	: 0585
		BC	01 D0 00022	MOVL	#1, @FORCE_PROMPT	: 0586
		54	50 E8 00026 3\$:	BLBS	PROMPT_FLAG, 8\$	: 0587
			5E DD 00029	PUSHL	SP	: 0592
	00000000G	00	01 FB 0002B	CALLS	#1, SYSSCLI	: 0593
		55	50 D0 00032	MOVL	R0, CLI_STATUS	: 0594
		47	55 E9 00035	BLBC	CLI_STATUS, 9\$	: 0595
		21	03 AE 91 00038	CMPB	CLI_REQ_BLOCK+3, #33	: 0597
			41 12 0003C	BNEQ	9\$	: 0599
		52	08 AE 9E 0003E	MOVAB	CLI_REQ_BLOCK+8, CLI_DSC	: 0600
04	B2	62	20 3A 00042	LOCC	#32, (CLI_DSC), @4(CLI_DSC)	: 0601
			02 12 00047	BNEQ	4\$	: 0602
			51 D4 00049	CLRL	R1	: 0607
			51 D5 0004B 4\$:	TSTL	POINTER	: 0608
			18 13 0004D	BEQL	6\$	: 0609
		54	62 3C 0004F	MOVZWL	(CLI_DSC), ORIG_LENGTH	: 0610
		A2	51 C3 00052	SUBL3	POINTER, 4(CLI_DSC), R0	: 0615
	50	04	54 C1 00057	ADDL3	ORIG_LENGTH, R0, LENGTH	: 0616
	53		20 3B 0005B	SKPC	#32, LENGTH, (POINTER)	: 0617
	61	53	02 12 0005F	BNEQ	5\$	: 0618
			51 D4 00061	CLRL	R1	: 0619
			51 D5 00063 5\$:	TSTL	POINTER	: 0620
			04 12 00065	BNEQ	7\$	: 0621
			62 B4 00067 6\$:	CLRW	(CLI_DSC)	: 0622
			14 11 00069	BRB	9\$	: 0623
		A2	51 C3 0006B 7\$:	SUBL3	POINTER, 4(CLI_DSC), R0	: 0624
	50	04	54 C1 00070	ADDL3	ORIG_LENGTH, R0, LENGTH	: 0625
	53		53 B0 00074	MOVW	LENGTH, (CLI_DSC)	: 0626
		A2	51 D0 00077	MOVL	POINTER, 4(CLI_DSC)	: 0627
			02 11 0007B	BRB	9\$	: 0628
			55 D4 0007D 8\$:	CLRL	CLI_STATUS	: 0629
		57	01 D0 0007F 9\$:	MOVL	#1, LGI_STATUS	: 0630
		05	55 E9 00082	BLBC	CLI_STATUS, 10\$	: 0631

			08	AE	B5	00085	TSTW	CLI_REQ_BLOCK+8		
			37	12	00088	BNEQ	12\$			
	52		08	AE	9E	0008A	10\$:	MOVAB	CLI_REQ_BLOCK+8, CLI_DSC	0621
	62	020E0000	00	8F	D0	0008E		MOVL	#3471936, (CLI_DSC)	0624
			04	A2	D4	00095		CLRL	4(CLI_DSC)	0625
	02			6C	91	00098		CMPB	(AP), #2	0631
				14	1F	0009B		BLSSU	11\$	
			08	AC	D5	0009D		TSTL	8(AP)	
				0F	13	000A0		BEQL	11\$	
			08	AC	DD	000A2		PUSHL	USER_PROMPT	0633
				52	DD	000A5		PUSHL	CLI_DSC	
	00000000G	00		02	FB	000A7		CALLS	#2, LIB\$GET_INPUT	
		57		50	D0	000AE		MOVL	R0, LGI_STATUS	
		58		62	3C	000B1	11\$:	MOVZWL	(CLI_DSC), DYNAM_LEN	0634
00000000G	00			62	2E	000B4		MOVTC	(CLI_DSC), @4(CLI_DSC), #32, -	0642
		04		62		000BE			LIB\$AB_UPCASE, (CLI_DSC), @4(CLI_DSC)	
		50	08	AE	9E	000C1	12\$:	MOVAB	CLI_REQ_BLOCK+8, R0	0649
		51	04	AC	D0	000C5		MOVL	GET_STRING, R1	
				00	16	000C9		JSB	LIB\$COPY_DXD6	
		52	00000000G	50	D0	000CF		MOVL	R0, RET_STATUS	
		8F		52	D1	000D2		CMPB	RET_STATUS, #LIB\$_STRTRU	0650
				07	12	000D9		BNEQ	13\$	
		52	00000000G	8F	D0	000DB		MOVL	#LIB\$_INPSTRTRU, RET_STATUS	0652
		03		6C	91	000E2	13\$:	CMPB	(AP), #3	0658
				17	1F	000E5		BLSSU	15\$	
			0C	AC	D5	000E7		TSTL	12(AP)	
				12	13	000EA		BEQL	15\$	
		50	08	AE	3C	000EC		MOVZWL	CLI_REQ_BLOCK+8, R0	0660
		50	04	BC	B1	000F0		CMPW	@GET_STRING, R0	
				04	1E	000F4		BGEQU	14\$	
		50	04	BC	3C	000F6		MOVZWL	@GET_STRING, R0	
	0C	BC		50	B0	000FA	14\$:	MOVW	R0, @OUTLEN	
				58	D5	000FE	15\$:	TSTL	DYNAM_LEN	0666
				0A	15	00100		BLEQ	16\$	
			08	AE	9F	00102		PUSHAB	CLI_REQ_BLOCK+8	0668
	00000000G	00		01	FB	00105		CALLS	#1, LIB\$FREE1_DD	
		04		57	E9	0010C	16\$:	BLBC	LGI_STATUS, 17\$	0674
		50		52	D0	0010F		MOVL	RET_STATUS, R0	0678
					04	00112		RET		
		50		57	D0	00113	17\$:	MOVL	LGI_STATUS, R0	
					04	00116		RET		0680

; Routine Size: 279 bytes, Routine Base: _LIB\$CODE + 0000

:	405	0681	1
:	406	0682	1 END
:	407	0683	1
:	408	0684	0 ELUDOM

!End of module LIB\$GET_FOREIGN

PSECT SUMMARY

LIB\$GET_FOREIGN
1-011

D 8
16-Sep-1984 00:59:05
14-Sep-1984 12:38:58

VAX-11 Bliss-32 V4.0-742
[LIBRTL.SRC]LIBGETFOR.B32;1

Page 12
(3)

```
:      Name                Bytes                Attributes
:  _LIB$CODE                279 NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)
```

Library Statistics

```
:      File                ----- Symbols ----- Pages      Processing
:                               Total   Loaded   Percent   Mapped      Time
:  _$255$DUA28:[SYSLIB]STARLET.L32;1    9776      13      0      581      00:00.7
```

COMMAND QUALIFIERS

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

```
: Size:      279 code + 0 data bytes
: Run Time:   00:06.4
: Elapsed Time: 00:32.9
: Lines/CPU Min: 6432
: Lexemes/CPU-Min: 28664
: Memory Used: 119 pages
: Compilation Complete
```

0207 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

