

DDDDDDDDDDDDDD		CCCCCCCCCCCC	LLL
DDDDDDDDDDDDDD		CCCCCCCCCCCC	LLL
DDDDDDDDDDDDDD		CCCCCCCCCCCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDD	DDD	CCC	LLL
DDDDDDDDDDDDDD		CCCCCCCCCCCC	LLLLLLLLLLLLLLLL
DDDDDDDDDDDDDD		CCCCCCCCCCCC	LLLLLLLLLLLLLLLL
DDDDDDDDDDDDDD		CCCCCCCCCCCC	LLLLLLLLLLLLLLLL

```
CCCCCCCC LL      IIIIII NN      NN      TTTTTTTTTT
CCCCCCCC LL      IIIIII NN      NN      TTTTTTTTTT
CC        LL      II      NN      NN      TT
CC        LL      II      NN      NN      TT
CC        LL      II      NN      NN      TT
CC        LL      II      NN      NN      TT
CC        LL      II      NN      NN      TT
CC        LL      II      NN      NN      TT
CC        LL      II      NN      NN      TT
CC        LL      II      NN      NN      TT
CCCCCCCC LL      IIIIII NN      NN      TT
CCCCCCCC LL      IIIIII NN      NN      TT
          LLLLLLLLLL
          LLLLLLLLLL
```

```
....
....
....
....
```

```
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 cli$interface (IDENT='V04-000',
2 0002 0 ADDRESSING_MODE(NONEXTERNAL=LONG_RELATIVE,
3 0003 0 EXTERNAL=GENERAL)) =
4 0004 1 BEGIN
5 0005 1
6 0006 1
7 0007 1 *****
8 0008 1 *
9 0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
10 0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
11 0011 1 * ALL RIGHTS RESERVED.
12 0012 1 *
13 0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
14 0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
15 0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
16 0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
17 0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
18 0018 1 * TRANSFERRED.
19 0019 1 *
20 0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
21 0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
22 0022 1 * CORPORATION.
23 0023 1 *
24 0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
25 0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
26 0026 1 *
27 0027 1 *
28 0028 1 *****
29 0029 1
30 0030 1
31 0031 1 ++
32 0032 1 FACILITY: Command language interface routines
33 0033 1
34 0034 1 ABSTRACT:
35 0035 1
36 0036 1 These routines are used to enable a newly activated
37 0037 1 image to obtain the command parameters and qualifiers
38 0038 1 from the command language interpreter.
39 0039 1
40 0040 1 ENVIRONMENT:
41 0041 1
42 0042 1 VAX/VMS operating system. unprivileged user mode.
43 0043 1
44 0044 1 AUTHOR: Peter George, October 1981
45 0045 1
46 0046 1 Modified by:
47 0047 1
48 0048 1 V03-011 PCG0013 Peter George 17-Feb-1983
49 0049 1 Include INTDEF instead of CLIDEF.
50 0050 1
51 0051 1 V03-010 PCG0012 Peter George 16-Feb-1983
52 0052 1 Fix typo.
53 0053 1
54 0054 1 V03-009 PCG0011 Peter George 08-Feb-1983
55 0055 1 No need to resignal STR$COPY_DX errors.
56 0056 1
57 0057 1 V03-008 PCG0010 Peter George 14-Dec-1982
```

```

58      0058 1  Allocate local descriptor for command string.
59      0059 1
60      0060 1  V03-007 PCG0009      Peter George      29-Nov-1982
61      0061 1  Add argument to str$analyze_sdesc call.
62      0062 1
63      0063 1  V03-006 PCG0008      Peter George      22-Nov-1982
64      0064 1  Add CLISNEXT QUAL. Fix optional length assignment
65      0065 1  in CLISGET_VALUE. Analyze input string descriptors.
66      0066 1
67      0067 1  V03-005 PCG0007      Peter George      18-Oct-1982
68      0068 1  Add prompt argument to CLISDCL_PARSE.
69      0069 1
70      0070 1  V03-004 PCG0006      Peter George      12-Oct-1982
71      0071 1  Remove declaration of CLISEND_PARSE.
72      0072 1
73      0073 1  V03-003 PCG0005      Peter George      10-Oct-1982
74      0074 1  Look in SYSS$LIBRARY for UTILDEF.
75      0075 1
76      0076 1  V03-002 PCG0004      Peter George      30-Sep-1982
77      0077 1  Use INT data structure.
78      0078 1  Add prompt and continuation routines to CLISDCL_PARSE.
79      0079 1  Add value address to CLISGET VALUE.
80      0080 1  Add user argument to CLISDISPATCH.
81      0081 1  Delete CLISEND_PARSE.
82      0082 1
83      0083 1  V03-001 PCG0003      Peter George      18-Jun-1982
84      0084 1  Add CLIS_LOCPRES and CLIS_LOCNEG statuses.
85      0085 1  Signal errors from STR$COPY_DX.
86      0086 1
87      0087 1  --
88      0088 1
89      0089 1
90      0090 1  Include files
91      0091 1
92      0092 1  LIBRARY 'SYSS$LIBRARY:STARLET';
93      0093 1
94      0094 1  REQUIRE 'SYSS$LIBRARY:UTILDEF';      ! Common VMS BLISS definitions
95      0270 1
96      0271 1  REQUIRE 'LIB$:INTDEF';      ! Interface definitions
```


98	0297	1	!		
99	0298	1	!	Table of contents	
100	0299	1	!		
101	0300	1	!	FORWARD ROUTINE	
102	0301	1	!	cli\$present,	! Determine if entity present
103	0302	1	!	cli\$get_value,	! Get value of entity
104	0303	1	!	cli\$dcl_parse,	! Parse a command line
105	0304	1	!	cli\$dispatch,	! Dispatch to user processing routine
106	0305	1	!	cli\$next_qual;	! Get next qualifier
107	0306	1	!		
108	0307	1	!	EXTERNAL ROUTINE	
109	0308	1	!	sys\$cli,	! CLI callback routine
110	0309	1	!	lib\$get_vm,	! Allocate virtual memory
111	0310	1	!	lib\$free_vm,	! Deallocate virtual memory
112	0311	1	!	str\$analyze_sdesc,	! Analyze string descriptor
113	0312	1	!	str\$copy_dx;	! Copy to any class string
114	0313	1	!		
115	0314	1	!	!	
116	0315	1	!	External message definitions	
117	0316	1	!		
118	0317	1	!	EXTERNAL LITERAL	
119	0318	1	!	cli\$_nocmd,	! Prompt was ctrl/z-ed
120	0319	1	!	cli\$_concat,	! Value is concatenated
121	0320	1	!	cli\$_comma,	! Value is comma separated
122	0321	1	!	cli\$_present,	! Entity is explicitly globally present
123	0322	1	!	cli\$_negated,	! Entity is explicitly not globally present
124	0323	1	!	cli\$_locpres,	! Entity is explicitly locally present
125	0324	1	!	cli\$_locneg,	! Entity is explicitly not locally present
126	0325	1	!	cli\$_defaulted,	! Entity is implicitly present
127	0326	1	!	cli\$_absent;	! Entity is implicitly not present

```
129 0327 1 GLOBAL ROUTINE cli$present (name) =
130 0328 1
131 0329 1 ---
132 0330 1
133 0331 1 Determine if an entity is present on the command line.
134 0332 1
135 0333 1 Inputs:
136 0334 1
137 0335 1 name = Address of entity name descriptor
138 0336 1
139 0337 1 Outputs:
140 0338 1
141 0339 1 routine value = True if present, else false.
142 0340 1 ---
143 0341 1
144 0342 2 BEGIN
145 0343 2
146 0344 2 LOCAL
147 0345 2 req_desc : BBLOCK [cli$c_reqdesc], ! Request descriptor block
148 0346 2 rpw : BBLOCK [cli$c_workarea], ! Result parse work area
149 0347 2 req_flags : BITVECTOR [32]; ! Request flags array
150 0348 2
151 0349 2 CH$FILL(0, cli$c_reqdesc, req_desc); ! Zero request descriptor block
152 0350 2 req_desc [int_b_type] = cli$k_present; ! Set request type
153 0351 2 req_desc [int_l_getvm] = lib$get_vm; ! Set address of get vm routine
154 0352 2 req_desc [int_l_freevm] = lib$free_vm; ! Set address of free vm routine
155 0353 2
156 0354 2 ! Set entity name
157 0355 2
158 0356 2 str$analyze_sdesc (.name, req_desc [int_w_entlen], req_desc [int_l_entaddr]);
159 0357 2 RETURN (SYS$CLI (req_desc, rpw, req_flags)); ! Call callback utility
160 0358 1 END;
```

```
.TITLE CLISINTERFACE
.IDENT \V04-000\
```

```
.EXTRN SYS$CLI, LIB$GET_VM
.EXTRN LIB$FREE_VM, STR$ANALYZE_SDESC
.EXTRN STR$COPY_DX, CLIS_NOCOMD
.EXTRN CLIS_CONCAT, CLIS_COMMA
.EXTRN CLIS_PRESENT, CLIS_NEGATED
.EXTRN CLIS_LOCPRES, CLIS_LOCNEG
.EXTRN CLIS_DEFAULTED, CLIS_ABSENT
```

```
.PSECT $CODE$,NOWRT,2
```

```
.ENTRY CLISPRESENT, Save R2,R3,R4,R5
MOVAB -160(SP), SP
MOVCS #0, (SP), #0, #28, REQ_DESC

MOVB #80, REQ_DESC
MOVAB LIB$GET_VM, REQ_DESC+16
MOVAB LIB$FREE_VM, REQ_DESC+20
PUSHAB REQ_DESC+12
PUSHAB REQ_DESC+8
PUSHL NAME
```

1C

00

```
003C 00000
5E FF60 CE 9E 00002
6E 00 2C 00007
E4 AD 00000
F4 AD 00000000G 00 9E 00013
F8 AD 00000000G 00 9E 0001B
F0 AD 9F 00023
EC AD 9F 00026
04 AC DD 00029
```

```
: 0327
: 0349
: 0350
: 0351
: 0352
: 0356
:
```


E 6
16-Sep-1984 00:32:01
14-Sep-1984 12:14:58

Page 5
(3)

	03	FB	0002C
	5E	DD	00033
08	AE	9F	00035
E4	AD	9F	00038
	03	FB	0003B
		04	00042

```
CALLS    #3, STR$ANALYZE_SDESC
PUSHL    SP
PUSHAB   RPW
PUSHAB   REQ_DESC
CALLS    #3, SYS$CLI
RET
```

0357
0358

```
; Routine Size: 67 bytes,    Routine Base: $CODE$ + 0000
```

```
162 0359 1 GLOBAL ROUTINE cli$get_value (name, retdesc, retlength) =
163 0360 1
164 0361 1 ---
165 0362 1
166 0363 1 This routine is called to obtain the next value
167 0364 1 associated with a named entity on the command line.
168 0365 1
169 0366 1 Inputs:
170 0367 1
171 0368 1 name = Address of entity name descriptor
172 0369 1 retdesc = Address of a return buffer descriptor
173 0370 1 retlength = Address of a longword to return the value
174 0371 1 length in
175 0372 1
176 0373 1 Outputs:
177 0374 1
178 0375 1 True = A value was returned (there may be more).
179 0376 1 False = There is no more values associated with entity.
180 0377 1
181 0378 1 All other errors are signaled.
182 0379 1 ---
183 0380 1
184 0381 2 BEGIN
185 0382 2
186 0383 2 MAP
187 0384 2 retdesc : REF BBLOCK;
188 0385 2
189 0386 2 BUILTIN
190 0387 2 NULLPARAMETER;
191 0388 2
192 0389 2 LOCAL
193 0390 2 req_desc : BBLOCK [cli$creqdesc], ! Request descriptor block
194 0391 2 rpw : BBLOCK [cli$workarea], ! Result parse work area
195 0392 2 req_flags : BITVECTOR [32], ! Request flags array
196 0393 2 status;
197 0394 2
198 0395 2 CH$FILL(0, cli$creqdesc, req_desc); ! Zero request descriptor block
199 0396 2 req_desc [int_b_type] = cli$creqdesc; ! Set request type
200 0397 2 req_desc [int_l_getvm] = lib$get_vm; ! Set address of get vm routine
201 0398 2 req_desc [int_l_freevm] = lib$free_vm; ! Set address of free vm routine
202 0399 2
203 0400 2
204 0401 2 Set entity name
205 0402 2
206 0403 2 str$analyze_sdesc (.name, req_desc [int_w_entlen], req_desc [int_l_entaddr]);
207 0404 2 status = sys$cli (req_desc, rpw, req_flags); ! Call DCL utility
208 0405 2
209 0406 2
210 0407 2 If return length was requested, then find it now.
211 0408 2
212 0409 2 IF NOT NULLPARAMETER (3) ! If return length requested
213 0410 2 THEN (.retlength) <0,16> = ! then original estimate
214 0411 2 .req_desc [int_w_retlen];
215 0412 2
216 0413 2 str$copy_dx (.retdesc, req_desc [int_w_retlen]); ! Copy result into return desc
217 0414 2
218 0415 2 IF NOT NULLPARAMETER (3) ! If return length requested
```



```
: 219      0416      3      THEN BEGIN      ! Then return correct value
: 220      0417      3      LOCAL temp : BBLOCK [dsc$s_s_bln];      !
: 221      0418      3      str$analyze_sdesc (.retdesc,      ! Get latest estimate
: 222      0419      3      temp [dsc$w_length], temp [dsc$a_pointer]);
: 223      0420      3      IF .temp LSSU .retdesc [dsc$w_length]
: 224      0421      3      THEN (.retlength) <0,16> = .temp;      ! Return the smaller estimate
: 225      0422      3      END;
: 226      0423      3
: 227      0424      2      RETURN .status;
: 228      0425      1      END;
```

1C	00	56	00000000G	00	9E	00002	.ENTRY	CLISGET VALUE, Save R2,R3,R4,R5,R6	0359
		5E	FF58	CE	9E	00009	MOVAB	STR\$ANALYZE_SDESC, R6	
		6E		00	2C	0000E	MOVAB	-168(SP), SP	0395
			E4	AD		00013	MOVC5	#0, (SP), #0, #28, REQ_DESC	
		E4	AD	51	8F	90	MOVB	#81, REQ_DESC	0396
		F4	AD	00000000G	00	9E	MOVAB	LIB\$GET VM, REQ_DESC+16	0397
		F8	AD	00000000G	00	9E	MOVAB	LIB\$FREE VM, REQ_DESC+20	0398
			F0	AD	9F	0002A	PUSHAB	REQ_DESC+12	0403
			EC	AD	9F	0002D	PUSHAB	REQ_DESC+8	
			04	AC	DD	00030	PUSHL	NAME	
		66		03	FB	00033	CALLS	#3, STR\$ANALYZE_SDESC	
				5E	DD	00036	PUSHL	SP	0404
			10	AE	9F	00038	PUSHAB	RPW	
			E4	AD	9F	0003B	PUSHAB	REQ_DESC	
	00000000G	00		03	FB	0003E	CALLS	#3, SYSSCLI	
		52		50	D0	00045	MOVL	R0, STATUS	
		03		6C	91	00048	CMPB	(AP), #3	0409
				0A	1F	0004B	BLSSU	1\$	
			0C	AC	D5	0004D	TSTL	12(AP)	
				05	13	00050	BEQL	1\$	
	0C	BC	EC	AD	B0	00052	MOVW	REQ_DESC+8, @RETLENGTH	0411
			EC	AD	9F	00057	PUSHAB	REQ_DESC+8	0413
			08	AC	DD	0005A	PUSHL	RETDESC	
	00000000G	00		02	FB	0005D	CALLS	#2, STR\$COPY_DX	
		03		6C	91	00064	CMPB	(AP), #3	0415
				1F	1F	00067	BLSSU	2\$	
			0C	AC	D5	00069	TSTL	12(AP)	
				1A	13	0006C	BEQL	2\$	
			08	AE	9F	0006E	PUSHAB	TEMP+4	0419
			08	AE	9F	00071	PUSHAB	TEMP	
			08	AC	DD	00074	PUSHL	RETDESC	
		66		03	FB	00077	CALLS	#3, STR\$ANALYZE_SDESC	
04	AE	08	BC	00	ED	0007A	CMPZV	#0, #16, @RETDESC, TEMP	0420
		10		05	1B	00081	BLEQU	2\$	
			0C	BC	04	AE	MOVW	TEMP, @RETLENGTH	0421
				50	D0	00088	MOVL	STATUS, R0	0424
				52	D0	00088	RET		0425
				04	00	0008B			

; Routine Size: 140 bytes, Routine Base: \$CODE\$ + 0043

```
230 0426 1 GLOBAL ROUTINE cli$dcl_parse (command, tables, promptrtn, continrtn, prompt) =
231 0427 1
232 0428 1 ---
233 0429 1
234 0430 1 This routine can be called to parse a command line.
235 0431 1
236 0432 1 Inputs:
237 0433 1
238 0434 1 command = address of descriptor of command string
239 0435 1 tables = address of command tables
240 0436 1 promptrtn = address of user missing parameter prompt routine
241 0437 1 continrtn = address of user line continuation routine
242 0438 1 prompt = address of user prompt string descriptor
243 0439 1
244 0440 1 Outputs:
245 0441 1
246 0442 1 The command is parsed and the command work area is initialized.
247 0443 1
248 0444 1 ---
249 0445 1
250 0446 2 BEGIN
251 0447 2
252 0448 2 BUILTIN
253 0449 2 NULLPARAMETER;
254 0450 2
255 0451 2 LITERAL
256 0452 2 elements = 4;
257 0453 2
258 0454 2 LOCAL
259 0455 2 command_desc : BBLOCK [dsc$s_bln], ! Local command descriptor
260 0456 2 rtnlist : BBLOCK [4*(elements+1)], ! Routine list
261 0457 2 req_desc : BBLOCK [cli$c_reqdesc], ! Request descriptor block
262 0458 2 rpw : BBLOCK [cli$c_workarea], ! Result parse work area
263 0459 2 req_flags : BITVECTOR [32]; ! Request flags array
264 0460 2
265 0461 2 CH$FILL(0, cli$c_reqdesc, req_desc); ! Zero request descriptor block
266 0462 2 req_desc [int_b_type] = cli$k_dclparse; ! Set request type
267 0463 2 req_desc [int_l_getvm] = lib$get_vm; ! Set address of get vm routine
268 0464 2 req_desc [int_l_freevm] = lib$free_vm; ! Set address of free vm routine
269 0465 2
270 0466 2 IF NOT NULLPARAMETER(1) ! Set command line descriptor
271 0467 2 THEN BEGIN
272 0468 2 str$analyze_sdesc (.command, command_desc [dsc$w_length],
273 0469 2 command_desc [dsc$a_pointer]);
274 0470 2 req_desc [int_l_entaddr] = command_desc;
275 0471 2 END;
276 0472 2
277 0473 2 req_desc [int_l_tables] = .tables; ! Set address of command tables
278 0474 2
279 0475 2 IF NOT NULLPARAMETER(3) OR NOT NULLPARAMETER(4) ! If prompt or continue routine
280 0476 2 THEN BEGIN
281 0477 2 CH$FILL(0, 4*(elements+1), rtnlist); ! Zero the list
282 0478 2 rtnlist [int_l_listlen] = elements; ! Set number of elements in list
283 0479 2 req_desc [int_l_list] = rtnlist; ! And connect it
284 0480 2 END;
285 0481 2
286 0482 2 IF NOT NULLPARAMETER(3)
```



```
: 287      0483 2 THEN rtnlist [int_l_prompt] = .prompt; ! Set address of prompt routine
: 288      0484 2 IF NOT NULLPARAMETER(4)
: 289      0485 2 THEN rtnlist [int_l_continue] = .continue; ! Set address of continue routine
: 290      0486 2 IF NOT NULLPARAMETER(5)
: 291      0487 2 THEN str$analyze_sdesc (.prompt, rtnlist [int_w_prompt],
: 292      0488 2 rtnlist [int_l_prompt]);
: 293      0489 2
: 294      0490 2 RETURN (sys$cli (req_desc, rpw, req_flags)); ! Call DCL utility
: 295      0491 1 END;
```

1C	00	56	00000000G	00	9E	00002	.ENTRY	CLISDCL PARSE, Save R2,R3,R4,R5,R6	0426
		5E	FF44	CE	9E	00009	MOVAB	STR\$ANALYZE_SDESC, R6	
		6E		00	2C	0000E	MOVAB	-188(SP), SP	0461
			C8	AD		00013	MOVCS	#0, (SP), #0, #28, REQ_DESC	
		C8	AD	8F	90	00015	MOVB	#83, REQ_DESC	0462
		DB	AD	00	9E	0001A	MOVAB	LIB\$GET_VM, REQ_DESC+16	0463
		DC	AD	00	9E	00022	MOVAB	LIB\$FREE_VM, REQ_DESC+20	0464
				6C	95	0002A	TSTB	(AP)	0466
				16	13	0002C	BEQL	1\$	
			04	AC	D5	0002E	TSTL	4(AP)	
				11	13	00031	BEQL	1\$	
			FC	AD	9F	00033	PUSHAB	COMMAND_DESC+4	0469
			F8	AD	9F	00036	PUSHAB	COMMAND_DESC	0468
			04	AC	DD	00039	PUSHL	COMMAND	
				03	FB	0003C	CALLS	#3, STR\$ANALYZE_SDESC	
		D4	AD	F8	AD	0003F	MOVAB	COMMAND_DESC, REQ_DESC+12	0470
		CC	AD	08	AC	D0	MOVL	TABLES, REQ_DESC+4	0473
			03	6C	91	00049	CMPB	(AP), #3	0475
				05	1F	0004C	BLSSU	2\$	
			0C	AC	D5	0004E	TSTL	12(AP)	
				0A	12	00051	BNEQ	3\$	
			04	6C	91	00053	CMPB	(AP), #4	
				15	1F	00056	BLSSU	4\$	
			10	AC	D5	00058	TSTL	16(AP)	
				10	13	0005B	BEQL	4\$	
14	00		6E	00	2C	0005D	MOVCS	#0, (SP), #0, #20, RTNLIST	0477
				E4	AD	00062			
		E4	AD	04	D0	00064	MOVL	#4, RTNLIST	0478
		E0	AD	E4	AD	00068	MOVAB	RTNLIST, REQ_DESC+24	0479
			03	6C	91	0006D	CMPB	(AP), #3	0482
				0A	1F	00070	BLSSU	5\$	
			0C	AC	D5	00072	TSTL	12(AP)	
				05	13	00075	BEQL	5\$	
		E8	AD	0C	AC	D0	MOVL	PROMPTRN, RTNLIST+4	0483
			04	6C	91	00077	CMPB	(AP), #4	0484
				0A	1F	0007F	BLSSU	6\$	
				10	AC	D5	TSTL	16(AP)	
				05	13	00084	BEQL	6\$	
		EC	AD	10	AC	D0	MOVL	CONTINRTN, RTNLIST+8	0485
			05	6C	91	0008B	CMPB	(AP), #5	0486
				11	1F	0008E	BLSSU	7\$	
				14	AC	D5	TSTL	20(AP)	

CLISINTERFACE
V04-000

J 6
16-Sep-1984 00:32:01
14-Sep-1984 12:14:58

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

Page 10
(5)

			OC	13	00093	BEQL	7\$	:	
	F4		AD	9F	00095	PUSHAB	RTNLIST+16	:	0488
	F0		AD	9F	00098	PUSHAB	RTNLIST+12	:	0487
	14		AC	DD	0009B	PUSHL	PROMPT	:	
66			03	FB	0009E	CALLS	#3, STR\$ANALYZE_SDESC	:	
			5E	DD	000A1	PUSHL	SP	:	0490
	08		AE	9F	000A3	PUSHAB	RPW	:	
	C8		AD	9F	000A6	PUSHAB	REQ_DESC	:	
00000000G	00		03	FB	000A9	CALLS	#3, -SYSSCLI	:	
			04	00	00B0	RET		:	0491

; Routine Size: 177 bytes, Routine Base: \$CODE\$ + 00CF


```
297 0492 1 GLOBAL ROUTINE cli$dispatch (argument) =
298 0493 1
299 0494 1 ---
300 0495 1
301 0496 1 This routine can be called to dispatch to any verb processing
302 0497 1 routines if the command has the ROUTINE attribute.
303 0498 1
304 0499 1 Inputs:
305 0500 1
306 0501 1 argument = address of user supplied argument to his own routine
307 0502 1
308 0503 1 Outputs:
309 0504 1
310 0505 1 The verb routine is called (if any).
311 0506 1
312 0507 1 The status passed back from the routine is returned in R0.
313 0508 1 If no routine is specified, success is returned.
314 0509 1 ---
315 0510 1
316 0511 2 BEGIN
317 0512 2
318 0513 2 BUILTIN
319 0514 2 NULLPARAMETER;
320 0515 2
321 0516 2 LOCAL
322 0517 2 req_desc : BBLOCK [cli$creqdesc], ! Request descriptor block
323 0518 2 rpw : BBLOCK [cli$workarea], ! Result parse work area
324 0519 2 req_flags : BITVECTOR [32]; ! Request flags array
325 0520 2
326 0521 2 CH$FILL(0, cli$creqdesc, req_desc); ! Zero request descriptor block
327 0522 2 req_desc [int_b_type] = cli$dispatch; ! Set request type
328 0523 2 req_desc [int_l_getvm] = lib$get_vm; ! Set address of get vm routine
329 0524 2 req_desc [int_l_freevm] = lib$free_vm; ! Set address of free vm routine
330 0525 2 IF NOT NULLPARAMETER (1)
331 0526 2 THEN req_desc [int_l_entaddr] = .argument; ! Set address of user argument
332 0527 2 RETURN (SYS$CLI (req_desc, rpw, req_flags)); ! Call callback utility
333 0528 1 END;
```

1C	00	5E	FF60	CE	9E	00000	.ENTRY	CLISDISPATCH, Save R2,R3,R4,R5	0492
		6E		00	2C	00007	MOVAB	-160(SP), SP	0521
			E4	AD		0000C	MOVCS	#0, (SP), #0, #28, REQ_DESC	
	E4	AD	54	8F	90	0000E	MOVB	#84, REQ_DESC	0522
	F4	AD	00000000G	00	9E	00013	MOVAB	LIB\$GET_VM, REQ_DESC+16	0523
	F8	AD	00000000G	00	9E	0001B	MOVAB	LIB\$FREE_VM, REQ_DESC+20	0524
				6C	95	00023	TSTB	(AP)	0525
				0A	13	00025	BEQL	1\$	
			04	AC	D5	00027	TSTL	4(AP)	
				05	13	0002A	BEQL	1\$	
	F0	AD	04	AC	D0	0002C	MOVL	ARGUMENT, REQ_DESC+12	0526
				5E	DD	00031	PUSHL	SP	0527
			08	AE	9F	00033	PUSHAB	RPW	
			E4	AD	9F	00036	PUSHAB	REQ_DESC	

CLISINTERFACE
V04-000

L 6
16-Sep-1984 00:32:01
14-Sep-1984 12:14:58

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

Page 12
(6)

00000000G 00

03 FB 00039
04 00040

CALLS #3, SYSSCLI
RET

: 0528

; Routine Size: 65 bytes, Routine Base: \$CODE\$ + 0180


```
335 0529 1 GLOBAL ROUTINE cli$next_qual (name) =
336 0530 1
337 0531 1 ---
338 0532 1
339 0533 1      Move to the next command qualifier on the line.
340 0534 1
341 0535 1      Inputs:
342 0536 1
343 0537 1          name = Address of entity name descriptor
344 0538 1
345 0539 1      Outputs:
346 0540 1
347 0541 1          routine value = True if present, else false.
348 0542 1
349 0543 1 ---
350 0544 2 BEGIN
351 0545 2
352 0546 2 MAP
353 0547 2     name : REF BBLOCK;
354 0548 2
355 0549 2 LOCAL
356 0550 2     req_desc : BBLOCK [cli$c_reqdesc], ! Request descriptor block
357 0551 2     rpw : BBLOCK [cli$c_workarea], ! Result parse work area
358 0552 2     req_flags : BITVECTOR [32]; ! Request flags array
359 0553 2
360 0554 2 CH$FILL(0, cli$c_reqdesc, req_desc); ! Zero request descriptor block
361 0555 2 req_desc [int_b_type] = cli$k_nextqual; ! Set request type
362 0556 2 req_desc [int_l_getvm] = lib$get_vm; ! Set address of get vm routine
363 0557 2 req_desc [int_l_freevm] = lib$free_vm; ! Set address of free vm routine
364 0558 2
365 0559 2 ! Set entity name
366 0560 2
367 0561 2 str$analyze_sdesc (.name, req_desc [int_w_entlen], req_desc [int_l_entaddr]);
368 0562 2 RETURN (sys$cli (req_desc, rpw, req_flags)); ! Call DCL utility
369 0563 1 END;
```

1C	00	5E	FF60	CE	9E	00002	.ENTRY	CLISNEXT_QUAL, Save R2,R3,R4,R5	0529
		6E		00	2C	00007	MOVAB	-160(SP), SP	
			E4	AD		0000C	MOVC5	#0, (SP), #0, #28, REQ_DESC	0554
		E4	AD	55	8F	90 0000E	MOVB	#85, REQ_DESC	0555
		F4	AD	00000000G	00	9E 00013	MOVAB	LIB\$GET_VM, REQ_DESC+16	0556
		F8	AD	00000000G	00	9E 0001B	MOVAB	LIB\$FREE_VM, REQ_DESC+20	0557
			F0	AD	9F	00023	PUSHAB	REQ_DESC+12	0561
			EC	AD	9F	00026	PUSHAB	REQ_DESC+8	
			04	AC	DD	00029	PUSHL	NAME	
	00000000G	00		03	FB	0002C	CALLS	#3, STR\$ANALYZE_SDESC	
				5E	DD	00033	PUSHL	SP	0562
			08	AE	9F	00035	PUSHAB	RPW	
			E4	AD	9F	00038	PUSHAB	REQ_DESC	
	00000000G	00		03	FB	0003B	CALLS	#3, -SYSSCLI	
				04	00	0042	RET		0563

CLISINTERFACE
V04-000

N 6
16-Sep-1984 00:32:01
14-Sep-1984 12:14:58

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[DCL.SRC]CLINT.B32;1 Page 14
(7)

; Routine Size: 67 bytes, Routine Base: \$CODE\$ + 01C1

CLISINTERFACE
V04-000

B 7
16-Sep-1984 00:32:01
14-Sep-1984 12:14:58

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[DCL.SRC]CLINT.B32;1 Page 15
(8)

: 371 0564 1 END
: 372 0565 0 ELUDOM

PSECT SUMMARY

:
: Name Bytes Attributes
: \$CODE\$ 516 NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)

Library Statistics

:
: File Total Symbols Loaded Percent Pages Mapped Processing Time
: _\$255\$DUA28:[SYSLIB]STARLET.L32;1 9776 15 0 581 00:01.0

COMMAND QUALIFIERS

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

: Size: 516 code + 0 data bytes
: Run Time: 00:10.2
: Elapsed Time: 00:40.1
: Lines/CPU Min: 3317
: Lexemes/CPU-Min: 20213
: Memory Used: 93 pages
: Compilation Complete

0069 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

INTDEF
SOL

COMMAND
LIS

DESCRVAL
LIS

CLIMAC
MAR

CLIMSG
LIS

CONNECT
LIS

DCLPARSE
LIS

CHARMANIP
LIS

EXAMDEP
LIS

EXIT
LIS

INTIMAGES
MAR

DCXSTART
LIS

CLIGBL
LIS

DISALLOW
LIS

CLINT
LIS

CANCEL
LIS