

VVV	VVV	MMM	MMM	SSSSSSSSSSSS	LLL	IIIIIIII	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSSSSSSSSSSS	LLL	IIIIIIII	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSSSSSSSSSSS	LLL	IIIIIIII	BBBBBBBBBBBB	
VVV	VVV	MMMMMM	MMMMMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMMMMM	MMMMMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMMMMM	MMMMMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSSSSSSSSS	LLL	III	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSSSSSSSSS	LLL	III	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSSSSSSSSS	LLL	III	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSS	LLL	III	BBB	BBB
VVV	VVV	MMM	MMM	SSSSSSSSSSSS	LLLLLLLLLLLLLLLL	IIIIIIII	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSSSSSSSSSSS	LLLLLLLLLLLLLLLL	IIIIIIII	BBBBBBBBBBBB	
VVV	VVV	MMM	MMM	SSSSSSSSSSSS	LLLLLLLLLLLLLLLL	IIIIIIII	BBBBBBBBBBBB	

```

LL      IIIIII  BBBB BBBB  EEEEEEEEEEE  XX      XX  EEEEEEEEEEE  CCCCCCCC  LL      IIIIII
LL      IIIIII  BBBB BBBB  EEEEEEEEEEE  XX      XX  EEEEEEEEEEE  CCCCCCCC  LL      IIIIII
LL      II      BB      BB  EE      EE  XX      XX  EEEEEEEEEEE  CC      LL      II
LL      II      BB      BB  EE      EE  XX      XX  EEEEEEEEEEE  CC      LL      II
LL      II      BB      BB  EE      EE  XX      XX  EEEEEEEEEEE  CC      LL      II
LL      II      BBBB BBBB  EEEEEEEEEEE  XX      XX  EEEEEEEEEEE  CC      LL      II
LL      II      BBBB BBBB  EEEEEEEEEEE  XX      XX  EEEEEEEEEEE  CC      LL      II
LL      II      BB      BB  EE      EE  XX      XX  EEEEEEEEEEE  CC      LL      II
LL      II      BB      BB  EE      EE  XX      XX  EEEEEEEEEEE  CC      LL      II
LL      II      BB      BB  EE      EE  XX      XX  EEEEEEEEEEE  CC      LL      II
LL      IIIIII  BBBB BBBB  EEEEEEEEEEE  XX      XX  EEEEEEEEEEE  CC      LL      IIIIII
LLLLLLLLLL  IIIIII  BBBB BBBB  EEEEEEEEEEE  XX      XX  EEEEEEEEEEE  CCCCCCCC  LLLLLLLLLL  IIIIII
LLLLLLLLLL  IIIIII  BBBB BBBB  EEEEEEEEEEE  XX      XX  EEEEEEEEEEE  CCCCCCCC  LLLLLLLLLL  IIIIII

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      IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS
```

```
0001 0 %TITLE 'LIB$EXECUTE_CLI - Execute CLI commands'
0002 0 MODULE LIB$EXECUTE_CLI ( ! Execute CLI commands
0003 0 IDENT = 'V04-000' ! File: LIB$EXECCLI.B32
0004 0 ) =
0005 1 BEGIN
0006 1
0007 1 *****
0008 1 *
0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0011 1 * ALL RIGHTS RESERVED.
0012 1 *
0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0018 1 * TRANSFERRED.
0019 1 *
0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0022 1 * CORPORATION.
0023 1 *
0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0026 1 *
0027 1 *
0028 1 *****
0029 1
0030 1
0031 1 ++
0032 1 FACILITY: General Utility Library
0033 1
0034 1 ABSTRACT:
0035 1
0036 1 A procedure to execute arbitrary CLI commands when called
0037 1 from a program.
0038 1
0039 1 ENVIRONMENT: Runs at any access mode - AST reentrant
0040 1
0041 1 AUTHOR: Len Kawell, CREATION DATE: 21-May-1979
0042 1
0043 1 MODIFIED BY:
0044 1
0045 1 3-005 MHB0078 Mark H. Bramhall 4-Nov-1981
0046 1 Remove the usage of ASTs so that the routine will correctly
0047 1 function with ASTs inhibited.
0048 1
0049 1 3-004 SBL0004 Steve Lionel 14-Sept-1981
0050 1 Make code agree with comments as to the restriction on the
0051 1 length of the command line. Comments said 256 but code
0052 1 said 132. Change code to read 256.
0053 1
0054 1 3-003 LMK0003 Len Kawell 10-Mar-1981
0055 1 Create subprocess with base priority of master process. Also
0056 1 change termination mailbox status check.
0057 1
```

:	58	0058	1	:	3-002	-	LMK0002	Len Kawell	18-January-1981
:	59	0059	1	:				Fix to only \$DELPRC if process was created successfully and correctly	
:	60	0060	1	:				return status to the caller.	
:	61	0061	1	:					
:	62	0062	1	:	3-001	-	SBL3001	Steven B. Lionel	16-January-1980
:	63	0063	1	:				Make code PIC. Make PSECT names conform to LIB\$ standards. Allow	
:	64	0064	1	:				arguments to be omitted by reference. Use authorized privs only	
:	65	0065	1	:				when creating subprocess. Send a final \$ to force EOF. Restrict	
:	66	0066	1	:				command lengths to 256 bytes.	
:	67	0067	1	:	--				
:	68	0068	1	:					

```
70 0069 1 %SBTTL 'Declarations'
71 0070 1
72 0071 1 SWITCHES:
73 0072 1
74 0073 1
75 0074 1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
76 0075 1
77 0076 1
78 0077 1 LINKAGES:
79 0078 1
80 0079 1 NONE
81 0080 1
82 0081 1 TABLE OF CONTENTS:
83 0082 1
84 0083 1
85 0084 1 FORWARD ROUTINE
86 0085 1 LIB$EXECUTE_CLI, ! Execute CLI commands
87 0086 1 EXIT_HANDLER : NOVALUE; ! Exit handler
88 0087 1
89 0088 1
90 0089 1 INCLUDE FILES:
91 0090 1
92 0091 1
93 0092 1 LIBRARY 'SYS$LIBRARY:STARLET'; ! System symbols
94 0093 1
95 0094 1
96 0095 1
97 0096 1 MACROS:
98 0097 1
99 0098 1
100 0099 1 MACRO
101 M 0100 1 CKSTATUS[] = ! Check routine status
102 M 0101 1 BEGIN
103 M 0102 1 LOCAL STATUS;
104 M 0103 1 STATUS = (%REMAINING);
105 M 0104 1 IF NOT .STATUS THEN LEAVE MAIN_BLOCK WITH .STATUS;
106 M 0105 1 END%;
107 0106 1
108 0107 1
109 0108 1 EQUATED SYMBOLS:
110 0109 1
111 0110 1
112 0111 1 LITERAL
113 0112 1 ESC = %X'1B', ! ASCII escape
114 0113 1 LOGNAM_SIZ = 64; ! Maximum logical name size
115 0114 1
116 0115 1
117 0116 1 FIELDS:
118 0117 1
119 0118 1 NONE
120 0119 1
121 0120 1 PSECTS:
122 0121 1
123 0122 1
124 0123 1 PSECT
125 0124 1 CODE = _LIB$CODE (READ, NOWRITE, EXECUTE, SHARE, PIC, ADDRESSING_MODE (WORD_RELATIVE)),
126 0125 1 PLIT = _LIB$CODE (READ, NOWRITE, EXECUTE, SHARE, PIC, ADDRESSING_MODE (WORD_RELATIVE));
```

```

: 127      0126 1
: 128      0127 1
: 129      0128 1   OWN STORAGE:
: 130      0129 1
: 131      0130 1       NONE
: 132      0131 1
: 133      0132 1   EXTERNAL REFERENCES:
: 134      0133 1
: 135      0134 1
: 136      0135 1   EXTERNAL ROUTINE
: 137      0136 1       LIB$GET_EF;
: 138      0137 1       LIB$FREE_EF;
: 139      0138 1
: 140      0139 1   EXTERNAL LITERAL
: 141      0140 1       LIB$_INSEF;
: 142      0141 1       LIB$_INVARG;
: 143      0142 1

```

! Allocates event flag number
! Deallocates event flag number
! Condition value symbols
! Insufficient event flags
! Invalid argument

```
145 0143 1 %SBTTL 'LIB$EXECUTE_CLI - Execute CLI commands'
146 0144 1 GLOBAL ROUTINE LIB$EXECUTE_CLI (
147 0145 1     COMMAND1
148 0146 1 ) =
149 0147 1
150 0148 1 ++
151 0149 1 FUNCTIONAL DESCRIPTION:
152 0150 1
153 0151 1     LIB$EXECUTE_CLI creates a subprocess to execute a CLI command
154 0152 1     or commands. To do this it first creates two mailboxes, one
155 0153 1     to be used as the subprocess's input device, and one to receive
156 0154 1     the subprocess's termination status. This routine then creates
157 0155 1     the subprocess running SYS$SYSTEM:LOGINOUT.EXE. This will
158 0156 1     initiate the appropriate command interpreter reading commands
159 0157 1     from the input mailbox.
160 0158 1
161 0159 1     When the subprocess terminates, the termination status will
162 0160 1     be placed in the termination mailbox and returned to the caller.
163 0161 1
164 0162 1
165 0163 1 CALLING SEQUENCE:
166 0164 1
167 0165 1     ret_status.wlc.v = LIB$EXECUTE_CLI (command1.rt.dx [, commandn.rt.dx...])
168 0166 1
169 0167 1 FORMAL PARAMETERS:
170 0168 1
171 0169 1     command1 - commandn    - Addresses of string descriptors containing
172 0170 1                           command line text.
173 0171 1
174 0172 1 IMPLICIT INPUTS:
175 0173 1
176 0174 1     NONE
177 0175 1
178 0176 1 IMPLICIT OUTPUTS:
179 0177 1
180 0178 1     NONE
181 0179 1
182 0180 1 COMPLETION STATUS:
183 0181 1
184 0182 1     SSS_NORMAL      Normal successful completion
185 0183 1     LIB$INVARG      Invalid argument
186 0184 1     LIB$INSEF       Insufficient event flags
187 0185 1     SSS_xyz         System service failure
188 0186 1
189 0187 1 SIDE EFFECTS:
190 0188 1
191 0189 1     NONE
192 0190 1
193 0191 1 --
194 0192 1
195 0193 2 BEGIN
196 0194 2
197 0195 2 BUILTIN
198 0196 2     ACTUALCOUNT,
199 0197 2     ACTUALPARAMETER;
200 0198 2 LOCAL
201 0199 2     OUTPUT_DESC      : VECTOR[2],
```

! Execute CLI commands
! One or more command strings

! Count of parameters
! Parameter address
! Output logical name descriptor

```
202      0200      2      OUTPUTBUF      : VECTOR[LOGNAM_SIZ, BYTE], ! Output logical name buffer
203      0201      2      OUTPUTBUF_DESC : VECTOR[2], ! Output logical name buffer desc
204      0202      2      EFN_MASK      : VECTOR[1], ! Command/Termination EFN mask
205      0203      2      CMDMBXCHAN    : VECTOR[1, WORD], ! Command mailbox channel
206      0204      2      CMDMBXNAMBUF  : VECTOR[LOGNAM_SIZ, BYTE], ! Command mailbox name buffer
207      0205      2      CMDMBXNAM_DESC : VECTOR[2], ! Command mailbox name desc
208      0206      2      CMDMBXEFN     : VECTOR[1], ! Command mailbox event flag number
209      0207      2      CMDMBXIOSB    : VECTOR[2], ! Termination mailbox IOSB
210      0208      2      TRMMBXCHAN    : VECTOR[1, WORD], ! Termination mailbox channel
211      0209      2      TRMMBXEFN     : VECTOR[1], ! Termination mailbox event flag number
212      0210      2      TRMMBXIOSB    : VECTOR[2], ! Termination mailbox IOSB
213      0211      2      TRMM$GBUF     : BLOCK[ACC$K_TERMLEN, BYTE], ! Termination message buffer
214      0212      2      CHARBUF       : BLOCK[DIB$K_LENGTH+16, BYTE], ! Characteristics buffer
215      0213      2      CHARBUF_DESC  : VECTOR[2], ! Characteristics buffer desc
216      0214      2      FAOSTR_DESC   : VECTOR[2], ! FAO control string descriptor
217      0215      2      IMAGE_DESC    : VECTOR[2], ! Descriptor for image name
218      0216      2      PRVMSK       : VECTOR[2], ! Permanent process privileges
219      0217      2      GETPRIB       : VECTOR[4], ! Get process priority list
220      0218      2      PRIB          : LONG, ! Process priority
221      0219      2      EXITSTATUS    : VECTOR[1], ! Image exit status
222      0220      2      EXITBLOCK     : VECTOR[5]; ! Exit control block
223      0221      2
224      0222      2      REGISTER
225      0223      2      STATUS        : ! Routine call status
226      0224      2
227      0225      2      BIND
228      0226      2      SUBPID        = EXITBLOCK[4]; ! Subprocess ID
229      0227      2
230      0228      2      LABEL
231      0229      2      MAIN_BLOCK; ! Name of main block
232      0230      2
233      0231      2      !
234      0232      2      ! Zero mailbox channels and event flag numbers, and the subprocess
235      0233      2      ! ID so we can unconditionally deassign, deallocate, and delete them
236      0234      2      ! in case of a failure in the middle of execution.
237      0235      2
238      0236      2      TRMMBXCHAN = CMDMBXCHAN = TRMMBXEFN = CMDMBXEFN = SUBPID = 0;
239      0237      2
240      0238      2      ! Main block of code in routine
241      0239      2
242      0240      2      STATUS =
243      0241      2      MAIN_BLOCK:
244      0242      2      BEGIN
245      0243      2
246      0244      2      !
247      0245      2      ! Get current process's base priority.
248      0246      2
249      0247      2      GETPRIB[0] = 4 OR (JPI$PRIB ^ 16);
250      0248      2      GETPRIB[1] = PRIB;
251      0249      2      GETPRIB[2] = GETPRIB[3] = 0;
252      0250      2      CKSTATUS($GETJPI(ITMLST=GETPRIB));
253      0251      2
254      0252      2      ! Get translation of current SYS$OUTPUT logical name
255      0253      2
256      0254      2      OUTPUTBUF_DESC[0] = LOGNAM_SIZ; ! Initialize output
257      0255      2      OUTPUTBUF_DESC[1] = OUTPUTBUF; ! logical name buffer desc
258      0256      2      OUTPUT_DESC [0] = %CHARCOUNT ('SYS$OUTPUT'); ! Name to translate
```

```
259      0257 3      OUTPUT_DESC [1] = UPLIT BYTE ('SYSS$OUTPUT');
260      0258 3      DO
261      0259 4          BEGIN
262      P 0260 4          STATUS = $TRNLOG(LOGNAM=OUTPUT_DESC, ! Translate name
263      P 0261 4          RSLLEN=OUTPUT_DESC[0],
264      0262 4          RSLBUF=OUTPUTBUF_DESC);
265      0263 4          OUTPUT_DESC[1] = OUTPUTBUF; ! Set pointer to translation
266      0264 4          IF (.OUTPUTBUF[0] EQLU ESC) AND ! If name is RMS process
267      0265 5          (.OUTPUTBUF[1] EQLU 0) ! permanent file name,
268      0266 4          THEN ! strip escape and IFI
269      0267 5          BEGIN
270      0268 5          OUTPUT_DESC[0] = .OUTPUT_DESC[0] - 4;
271      0269 5          OUTPUT_DESC[1] = .OUTPUT_DESC[1] + 4;
272      0270 4          END;
273      0271 4          END
274      0272 3      UNTIL NOT .STATUS OR (.STATUS EQLU SSS$NOTRAN);
275      0273 3      IF .STATUS NEQU SSS$NOTRAN THEN LEAVE MAIN_BLOCK WITH .STATUS;
276      0274 3      ! Create the command mailbox and get its name
277      0275 3      CKSTATUS($CREMBX(CHAN=CMDMBXCHAN,
278      P 0276 3      MAXMSG=256)); ! Create mailbox
279      0277 3      CHARBUF_DESC[0] = DIB$K LENGTH + 16; ! Initialize characteristics
280      0278 3      CHARBUF_DESC[1] = CHARBUF; ! buffer descriptor
281      0279 3      CKSTATUS($GETCHN(CHAN=CMDMBXCHAN, ! Get mailbox characteristics
282      P 0280 3      PRIBUF=CHARBUF_DESC));
283      0281 3      CMDMBXNAM_DESC[0] = LOGNAM_SIZ; ! Initialize mailbox name
284      0282 3      CMDMBXNAM_DESC[1] = CMDMBXNAMBUF; ! buffer descriptor
285      0283 3      FAOSTR_DESC[0] = %CHARCOUNT ('!AC!UW:'); ! Device name format
286      0284 3      FAOSTR_DESC[1] = UPLIT BYTE ('!AC!UW:');
287      0285 3      CKSTATUS($FAO( ! Format mailbox name
288      P 0286 3      FAOSTR_DESC, ! Control string descriptor
289      P 0287 3      CMDMBXNAM_DESC, ! output length address
290      P 0288 3      CMDMBXNAM_DESC, ! output buffer desc address
291      P 0289 3      (.CHARBUF[DIB$W_DEVNAMOFF] + CHARBUF),
292      P 0290 3      .CHARBUF[DIB$W_UNIT]));
293      0291 3      ! Create the termination mailbox and get its number
294      0292 3      CKSTATUS($CREMBX(CHAN=TRMMBXCHAN)); ! Create the mailbox
295      0293 3      CKSTATUS($GETCHN(CHAN=TRMMBXCHAN, ! Get its characteristics
296      P 0294 3      PRIBUF=CHARBUF_DESC));
297      0295 3      ! Declare an exit handler to delete the subprocess if we (the
298      0296 3      creator) are rundown.
299      0297 3      EXITBLOCK[1] = EXIT_HANDLER; ! Set exit handler address
300      0298 3      EXITBLOCK[2] = 2; ! Set argument count
301      0299 3      EXITBLOCK[3] = EXITSTATUS; ! Set address to store
302      0300 3      ! exit status
303      0301 3      CKSTATUS($DCLEXH(DESBLOCK=EXITBLOCK)); ! Declare exit handler
304      0302 3      ! Create the subprocess running the LOGIN image. LOGIN will
305      0303 3      initiate a command interpreter to read and execute the commands
306      0304 3      we'll write in the command mailbox.
307      0305 3      CKSTATUS($SETPRV(PRVPRV=PRVMSK, ! Get process's permanent
308      0306 3
309      0307 3
310      0308 3
311      0309 3
312      0310 3
313      0311 3
314      0312 3
315      P 0313 3
```

```
PRMFLG=1)); ! privileges; don't use image's
IMAGE_DESC[0] = %CHARCOUNT ('SYSS$SYSTEM:LOGINOUT'); ! Image to run
IMAGE_DESC[1] = UPLIT BYTE ('SYSS$SYSTEM:LOGINOUT');
CKSTATUS($CREPRC (IMAGE=IMAGE_DESC,
                  INPUT=CMDMBXNAM_DESC,
                  OUTPUT=OUTPUT_DESC,
                  ERROR=OUTPUT_DESC,
                  MBXUNT=.CHARBUF[DIB$W_UNIT],
                  PRCNAM=CMDMBXNAM_DESC,
                  PRVADR=PRVMSK,
                  BASPRI=.PRIB,
                  PIDADR=SUBPID));

! Allocate the termination EFN and the command EFN. We need the
! EFNs to be in the same 32 bit group so that we can form an EFN
! mask for $WFLOR.
CKSTATUS(LIB$GET_EF(TRMMBXEFN)); ! Allocate termination event flag
CKSTATUS(LIB$GET_EF(CMDMBXEFN)); ! Allocate command event flag
IF (.TRMMBXEFN / 32) NEQ (.CMDMBXEFN / 32)
THEN
    BEGIN ! A 3rd GET_EF should find same group
    LOCAL
        TMPMBXEFN : VECTOR[1];
    CKSTATUS(LIB$GET_EF(TMPMBXEFN));
    LIB$FREE_EF(TRMMBXEFN);
    TRMMBXEFN = TMPMBXEFN;
    IF (.TRMMBXEFN / 32) NEQ (.CMDMBXEFN / 32)
    THEN
        LEAVE MAIN_BLOCK WITH LIB$_INSEF;
    END;
EFN_MASK = (1 ^ (.TRMMBXEFN AND (32 - 1))) OR
            (1 ^ (.CMDMBXEFN AND (32 - 1)));

! Issue a QIO request to read the termination mailbox specifying
! the termination EFN. If the EFN gets set, then subprocess died
! and we have to give up.
CKSTATUS( $QIO(CHAN=.TRMMBXCHAN, ! Issue read QIO request
               EFN=.TRMMBXEFN,
               FUNC=IOS$ READLBLK,
               IOSB=TRMMBXIOSB,
               P1=TRMM$GBUF,
               P2=ACC$K_TERMLEN));

! Write the specified command strings to the command mailbox
! so the subprocess can execute them.
INCR PARAM FROM 1 TO ACTUALCOUNT()
DO
    IF ACTUALPARAMETER(.PARAM) NEQ 0
    THEN
        BEGIN
            ! Check command string for invalid length.
```

```
373      0371  4      IF .(ACTUALPARAMETER(.PARAM))<0,16> GTRU 256
374      0372  4      THEN
375      0373  4          LEAVE MAIN_BLOCK WITH LIB$_INVARG;
376      0374  4
377      P 0375  4      CKSTATUS($QIO( CHAN=.CMDMBXCHAN, ! Write command to mailbox
378      P 0376  4          EFN=.CMDMBXEFN,
379      P 0377  4          FUNC=IOS$ WRITELBLK,
380      P 0378  4          IOSB=CMDMBXIOSB,
381      P 0379  4          P1=.(ACTUALPARAMETER(.PARAM) + 4),
382      0380  4          P2=.ACTUALPARAMETER(.PARAM));
383      0381  4      CKSTATUS($WFLOR(EFN=.TRMMBXEFN,MASK=.EFN_MASK));
384      0382  4      STATUS = .CMDMBXIOSB<0,16>; ! Get completion status
385      0383  4      IF NOT .STATUS THEN LEAVE MAIN_BLOCK WITH .STATUS;
386      0384  4      END;
387      0385  4
388      0386  4
389      0387  4      + Write a final $ record to the subprocess. This will insure that
390      0388  4      the previous command didn't leave the CLI waiting for more input.
391      0389  4      -
392      0390  4
393      P 0391  4      CKSTATUS($QIO( CHAN=.CMDMBXCHAN, ! Write to mailbox
394      P 0392  4          EFN=.CMDMBXEFN,
395      P 0393  4          FUNC=IOS$ WRITELBLK,
396      P 0394  4          IOSB=CMDMBXIOSB,
397      P 0395  4          P1=UPLIT BYTE ('$'),
398      0396  4          P2=%CHARCOUNT ('$'));
399      0397  4      CKSTATUS($WFLOR(EFN=.TRMMBXEFN,MASK=.EFN_MASK));
400      0398  4      STATUS = .CMDMBXIOSB<0,16>; ! Get completion status
401      0399  4      IF NOT .STATUS THEN LEAVE MAIN_BLOCK WITH .STATUS;
402      0400  4
403      0401  4      !
404      0402  4      ! Write an End-of-File marker to command mailbox.
405      0403  4
406      P 0404  4      CKSTATUS($QIO( CHAN=.CMDMBXCHAN, ! Write EOF
407      P 0405  4          EFN=.CMDMBXEFN,
408      P 0406  4          FUNC=IOS$ WRITEOF,
409      0407  4          IOSB=CMDMBXIOSB));
410      0408  4      CKSTATUS($WFLOR(EFN=.TRMMBXEFN,MASK=.EFN_MASK));
411      0409  4      STATUS = .CMDMBXIOSB<0,16>;
412      0410  4      IF NOT .STATUS THEN LEAVE MAIN_BLOCK WITH .STATUS;
413      0411  4
414      0412  4      STATUS = $WAITFR(EFN=.TRMMBXEFN) ! Wait for subprocess
415      0413  4      ! to complete
416      0414  2      END;
417      0415  2
418      0416  2      ! Cleanup everything that we used...
419      0417  2
420      0418  2      $DASSGN(CHAN=.CMDMBXCHAN); ! Deassign the mailbox
421      0419  2      $DASSGN(CHAN=.TRMMBXCHAN); ! channels
422      0420  2      IF .SUBPID NEQ 0
423      0421  2      THEN
424      0422  2          $DELPRC(PIDADR=SUBPID); ! Make sure subprocess
425      0423  2          ! is stopped
426      0424  2      LIB$FREE_EF(CMDMBXEFN); ! Deallocate event flag
427      0425  2      LIB$FREE_EF(TRMMBXEFN); ! numbers
428      0426  2      $CANEXH(DESBLK=EXITBLOCK); ! Cancel exit handler
429      0427  2      IF .TRMMBXIOSB ! If termination message,
```

Page 10
(3)

```
; ! Get subprocess status
! Return the status

! Of routine LIB$EXECUTE_CLI
```

[illegible]

		7E	D4	00069	CLRL	-(SP)	
		B0	AD	9F	PUSHAB	OUTPUTBUF_DESC	
		F8	AD	9F	PUSHAB	OUTPUT_DESC	
		F8	AD	9F	PUSHAB	OUTPUT_DESC	
00000000G	00		06	FB	CALLS	#6, SYSTRNLOG	
	52		50	D0	MOVL	R0, STATUS	
FC	AD	B8	AD	9E	MOVAB	OUTPUTBUF, OUTPUT_DESC+4	0263
	1B	B8	AD	91	CMPB	OUTPUTBUF, #27	0264
			0D	12	BNEQ	2\$	
		B9	AD	95	TSTB	OUTPUTBUF+1	0265
			08	12	BNEQ	2\$	
F8	AD		04	C2	SUBL2	#4, OUTPUT_DESC	0268
FC	AD		04	C0	ADDL2	#4, OUTPUT_DESC+4	0269
	09		52	E9	BLBC	STATUS, 3\$	0272
00000629	8F		52	D1	CMPL	STATUS, #1577	
			C5	12	BNEQ	1\$	
00000629	8F		52	D1	CMPL	STATUS, #1577	0273
			03	13	BEQL	4\$	
		0247	31	000AB	BRW	20\$	
			7E	7C	CLRQ	-(SP)	0278
			7E	7C	CLRQ	-(SP)	
	7E	0100	8F	3C	MOVZWL	#256, -(SP)	
		18	AE	9F	PUSHAB	CMDMBXCHAN	
			7E	D4	CLRL	-(SP)	
00000000G	00		07	FB	CALLS	#7, SYSCREMBX	
	7E		50	E9	BLBC	STATUS, 6\$	
58	AE	84	8F	9A	MOVZBL	#132, CHARBUF_DESC	0279
5C	AE	60	AE	9E	MOVAB	CHARBUF, CHARBUF_DESC+4	0280
			7E	7C	CLRQ	-(SP)	0282
		60	AE	9F	PUSHAB	CHARBUF_DESC	
			7E	D4	CLRL	-(SP)	
00000000G	7E	14	AE	3C	MOVZWL	CMDMBXCHAN, -(SP)	
	00		05	FB	CALLS	#5, SYSGETCHN	
	7B		50	E9	BLBC	STATUS, 7\$	
FF68	CD	40	8F	9A	MOVZBL	#64, CMDMBXNAM_DESC	0283
FF6C	CD	FF70	CD	9E	MOVAB	CMDMBXNAMBUF, CMDMBXNAM_DESC+4	0284
50	AE		08	D0	MOVL	#8, FAOSTR_DESC	0285
54	AE	FEEA	CF	9E	MOVAB	P.AAB, FAOSTR_DESC+4	0286
	7E	6C	AE	3C	MOVZWL	CHARBUF+12, -(SP)	0292
	50	72	AE	3C	MOVZWL	CHARBUF+14, R0	
		64	AE40	9F	PUSHAB	CHARBUF[R0]	
		FF68	CD	9F	PUSHAB	CMDMBXNAM_DESC	
		FF68	CD	9F	PUSHAB	CMDMBXNAM_DESC	
		60	AE	9F	PUSHAB	FAOSTR_DESC	
00000000G	00		05	FB	CALLS	#5, SYSSFAO	
	54		50	E9	BLBC	STATUS, 8\$	
			7E	7C	CLRQ	-(SP)	0296
			7E	7C	CLRQ	-(SP)	
			7E	D4	CLRL	-(SP)	
		1C	AE	9F	PUSHAB	TRMMBXCHAN	
			7E	D4	CLRL	-(SP)	
00000000G	00		07	FB	CALLS	#7, SYSCREMBX	
	77		50	E9	BLBC	STATUS, 9\$	
			7E	7C	CLRQ	-(SP)	0298
		60	AE	9F	PUSHAB	CHARBUF_DESC	
			7E	D4	CLRL	-(SP)	
	7E	18	AE	3C	MOVZWL	TRMMBXCHAN, -(SP)	

LIB\$EXECUTE_CLI
V04-000

LIB\$EXECUTE_CLI - Execute CLI commands
LIB\$EXECUTE_CLI - Execute CLI commands

E 2
16-Sep-1984 02:25:23
14-Sep-1984 13:34:25

VAX-11 Bliss-32 V4.0-742
[VMSLIB.SRC]LIBEXECLI.B32;1

Page 12
(3)

00000000G	00	05	FB	0013D	CALLS	#5, SYSS\$GETCHN	
	74	50	E9	00144	BLBC	STATUS, 10\$	
20	AE	0000V	CF	9E 00147	MOVAB	EXIT_HANDLER, EXITBLOCK+4	0303
24	AE		02	D0 0014D	MOVL	#2, EXITBLOCK+8	0304
28	AE	0C	AE	9E 00151	MOVAB	EXITSTATUS, EXITBLOCK+12	0305
		1C	AE	9F 00156	PUSHAB	EXITBLOCK	0307
00000000G	00		01	FB 00159	CALLS	#1, SYSS\$DCLEXH	
	73		50	E9 00160	BLBC	STATUS, 11\$	
		40	AE	9F 00163	PUSHAB	PRVMSK	0314
			01	DD 00166	PUSHL	#1	
			7E	7C 00168	CLRQ	-(SP)	
00000000G	00		04	FB 0016A	CALLS	#4, SYSS\$SETPRV	
	62		50	E9 00171	BLBC	STATUS, 11\$	
48	AE		13	D0 00174	MOVL	#19, IMAGE_DESC	0315
4C	AE	FE70	CF	9E 00178	MOVAB	P.AAC, IMAGE_DESC+4	0316
			7E	7C 0017E	CLRQ	-(SP)	0325
	7E	74	AE	3C 00180	MOVZWL	CHARBUF+12, -(SP)	
			7E	D4 00184	CLRL	-(SP)	
		10	AE	DD 00186	PUSHL	PRIB	
		FF68	CD	9F 00189	PUSHAB	CMDMBXNAM_DESC	
			7E	D4 0018D	CLRL	-(SP)	
		5C	AE	9F 0018F	PUSHAB	PRVMSK	
		F8	AD	9F 00192	PUSHAB	OUTPUT_DESC	
		F8	AD	9F 00195	PUSHAB	OUTPUT_DESC	
		FF68	CD	9F 00198	PUSHAB	CMDMBXNAM_DESC	
		74	AE	9F 0019C	PUSHAB	IMAGE_DESC	
		5C	AE	9F 0019F	PUSHAB	SUBPID	
00000000G	00		0D	FB 001A2	CALLS	#13, SYSS\$CREPRC	
	2A		50	E9 001A9	BLBC	STATUS, 11\$	
		18	AE	9F 001AC	PUSHAB	TRMMBXEFN	0331
	69		01	FB 001AF	CALLS	#1, LIB\$GET_EF	
	79		50	E9 001B2	BLBC	STATUS, 13\$	
		14	AE	9F 001B5	PUSHAB	CMDMBXEFN	0332
	69		01	FB 001B8	CALLS	#1, LIB\$GET_EF	
50	18		50	E9 001BB	BLBC	STATUS, 13\$	
			20	C7 001BE	DIVL3	#32, TRMMBXEFN, R0	0333
53		14	AE	D0 001C3	MOVL	CMDMBXEFN, R7	
	57		20	C7 001C7	DIVL3	#32, R7, R3	
	53		50	D1 001CB	CMPL	R0, R3	
			27	13 001CE	BEQL	12\$	
		10	AE	9F 001D0	PUSHAB	TMPMBXEFN	0338
	69		01	FB 001D3	CALLS	#1, LIB\$GET_EF	
	55		50	E9 001D6	BLBC	STATUS, 13\$	
		18	AE	9F 001D9	PUSHAB	TRMMBXEFN	0339
	6A		01	FB 001DC	CALLS	#1, LIB\$FREE_EF	
	18		AE	D0 001DF	MOVL	TMPMBXEFN, TRMMBXEFN	0340
50	18		20	C7 001E4	DIVL3	#32, TRMMBXEFN, R0	0341
			50	D1 001E9	CMPL	R0, R3	
			09	13 001EC	BEQL	12\$	
	52	00000000G	8F	D0 001EE	MOVL	#LIB\$_INSEF, STATUS	0343
			58	11 001F5	BRB	15\$	
	55		AE	D0 001F7	MOVL	TRMMBXEFN, R5	0345
51	55		00	EF 001FB	EXTZV	#0, #5, R5, R1	
	51		51	78 00200	ASHL	R1, #1, R1	
50	57		00	EF 00204	EXTZV	#0, #5, R7, R0	
	50		50	78 00209	ASHL	R0, #1, R0	0346
	54		51	C9 0020D	BISL3	R1, R0, EFN_MASK	

		7E	7C	00211	CLRQ	-(SP)	0357
		7E	7C	00213	CLRQ	-(SP)	
7E	54	8F	9A	00215	MOVZBL	#84, -(SP)	
	00F8	CE	9F	00219	PUSHAB	TRMM\$GBUF	
		7E	7C	0021D	CLRQ	-(SP)	
	FF58	CD	9F	0021F	PUSHAB	TRMM\$BXIOSB	
		21	DD	00223	PUSHL	#33	
7E	30	AE	3C	00225	MOVZWL	TRMM\$BXCHAN, -(SP)	
		55	DD	00229	PUSHL	R5	
68		0C	FB	0022B	CALLS	#12, SYSS\$QIO	
77		50	E9	0022E	BLBC	STATUS, 18\$	
56		6C	9A	00231	MOVZBL	(AP), R6	0362
		53	D4	00234	CLRL	PARAM	
		51	11	00236	BRB	17\$	
		6C43	D5	00238	TSTL	(AP)[PARAM]	0364
		4C	13	0023B	BEQL	17\$	
50		6C43	D0	0023D	MOVL	(AP)[PARAM], R0	0371
0100		8F	B1	00241	CMPW	(R0), #256	
		0A	1B	00246	BLEQU	16\$	
52	00000000G	8F	D0	00248	MOVL	#LIB\$_INVARG, STATUS	0373
		00A3	31	0024F	BRW	20\$	
		7E	7C	00252	CLRQ	-(SP)	0380
		7E	7C	00254	CLRQ	-(SP)	
50		6C43	D0	00256	MOVL	(AP)[PARAM], R0	
		60	DD	0025A	PUSHL	(R0)	
50		6C43	D0	0025C	MOVL	(AP)[PARAM], R0	
	04	A0	DD	00260	PUSHL	4(R0)	
		7E	7C	00263	CLRQ	-(SP)	
	FF60	CD	9F	00265	PUSHAB	CMDMBXIOSB	
		20	DD	00269	PUSHL	#32	
7E	2C	AE	3C	0026B	MOVZWL	CMDMBXCHAN, -(SP)	
		57	DD	0026F	PUSHL	R7	
68		0C	FB	00271	CALLS	#12, SYSS\$QIO	
7B		50	E9	00274	BLBC	STATUS, 19\$	
		54	DD	00277	PUSHL	EFN_MASK	0381
		55	DD	00279	PUSHL	R5	
68		02	FB	0027B	CALLS	#2, SYSS\$WFLOR	
71		50	E9	0027E	BLBC	STATUS, 19\$	
52	FF60	CD	3C	00281	MOVZWL	CMDMBXIOSB, STATUS	0382
6C		52	E9	00286	BLBC	STATUS, 20\$	0383
AB		56	F3	00289	AOBLEQ	R6, PARAM, 14\$	0364
53		7E	7C	0028D	CLRQ	-(SP)	0396
		7E	7C	0028F	CLRQ	-(SP)	
		01	DD	00291	PUSHL	#1	
	FD68	CF	9F	00293	PUSHAB	P.AAD	
		7E	7C	00297	CLRQ	-(SP)	
	FF60	CD	9F	00299	PUSHAB	CMDMBXIOSB	
		20	DD	0029D	PUSHL	#32	
7E	2C	AE	3C	0029F	MOVZWL	CMDMBXCHAN, -(SP)	
		57	DD	002A3	PUSHL	R7	
68		0C	FB	002A5	CALLS	#12, SYSS\$QIO	
47		50	E9	002A8	BLBC	STATUS, 19\$	
		54	DD	002AB	PUSHL	EFN_MASK	0397
		55	DD	002AD	PUSHL	R5	
68		02	FB	002AF	CALLS	#2, SYSS\$WFLOR	
3D		50	E9	002B2	BLBC	STATUS, 19\$	
52	FF60	CD	3C	002B5	MOVZWL	CMDMBXIOSB, STATUS	0398

LIB\$EXECUTE_CLI LIB\$EXECUTE_CLI - Execute CLI commands
V04-000 LIB\$EXECUTE_CLI - Execute CLI commands

G 2
16-Sep-1984 02:25:23
14-Sep-1984 13:34:25

VAX-11 Bliss-32 V4.0-742
[VMSLIB.SRC]LIBEXECLI.B32;1

Page 14
(3)

38		52	E9	002BA	BLBC	STATUS, 20\$	: 0399
		7E	7C	002BD	CLRQ	-(SP)	: 0407
		7E	7C	002BF	CLRQ	-(SP)	
		7E	7C	002C1	CLRQ	-(SP)	
		7E	7C	002C3	CLRQ	-(SP)	
	FF60	CD	9F	002C5	PUSHAB	CMDMBXIOSB	
		28	DD	002C9	PUSHL	#40	
7E	2C	AE	3C	002CB	MOVZWL	CMDMBXCHAN, -(SP)	
		57	DD	002CF	PUSHL	R7	
68		0C	FB	002D1	CALLS	#12, SYSSQIO	
1B		50	E9	002D4	BLBC	STATUS, 19\$	
		54	DD	002D7	PUSHL	EFN_MASK	: 0408
		55	DD	002D9	PUSHL	R5	
6B		02	FB	002DB	CALLS	#2, SYSSWFLOR	
11		50	E9	002DE	BLBC	STATUS, 19\$	
52	FF60	CD	3C	002E1	MOVZWL	CMDMBXIOSB, STATUS	: 0409
0C		52	E9	002E6	BLBC	STATUS, 20\$	: 0410
		55	DD	002E9	PUSHL	R5	: 0412
00000000G	00	01	FB	002EB	CALLS	#1, SYSSWAITFR	
	52	50	D0	002F2	19\$: MOVL	R0, STATUS	
	7E	04	AE	3C	20\$: MOVZWL	CMDMBXCHAN, -(SP)	: 0418
00000000G	00	01	FB	002F9	CALLS	#1, SYSSDASSGN	
	7E	08	AE	3C	MOVZWL	TRMBXCHAN, -(SP)	: 0419
00000000G	00	01	FB	00304	CALLS	#1, SYSSDASSGN	
		2C	AE	D5	TSTL	SUBPID	: 0420
		0C	13	0030E	BEQL	21\$	
		7E	D4	00310	CLRL	-(SP)	: 0422
	30	AE	9F	00312	PUSHAB	SUBPID	
00000000G	00	02	FB	00315	CALLS	#2, SYSSDELPRC	
		14	AE	9F	21\$: PUSHAB	CMDMBXEFN	: 0424
	6A	01	FB	0031F	CALLS	#1, LIB\$FREE_EF	
		18	AE	9F	PUSHAB	TRMBXEFN	: 0425
	6A	01	FB	00325	CALLS	#1, LIB\$FREE_EF	
		1C	AE	9F	PUSHAB	EXITBLOCK	: 0426
00000000G	00	01	FB	0032B	CALLS	#1, SYSSCANEXH	
	05	FF58	CD	E9	BLBC	TRMBXIOSB, 22\$	: 0427
	52	00E8	CE	D0	MOVL	TRMSGBUF+4, STATUS	: 0429
	50	52	D0	0033C	22\$: MOVL	STATUS, R0	: 0430
		04	0033F	RET			: 0432

; Routine Size: 832 bytes, Routine Base: _LIB\$CODE + 0026

; 435 0433 1

LIB\$EXECUTE_CLI LIB\$EXECUTE_CLI - Execute CLI commands
V04-000 LIB\$EXECUTE_CLI - Execute CLI commands

H 2
16-Sep-1984 02:25:23
14-Sep-1984 13:34:25

VAX-11 Bliss-32 V4.0-742
[VMSLIB.SRC]LIB\$EXECUTE_CLI.B32;1

Page 15
(4)

```

: 437      0434 1 ROUTINE EXIT_HANDLER(EXITSTATUS,SUBPID) : NOVALUE =
: 438      0435 1 ++
: 439      0436 1 FUNCTIONAL DESCRIPTION:
: 440      0437 1
: 441      0438 1 EXIT_HANDLER is called as an image exit handler when the
: 442      0439 1 current process is rundown. Its only function is to delete
: 443      0440 1 the subprocess.
: 444      0441 1
: 445      0442 1 INPUTS:
: 446      0443 1
: 447      0444 1 EXITSTATUS - Address of exit status.
: 448      0445 1
: 449      0446 1 SUBPID - Subprocess ID.
: 450      0447 1
: 451      0448 1 OUTPUTS:
: 452      0449 1
: 453      0450 1 Subprocess deleted.
: 454      0451 1
: 455      0452 1 --
: 456      0453 2 BEGIN
: 457      0454 2 IF SUBPID NEQ 0 ! If process created
: 458      0455 2 THEN
: 459      0456 3 $DELPRC(PIDADR=SUBPID) ! Delete the subprocess
: 460      0457 1 END;
```

```

                                0000 00000 EXIT_HANDLER:
                                .WORD Save nothing
                                08 AC D5 00002 TSTL SUBPID
                                0C 13 00005 BEQL 1$
                                7E D4 00007 CLRL -(SP)
                                08 AC 9F 00009 PUSHAB SUBPID
                                02 FB 0000C CALLS #2, SYSS$DELPRC
                                04 00013 1$: RET
00000000G 00
```

; Routine Size: 20 bytes, Routine Base: _LIB\$CODE + 0366

; 461 0458 1

LIB\$EXECUTE_CLI LIB\$EXECUTE_CLI - Execute CLI commands
V04-000 LIB\$EXECUTE_CLI - Execute CLI commands

I 2
16-Sep-1984 02:25:23 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 13:34:25 [VMSLIB.SRC]LIBEXECLI.B32;1

Page 16
(5)

: 463 0459 1 END
: 464 0460 1
: 465 0461 0 ELUDOM

! End of module LIB\$EXECUTE_CLI

PSECT SUMMARY

Name	Bytes	Attributes
_LIB\$CODE	890	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	26	0	581	00:01.0

: Information: 1
: Warnings: 0
: Errors: 0

COMMAND QUALIFIERS

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

: Size: 852 code + 38 data bytes
: Run Time: 00:19.0
: Elapsed Time: 00:38.8
: Lines/CPU Min: 1455
: Lexemes/CPU-Min: 25458
: Memory Used: 270 pages
: Compilation Complete

0436 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY