

[illegible]

[illegible]

```

LL          IIIII
LL          IIIII
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LLLLLLLLLLL
LLLLLLLLLLL

SSSSSSSSS
SSSSSSSSS
SS
SS
SS
SS
SSSSSSS
SSSSSSS
SS
SS
SS
SS
SSSSSSSSS
SSSSSSSSS

```

OP
VO
;


```
1 0001 0 MODULE opc$rqstmain ( %TITLE 'REQUEST command main module' %SBTTL 'Copyright notice'
2 0002 0
3 0003 0 LANGUAGE (BLISS32),
4 0004 0 IDENT = 'V04-000',
5 0005 0 MAIN = rqstmain_main
6 0006 0 ) =
7 0007 0
8 0008 0 *****
9 0009 0 *
10 0010 0 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
11 0011 0 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
12 0012 0 * ALL RIGHTS RESERVED.
13 0013 0 *
14 0014 0 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
15 0015 0 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
16 0016 0 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
17 0017 0 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
18 0018 0 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
19 0019 0 * TRANSFERRED.
20 0020 0 *
21 0021 0 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
22 0022 0 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
23 0023 0 * CORPORATION.
24 0024 0 *
25 0025 0 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
26 0026 0 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
27 0027 0 *
28 0028 0 *****
29 0029 0
30 0030 0 ++
31 0031 0 FACILITY:
32 0032 0
33 0033 0 REQUEST command
34 0034 0
35 0035 0 ABSTRACT:
36 0036 0
37 0037 0 This module contains the top level logic for the DCL REQUEST command.
38 0038 0
39 0039 0 Environment:
40 0040 0
41 0041 0 VAX/VMS operating system.
42 0042 0
43 0043 0 Author:
44 0044 0
45 0045 0 CW Hobbs, macro module REQUEST.MAR used as guide
46 0046 0
47 0047 0 Creation date:
48 0048 0
49 0049 0 22-Aug-1983
50 0050 0
51 0051 0 Revision history:
52 0052 0
53 0053 0 V03-003 CWH3003 CW Hobbs 18-May-1984
54 0054 0 Add a . to the check for request pending.
55 0055 0
56 0056 0 V03-002 CWH3169 CW Hobbs 5-May-1984
57 0057 0 Second pass for cluster-wide OPCOM:
```

OPC\$RQSTMAIN
V04-000

REQUEST command main module
Copyright notice

M 11

16-Sep-1984 01:47:25

14-Sep-1984 12:50:54

VAX-11 Bliss-32 V4.0-742

[OPCOM.SRC]RQSTMAIN.B32;1

Page 2
(1)

: 58
: 59
: 60
: 61
: 62
: 0058 0 !
: 0059 0 !
: 0060 0 !
: 0061 0 !
: 0062 0 !--

- If a reply is requested (\$ REQUEST /REPLY), then return
the operator response code. This restores the behaviour
of the V3 macro program.


```

: 64      0063 1 BEGIN                                     %SBTTL 'Start of rqstmain'
: 65      0064 1
: 66      0065 1 LIBRARY 'SYSS$LIBRARY:LIB.L32';
: 67      0066 1 LIBRARY 'LIB$OPCOMLIB';
: 68      0067 1
: 69      0068 1 FORWARD ROUTINE
: 70      0069 1     rqstmain_ctrlc_ast : NOVALUE,          ! Handle control c ast
: 71      0070 1     rqstmain_init,          ! Initializations
: 72      0071 1     rqstmain_main,          ! Entry point, main routine
: 73      0072 1     rqstmain_setmode_qio : NOVALUE,        ! Set up control c ast
: 74      0073 1     rqstmain_wait_reply;          ! Wait for reply from opcom
: 75      0074 1
: 76      0075 1 EXTERNAL ROUTINE
: 77      0076 1     share_lookup_oper_bit,          ! Convert text string to operator bit number
: 78      0077 1     share_trnlog : NOVALUE;          ! Recursively translate a name
: 79      0078 1
: 80      0079 1 LITERAL
: 81      0080 1     mb_buf_siz = 256,
: 82      0081 1     message_buf_siz = 512;
: 83      0082 1
: 84      0083 1 OWN
: 85      0084 1     dvi_terminal_buf : $bvector [max_dev_nam],
: 86      0085 1     dvi_terminal_len,          ! Length of terminal name
: 87      0086 1     dvi_terminal_ptr : INITIAL (dvi_terminal_buf),
: 88      0087 1     dvi_devchar : $bblock [4],
: 89      0088 1     dvi_items : VECTOR [7, LONG] PRESET (
: 90      0089 1         [0] = (dvi$devchar^16 OR 4),
: 91      0090 1         [1] = dvi_devchar,
: 92      0091 1         [2] = 0,
: 93      0092 1         [3] = (dvi$fulldevnam^16 OR max_dev_nam),
: 94      0093 1         [4] = dvi_terminal_buf,
: 95      0094 1         [5] = dvi_terminal_len,
: 96      0095 1         [6] = 0),
: 97      0096 1     jpi_prcnam_buf : $bvector [max_dev_nam],
: 98      0097 1     jpi_prcnam_len,
: 99      0098 1     jpi_prcnam_ptr : INITIAL (jpi_prcnam_buf),
100      0099 1     jpi_items : VECTOR [4, LONG] PRESET (
101      0100 1         [0] = (jpi$prcnam^16 OR max_dev_nam),
102      0101 1         [1] = jpi_prcnam_buf,
103      0102 1         [2] = jpi_prcnam_len,
104      0103 1         [3] = 0),
105      0104 1     mb_chan,
106      0105 1     mb_iosb : VECTOR [4, WORD],
107      0106 1     mb_buffer : $bvector [mb_buf_siz],
108      0107 1     tt_chan,
109      0108 1     tt_iosb : VECTOR [4, WORD],
110      0109 1     text : $dyn_str_desc,
111      0110 1     message : $bblock [message_buf_siz]          ! Buffer to build message for sndopr
112      0111 1     message_desc : VECTOR [2, LONG] PRESET ([1] = message);
113      0112 1
114      0113 1 ! Define ascii text descriptors once
115      0114 1
116      0115 1 BIND
117      0116 1     ascid_P1 = %ASCID 'P1',
118      0117 1     ascid_REPLY = %ASCID 'REPLY',
119      0118 1     ascid_SYSCOMMAND = %ASCID 'SYSS$COMMAND',
120      0119 1     ascid_TO = %ASCID 'TO';
```



```
: 122 0120 1 GLOBAL ROUTINE rqstmain_ctrlc_ast : NOVALUE = %SBTTL 'rqstmain_ctrlc_ast'
: 123 0121 1
: 124 0122 1 ++
: 125 0123 1 Functional description:
: 126 0124 1
: 127 0125 1 Handle a control c ast from the terminal.
: 128 0126 1
: 129 0127 1 Input:
: 130 0128 1
: 131 0129 1 None.
: 132 0130 1
: 133 0131 1 Implicit Input:
: 134 0132 1
: 135 0133 1 None.
: 136 0134 1
: 137 0135 1 Output:
: 138 0136 1
: 139 0137 1 None.
: 140 0138 1
: 141 0139 1 Implicit output:
: 142 0140 1
: 143 0141 1 None.
: 144 0142 1
: 145 0143 1 Side effects:
: 146 0144 1
: 147 0145 1 None.
: 148 0146 1
: 149 0147 1 Routine value:
: 150 0148 1
: 151 0149 1 None.
: 152 0150 1 --
: 153 0151 1
: 154 0152 2 BEGIN ! Start of rqstmain_ctrlc_ast
: 155 0153 2
: 156 0154 2 LOCAL
: 157 0155 2 prmt : $bvector [128],
: 158 0156 2 dsc : VECTOR [2, LONG] PRESET ([0] = 128, [1] = prmt),
: 159 0157 2 status;
: 160 0158 2
: 161 0159 2 BIND
: 162 0160 2 buf = (message [opc$t_request_text] + 2 + .dvi_terminal_len) : $bvector;
: 163 0161 2
: 164 0162 2 Get the prompt string
: 165 0163 2
: 166 0164 3 IF NOT (status = $getmsg (msgid=opc$_rqst_prompt, msglen=dsc, bufadr=dsc, flags=0))
: 167 0165 2 THEN
: 168 0166 2 $signal_stop (.status);
: 169 0167 2
: 170 0168 2 Get the response from the user, read it into the message right after the terminal name.
: 171 0169 2
: 172 P 0170 2 status = $qiow (efn=1, chan=.tt_chan, func=io$_readprompt, iosb=tt_iosb,
: 173 0171 2 p1=buf, p2=256, p5=.dsc [1], p6=.dsc [0]);
: 174 0172 2 IF .status
: 175 0173 2 THEN
: 176 0174 2 status = .tt_iosb [0];
: 177 0175 2 IF NOT .status
: 178 0176 2 THEN
```



```
179 0177 2 $signal_stop (.status);
180 0178 2
181 0179 2 If ^Z, then cancel the request, otherwise send the message to the operator
182 0180 2
183 0181 2 IF .(tt_iosb [2]) <0,8,0> EQL %X'1A'
184 0182 2 THEN
185 0183 2 BEGIN
186 0184 2 message [opc$b_rqstcode] = opc$x_cancel;
187 0185 2 message [opc$w_request_length] = 0;
188 0186 2 message_desc [0] = $byteoffset (opc$t_request_text);
189 0187 2 IF NOT (status = $sndopr (msgbuf=message_desc, chan=.mb_chan))
190 0188 2 THEN
191 0189 2 $signal_stop (.status);
192 0190 2 END
193 0191 2 ELSE IF .tt_iosb [1] EQL 0
194 0192 2 THEN
195 0193 2 RETURN rqstmain_ctrlc_ast () ! Try again if no input
196 0194 2 ELSE
197 0195 2 BEGIN
198 0196 2 LOCAL
199 0197 2 add;
200 0198 2 add = (2 + .dvi_terminal_len + .tt_iosb [1]);
201 0199 2 message [opc$w_request_length] = .add;
202 0200 2 message_desc [0] = $byteoffset (opc$t_request_text) + .add;
203 0201 2 IF NOT (status = $sndopr (msgbuf=message_desc, chan=0))
204 0202 2 THEN
205 0203 2 $signal_stop (.status);
206 0204 2 rqstmain_setmode_qio (); ! Reenable the AST
207 0205 2 END;
208 0206 2
209 0207 2 RETURN;
210 0208 1 END; ! End of rqstmain_ctrlc_ast
```

```
.TITLE OPC$RQSTMAIN REQUEST command main module
.IDENT \V04-000\
```

```
.PSECT $PLITS$,NOWRT,NOEXE,2
```

```
00 00 31 50 00000 P.AAB: .ASCII \P1\<0><0>
010E0002 00004 P.AAA: .LONG 17694722
00000000' 00008 .ADDRESS P.AAB
00 00 00 59 4C 50 45 52 0000C P.AAD: .ASCII \REPLY\<0><0><0>
010E0005 00014 P.AAC: .LONG 17694725
00000000' 00018 .ADDRESS P.AAD
00 44 4E 41 4D 4D 4F 43 24 53 59 53 0001C P.AAF: .ASCII \SYS$COMMAND\<0>
010E000B 00028 P.AAE: .LONG 17694731
00000000' 0002C .ADDRESS P.AAF
00 00 4F 54 00030 P.AAH: .ASCII \T0\<0><0>
010E0002 00034 P.AAG: .LONG 17694722
00000000' 00038 .ADDRESS P.AAH
```

```
.PSECT $OWNS$,NOEXE,2
```

```
00000 DVI_TERMINAL_BUF:
.BLK 64
00040 DVI_TERMINAL_LEN:
```



```
00000000' 00044 DVI_TERMINAL_PTR: .BLKB 4
                                .ADDRESS DVI_TERMINAL_BUF
00048 DVI_DEVCHAR: .BLKB 4
00020004 0004C DVI_ITEMS: .LONG 131076
                                .ADDRESS DVI_DEVCHAR
00E80040 00000000' 00050 .LONG 0, 15204416
00000000' 00000000' 00054 .ADDRESS DVI_TERMINAL_BUF, DVI_TERMINAL_LEN
00000000' 00000000' 0005C .LONG 0
00000000' 00064 JPI_PRCNAM_BUF: .BLKB 64
00068 JPI_PRCNAM_LEN: .BLKB 4
000A8 JPI_PRCNAM_PTR: .ADDRESS JPI_PRCNAM_BUF
00000000' 000AC JPI_ITEMS: .LONG 52166720
031C0040 000B0 .ADDRESS JPI_PRCNAM_BUF, JPI_PRCNAM_LEN
00000000' 00000000' 000B4 .LONG 0
00000000' 000BC MB_CHAN: .BLKB 4
000C0 MB_IOSB: .BLKB 8
000C4 MB_BUFFER: .BLKB 256
000CC MB_BUFFER: .BLKB 4
001CC TT_CHAN: .BLKB 8
001D0 TT_IOSB: .BLKB 8
0000 001D8 TEXT: .WORD 0
02 0E 001DA .BYTE 14, 2
00000000 001DC .LONG 0
001E0 MESSAGE: .BLKB 512
00# 003E0 MESSAGE_DESC: .BYTE 0[4]
00000000' 003E4 .ADDRESS MESSAGE
```

```
ASCID_P1= P.AAA
ASCID_REPLY= P.AAC
ASCID_SYSCOMMAND= P.AAE
ASCID_TO= P.AAG
```

```
.EXTRN SHARE_LOOKUP_OPER_BIT
.EXTRN SHARE_TRNLOG, SYS$GETMSG
.EXTRN LIB$STOP, SYS$QIOW
.EXTRN SYS$SNDOPR
```

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

```
04 55 00000000G 003C 00000 .ENTRY RQSTMAIN_CTRLC_AST, Save R2,R3,R4,R5
54 0000' 00 9E 00002 MOVAB SYS$SNDOPR, R5
5E FF7C CE 9E 00009 MOVAB MESSAGE_DESC, R4
7E 80 8F 9A 00013 MOVAB -132(SP), SP
AE 08 AE 9E 00017 MOVZBL #128, DSC
52 FE1E C4 9E 0001C MOVAB PRMP, DSC+4
52 FC60 C4 C0 00021 MOVAB MESSAGE+30, R2
ADDL2 DVI_TERMINAL_LEN, R2
CLRQ -(SP)
PUSHAB DSC
PUSHAB DSC
```

0120
0156
0160
0164

00000000G	00	000582BB	8F	DD	0002E	PUSHL	#361147	:	
	53		05	FB	00034	CALLS	#5, SYSS\$GETMSG	:	
	50		50	DO	0003B	MOVL	R0, STATUS	:	
			53	E9	0003E	BLBC	STATUS, 1\$	:	
		08	6E	DD	00041	PUSHL	DSC	:	0171
			AE	DD	00043	PUSHL	DSC+4	:	
			7E	7C	00046	CLRQ	-(SP)	:	
	7E	0100	8F	3C	00048	MOVZWL	#256, -(SP)	:	
			52	DD	0004D	PUSHL	R2	:	
			7E	7C	0004F	CLRQ	-(SP)	:	
		FDF0	C4	9F	00051	PUSHAB	TT_IOSB	:	
			37	DD	00055	PUSHL	#55	:	
		FDEC	C4	DD	00057	PUSHL	TT_CHAN	:	
			01	DD	0005B	PUSHL	#1	:	
00000000G	00		0C	FB	0005D	CALLS	#12, SYSS\$QIOW	:	
	53		50	DO	00064	MOVL	R0, STATUS	:	0172
	57		53	E9	00067	BLBC	STATUS, 4\$	:	0174
	53	FDF0	C4	3C	0006A	MOVZWL	TT_IOSB, STATUS	:	0175
	4F		53	E9	0006F	BLBC	STATUS, 4\$	:	0181
	1A	FDF4	C4	91	00072	CMPB	TT_IOSB+4, #26	:	
			1C	12	00077	BNEQ	2\$	:	
FE00	C4		0E	90	00079	MOVB	#14, MESSAGE	:	0184
		FE1A	C4	B4	0007E	CLRQ	MESSAGE+26	:	0185
	64		1C	DO	00082	MOVL	#28, MESSAGE_DESC	:	0186
		FCE0	C4	DD	00085	PUSHL	MB_CHAN	:	0187
			54	DD	00089	PUSHL	R4	:	
	65		02	FB	0008B	CALLS	#2, SYSS\$NDOPR	:	
	53		50	DO	0008E	MOVL	R0, STATUS	:	
	2D		53	E9	00091	BLBC	STATUS, 4\$	:	0189
			04		00094	RET		:	0191
	52	FDF2	C4	3C	00095	MOVZWL	TT_IOSB+2, R2	:	
			06	12	0009A	BNEQ	3\$	:	
FF5F	CF		00	FB	0009C	CALLS	#0, RQSTMAIN_CTRL_C_AST	:	0193
			04		000A1	RET		:	
50	52	FC60	C4	C1	000A2	ADDL3	DVI_TERMINAL_LEN, R2, R0	:	0198
	50		02	C0	000A8	ADDL2	#2, ADD	:	
FE1A	C4		50	B0	000AB	MOVW	ADD, MESSAGE+26	:	0199
	64	1C	A0	9E	000B0	MOVAB	28(R0), MESSAGE_DESC	:	0200
			7E	D4	000B4	CLRL	-(SP)	:	0201
			54	DD	000B6	PUSHL	R4	:	
	65		02	FB	000B8	CALLS	#2, SYSS\$NDOPR	:	
	53		50	DO	000BB	MOVL	R0, STATUS	:	
	0A		53	E8	000BE	BLBS	STATUS, 5\$	:	
			53	DD	000C1	PUSHL	STATUS	:	0203
00000000G	00		01	FB	000C3	CALLS	#1, LIB\$STOP	:	
			04		000CA	RET		:	
0000V	CF		00	FB	000CB	CALLS	#0, RQSTMAIN_SETMODE_QIO	:	0204
			04		000D0	RET		:	0208

; Routine Size: 209 bytes, Routine Base: \$CODE\$ + 0000


```
: 212      0209 1 GLOBAL ROUTINE rqstmain_init =          %SBTTL 'rqstmain_init routine'
: 213      0210 1
: 214      0211 1 ++
: 215      0212 1 Functional description:
: 216      0213 1
: 217      0214 1 This is the initialization routine for REQUEST. Various common initializations are done.
: 218      0215 1
: 219      0216 1 Input:
: 220      0217 1
: 221      0218 1 None.
: 222      0219 1
: 223      0220 1 Implicit Input:
: 224      0221 1
: 225      0222 1 Command values from CLI routines.
: 226      0223 1
: 227      0224 1 Output:
: 228      0225 1
: 229      0226 1 None.
: 230      0227 1
: 231      0228 1 Implicit output:
: 232      0229 1
: 233      0230 1 None.
: 234      0231 1
: 235      0232 1 Side effects:
: 236      0233 1
: 237      0234 1 None.
: 238      0235 1
: 239      0236 1 Routine value:
: 240      0237 1
: 241      0238 1 None.
: 242      0239 1 --
: 243      0240 1
: 244      0241 2 BEGIN                                ! Start of rqstmain_init
: 245      0242 2
: 246      0243 2 LOCAL
: 247      0244 2 status;
: 248      0245 2
: 249      0246 2 Initialize the message
: 250      0247 2
: 251      0248 2 NOTE: We are using an internal interface to OPCOM which is subject to change!
: 252      0249 2
: 253      0250 2 CH$FILL (0, opc$k_request_min_size, message); ! Init all fixed fields to zero
: 254      0251 2 message [opc$b_rqstcode] = opc$x_request;
: 255      0252 2 message [opc$b_scope] = opc$k_system;
: 256      0253 2
: 257      0254 2 Do a $GETJPI to get information about the current process
: 258      0255 2
: 259      0256 3 IF NOT (status = $getjpi (itmlst=jpi_items))
: 260      0257 2 THEN
: 261      0258 2 $signal_stop (.status);
: 262      0259 2
: 263      0260 2 Do a $GETDVI to get the name of the command terminal.
: 264      0261 2
: 265      0262 3 IF NOT (status = $getdvi (devnam=ascid_SYSCOMMAND, itmlst=dvi_items))
: 266      0263 2 THEN
: 267      0264 2 $signal_stop (.status);
: 268      0265 2 !
```



```
269 0266 2 ! If sys$command is a terminal, assign a channel
270 0267 2
271 0268 2 IF .dvi_devchar [dev$v_trm]
272 0269 2 THEN
273 0270 2 BEGIN
274 0271 2 IF NOT (status = $assign (devnam=dvi_terminal_len, chan=tt_chan))
275 0272 2 THEN
276 0273 2 $signal_stop (.status);
277 0274 2 END
278 0275 2
279 0276 2 ! If sys$command is not a terminal, then substitute the process name for the terminal name
280 0277 2
281 0278 2 ELSE
282 0279 2 BEGIN
283 0280 2 dvi_terminal_len = .jpi_prcnam_len;
284 0281 2 dvi_terminal_ptr = .jpi_prcnam_ptr;
285 0282 2 END;
286 0283 2
287 0284 2 ! If /TO is requested, then set the attention bitmask to those operators
288 0285 2
289 0286 2 IF cli$get_value (ascid_TO, text)
290 0287 2 THEN
291 0288 2 BEGIN
292 0289 2 DO
293 0290 2 $bblock [message [opc$l_attnmask1], 0, share_lookup_oper_bit (text), 1, 0] = 1
294 0291 2 UNTIL
295 0292 2 NOT cli$get_value (ascid_TO, text);
296 0293 2
297 0294 2 END
298 0295 2
299 0296 2 ELSE
300 0297 2 BEGIN
301 0298 2 message [opc$l_attnmask1] = known_attn_mask1;
302 0299 2 message [opc$l_attnmask2] = known_attn_mask2;
303 0300 2 END;
304 0301 2
305 0302 2 ! If /REPLY is requested, create a mailbox
306 0303 2
307 0304 2 IF cli$present (ascid_REPLY)
308 0305 2 THEN
309 0306 2 BEGIN
310 0307 2 IF NOT (status = $crembx (chan=mb_chan, maxmsg=mb_buf_siz, bufquo=2*mb_buf_siz, promsk=%X'FF'))
311 0308 2 THEN
312 0309 2 $signal_stop (.status);
313 0310 2 END;
314 0311 2
315 0312 2 ! Move the terminal (process) name and reply text, if any
316 0313 2
317 0314 2 cli$get_value (ascid_P1, text); ! Get the parameter, null is fine
318 0315 2 CH$COPY (.dvi_terminal_len, .dvi_terminal_ptr,
319 0316 2 2, UPLIT BYTE (' '), .text [dsc$w_length], .text [dsc$a_pointer], 0,
320 0317 2 message_buf_siz-$byteoffset (opc$t_request_text), message [opc$t_request_text]);
321 0318 2 message [opc$w_request_length] = .text [dsc$w_length] + 2 + .dvi_terminal_len;
322 0319 2 message_desc [0] = .message [opc$w_request_length] + ! Set total length
323 0320 2 $byteoffset (opc$t_request_text);
324 0321 2
325 0322 2 RETURN .status;
```

```
! End of rqstmain_init
```

			.PSECT	\$SPLITS,NOWRT,NOEXE,2	
20	2C	0003C	P.AAI:	.ASCII	\, \
			.EXTRN	SYSS\$GETJPI, SYSS\$GETDVI	
			.EXTRN	SYSS\$ASSIGN, CLIS\$GET_VALUE	
			.EXTRN	CLIS\$PRESENT, SYSS\$CREMBX	
			.PSECT	\$CODE\$,NOWRT,2	
	OFFC	00000	.ENTRY	RQSTMAIN_INIT, Save R2,R3,R4,R5,R6,R7,R8,-	0209
CF	9E	00002	MOVAB	ASCID_TO, R11	
CF	9E	00007	MOVAB	DVI_TERMINAL_LEN, R10	
00	2C	0000C	MOVCS	#0, -(SP), #0, #28, MESSAGE	0250
CA		00011			
8F	B0	00014	MOVW	#268, MESSAGE	0251
7E	7C	0001B	CLRQ	-(SP)	0256
7E	D4	0001D	CLRL	-(SP)	
AA	9F	0001F	PUSHAB	JFI_ITEMS	
7E	7C	00022	CLRQ	-(SP)	
7E	D4	00024	CLRL	-(SP)	
07	FB	00026	CALLS	#7, SYSS\$GETJPI	
50	D0	0002D	MOVL	R0, STATUS	
59	E9	00030	BLBC	STATUS, 1\$	
7E	7C	00033	CLRQ	-(SP)	0262
7E	7C	00035	CLRQ	-(SP)	
AA	9F	00037	PUSHAB	DVI_ITEMS	
AB	9F	0003A	PUSHAB	ASCID_SYSCOMMAND	
7E	7C	0003D	CLRQ	-(SP)	
08	FB	0003F	CALLS	#8, SYSS\$GETDVI	
50	D0	00046	MOVL	R0, STATUS	
59	E9	00049	BLBC	STATUS, 1\$	
02	E1	0004C	BBC	#2, DVI_DEVCHAR, 2\$	0268
7E	7C	00051	CLRQ	-(SP)	0271
CA	9F	00053	PUSHAB	TT_CHAN	
5A	DD	00057	PUSHL	R10	
04	FB	00059	CALLS	#4, SYSS\$ASSIGN	
50	D0	00060	MOVL	R0, STATUS	
59	E8	00063	BLBS	STATUS, 3\$	
72	11	00066	BRB	8\$	0273
AA	7D	00068	MOVQ	JPI_PRCNAM_LEN, DVI_TERMINAL_LEN	0280
CA	9F	0006C	PUSHAB	TEXT	0286
5B	DD	00070	PUSHL	R11	
02	FB	00072	CALLS	#2, CLIS\$GET_VALUE	
50	E9	00079	BLBC	R0, 6\$	
CA	9F	0007C	PUSHAB	TEXT	0291
01	FB	00080	CALLS	#1, SHARE_LOOKUP_OPER_BIT	
50	E2	00085	BBSS	R0, MESSAGE+10, 5\$	
CA	9F	0008B	PUSHAB	TEXT	0293
5B	DD	0008F	PUSHL	R11	
02	FB	00091	CALLS	#2, CLIS\$GET_VALUE	
50	E9	00098	BLBC	R0, 7\$	

01AA	CA	00FFF1FF	DF	11	0009B	BRB	4\$		
		01AE	8F	D0	0009D	6\$:	MOVL	#16773631, MESSAGE+10	0298
		E0	CA	D4	000A6		CLRL	MESSAGE+14	0299
00000000G	00		AB	9F	000AA	7\$:	PUSHAB	ASCID REPLY	0304
	2D		01	FB	000AD		CALLS	#1, C[IS]PRESENT	
			50	E9	000B4		BLBC	R0, 9\$	
	7E	FF	7E	7C	000B7		CLRL	-(SP)	0307
	7E	0200	8F	9A	000B9		MOVZBL	#255, -(SP)	
	7E	0100	8F	3C	000BD		MOVZWL	#512, -(SP)	
		0080	8F	3C	000C2		MOVZWL	#256, -(SP)	
			CA	9F	000C7		PUSHAB	MB CHAN	
00000000G	00		7E	D4	000CB		CLRL	-(SP)	
	59		07	FB	000CD		CALLS	#7, SYSSCREMBX	
	0A		50	D0	000D4		MOVL	R0, STATUS	
00000000G	00		59	E8	000D7		BLBS	STATUS, 9\$	
			59	DD	000DA	8\$:	PUSHL	STATUS	0309
			01	FB	000DC		CALLS	#1, LIB\$STOP	
				04	000E3		RET		
		0198	CA	9F	000E4	9\$:	PUSHAB	TEXT	0314
		D0	AB	9F	000E8		PUSHAB	ASCID P1	
00000000G	00		02	FB	000EB		CALLS	#2, C[IS]GET_VALUE	
	58	0198	CA	3C	000F2		MOVZWL	TEXT, R8	0316
	57	01E4	8F	3C	000F7		MOVZWL	#484, R7	0317
	56	01BC	CA	9E	000FC		MOVAB	MESSAGE+28, R6	
57	00	04	6A	2C	00101		MOVCS	DVI_TERMINAL_LEN, @DVI_TERMINAL_PTR, #0, -	
			66		00107			R7, (R6)	
			1D	18	00108		BGEQ	10\$	
			6A	C0	0010A		ADDL2	DVI_TERMINAL_LEN, R6	
57	00	08	6A	C2	0010D		SUBL2	DVI_TERMINAL_LEN, R7	
			02	2C	00110		MOVCS	#2, P.AAI, #0, R7, (R6)	
			66		00116				
			0E	18	00117		BGEQ	10\$	
			02	C0	00119		ADDL2	#2, R6	
			02	C2	0011C		SUBL2	#2, R7	
57	00	019C	58	2C	0011F		MOVCS	R8, @TEXT+4, #0, R7, (R6)	
			66		00126				
			CA	3C	00127	10\$:	MOVZWL	TEXT, R0	0318
			6A	C0	0012C		ADDL2	DVI_TERMINAL_LEN, R0	
01BA	CA		02	A1	0012F		ADDW3	#2, R0, MESSAGE+26	
		03A0	CA	3C	00135		MOVZWL	MESSAGE+26, MESSAGE_DESC	0319
		03A0	1C	C0	0013C		ADDL2	#28, MESSAGE_DESC	
			59	D0	00141		MOVL	STATUS, R0	0322
			04	00144			RET		0323

; Routine Size: 325 bytes, Routine Base: \$CODE\$ + 00D1

```
.. 328 0324 1 GLOBAL ROUTINE rqstmain_main = %SBTTL 'rqstmain_main routine'
.. 329 0325 1
.. 330 0326 1 !++
.. 331 0327 1 Functional description:
.. 332 0328 1
.. 333 0329 1 This is the main routine for REQUEST. When REQUEST is started, control is transfered here.
.. 334 0330 1
.. 335 0331 1 Input:
.. 336 0332 1
.. 337 0333 1 None.
.. 338 0334 1
.. 339 0335 1 Implicit Input:
.. 340 0336 1
.. 341 0337 1 None.
.. 342 0338 1
.. 343 0339 1 Output:
.. 344 0340 1
.. 345 0341 1 None.
.. 346 0342 1
.. 347 0343 1 Implicit output:
.. 348 0344 1
.. 349 0345 1 None.
.. 350 0346 1
.. 351 0347 1 Side effects:
.. 352 0348 1
.. 353 0349 1 None.
.. 354 0350 1
.. 355 0351 1 Routine value:
.. 356 0352 1
.. 357 0353 1 None.
.. 358 0354 1 --
.. 359 0355 1
.. 360 0356 2 BEGIN ! Start of rqstmain_main
.. 361 0357 2
.. 362 0358 2 LOCAL
.. 363 0359 2 status;
.. 364 0360 2
.. 365 0361 2 Perform common initializations
.. 366 0362 2
.. 367 0363 2 rqstmain_init ();
.. 368 0364 2
.. 369 0365 2 Send the message to OPCOM
.. 370 0366 2
.. 371 0367 2 IF NOT (status = $sndopr (msgbuf=message_desc, chan=.mb_chan))
.. 372 0368 2 THEN
.. 373 0369 2 $signal_stop (.status);
.. 374 0370 2
.. 375 0371 2 If we are expecting a reply, then wait for it
.. 376 0372 2
.. 377 0373 2 IF .mb_chan NEQ 0
.. 378 0374 2 THEN
.. 379 0375 2 DO
.. 380 0376 2 status = rqstmain_wait_reply ()
.. 381 0377 2 UNTIL
.. 382 0378 2 .status NEQ 0;
.. 383 0379 2
.. 384 0380 2 RETURN .status;
```


OPC\$RQSTMAIN
V04-000

REQUEST command main module
rqstmain_main routine

K 12
16-Sep-1984 01:47:25
14-Sep-1984 12:50:54

VAX-11 Bliss-32 V4.0-742
[OPCOM.SRC]RQSTMAIN.B32;1

Page 13
(5)

; 385

0381 1 END;

! End of rqstmain_main

FEB4	CF		0000	0000	.ENTRY	RQSTMAIN MAIN, Save nothing	:	0324
			00	FB 00002	CALLS	#0, RQSTMAIN_INIT	:	0363
		0000'	CF	DD 00007	PUSHL	MB_CHAN	:	0367
		0000'	CF	9F 0000B	PUSHAB	MESSAGE_DESC	:	
00000000G	00		02	FB 0000F	CALLS	#2, SYSSNDOPR	:	
	51		50	D0 00016	MOVL	R0, STATUS	:	
	0A		51	E8 00019	BLBS	STATUS, 1\$	:	
			51	DD 0001C	PUSHL	STATUS	:	0369
00000000G	00		01	FB 0001E	CALLS	#1, LIB\$STOP	:	
				04 00025	RET		:	
		0000'	CF	D5 00026 1\$:	TSTL	MB_CHAN	:	0373
			0A	13 0002A	BEQL	3\$	:	
0000V	CF		00	FB 0002C 2\$:	CALLS	#0, RQSTMAIN_WAIT_REPLY	:	0376
	51		50	D0 00031	MOVL	R0, STATUS	:	
			F6	13 00034	BEQL	2\$	:	0378
	50		51	D0 00036 3\$:	MOVL	STATUS, R0	:	0380
				04 00039	RET		:	0381

; Routine Size: 58 bytes, Routine Base: \$CODE\$ + 0216

	0000	00000	.ENTRY	RQSTMAIN_SETMODE_QIO, Save nothing
	7E	7C 00002	CLRQ	-(SP)
	7E	7C 00004	CLRQ	-(SP)
	7E	D4 00006	CLRL	-(SP)
FDA4	CF	9F 00008	PUSHAB	RQSTMAIN_CTRL_C_AST
	7E	7C 0000C	CLRQ	-(SP)
	7E	D4 0000E	CLRL	-(SP)
0123	8F	3C 00010	MOVZWL	#291, -(SP)
0000'	CF	DD 00015	PUSHL	TT_CHAN
	7E	D4 00019	CLRL	-(SP)

: 0382
: 0419

OPC\$RQSTMAIN
V04-000

REQUEST command main module
rqstmain_setmode_qio

M 12
16-Sep-1984 01:47:25
14-Sep-1984 12:50:54

VAX-11 Bliss-32 V4.0-742
[OPCOM.SRC]RQSTMAIN.B32;1

Page 15
(6)

00000000G 00

7E

00058089

0C

FB 0001B

01

7D 00022

8F

DD 00025

03

FB 0002B

04 00032

CALLS

#12, SYSSQIOW

MOVQ

#1, -(SP)

PUSHL

#360585

CALLS

#3, LIB\$SIGNAL

RET

: 0420

: 0423

; Routine Size: 51 bytes, Routine Base: \$CODE\$ + 0250

```

: 430      0424 1 GLOBAL ROUTINE rqstmain_wait_reply =                %SBTTL 'rqstmain_wait_reply routine'
: 431      0425 1
: 432      0426 1 !++
: 433      0427 1 | Functional description:
: 434      0428 1 |
: 435      0429 1 |     This routines waits for a reply from OPCOM
: 436      0430 1 |
: 437      0431 1 | Input:
: 438      0432 1 |
: 439      0433 1 |     None.
: 440      0434 1 |
: 441      0435 1 | Implicit Input:
: 442      0436 1 |
: 443      0437 1 |     None.
: 444      0438 1 |
: 445      0439 1 | Output:
: 446      0440 1 |
: 447      0441 1 |     None.
: 448      0442 1 |
: 449      0443 1 | Implicit output:
: 450      0444 1 |
: 451      0445 1 |     None.
: 452      0446 1 |
: 453      0447 1 | Side effects:
: 454      0448 1 |
: 455      0449 1 |     None.
: 456      0450 1 |
: 457      0451 1 | Routine value:
: 458      0452 1 |
: 459      0453 1 |     None.
: 460      0454 1 | --
: 461      0455 1
: 462      0456 2 BEGIN                                           ! Start of rqstmain_wait_reply
: 463      0457 2
: 464      0458 2 LOCAL
: 465      0459 2 |     status;
: 466      0460 2 |
: 467      0461 2 | Enable the ast on control c
: 468      0462 2 |
: 469      0463 2 | rqstmain_setmode_qio ();
: 470      0464 2 |
: 471      0465 2 | Read from the mailbox
: 472      0466 2 |
: 473      0467 2 | status = $qiow (efn=2, chan=.mb_chan, func=io$_readvblk, iosb=mb_iosb, p1=mb_buffer, p2=mb_buf_siz);
: 474      0468 2 | IF .status
: 475      0469 2 | THEN
: 476      0470 2 |     status = .mb_iosb [0];
: 477      0471 2 | IF NOT .status
: 478      0472 2 | THEN
: 479      0473 2 |     $signal_stop (.status);
: 480      0474 2 |
: 481      0475 2 | Display the mailbox message on the terminal
: 482      0476 2 |
: 483      0477 2 | $signal (opc$_opreply, 2, .mb_iosb [1]-8, mb_buffer [8]);
: 484      0478 2 |
: 485      0479 2 | If the code is RQSTPEND, then we should continue, return 0 so the outer routine will loop
: 486      0480 2 |
```



```
: 487      0481 3 IF .(mb_buffer [2]) <0,16,0> EQL (opc$_rqstpend AND %X'FFFF')
: 488      0482 2 THEN
: 489      0483 2     RETURN 0;
: 490      0484 2
: 491      0485 2     Request is done, force OPCOM facility code back into the high word and return the
: 492      0486 2     status from the reply
: 493      0487 2
: 494      0488 2     (mb_buffer [4]) <0,16,0> = sts$m_inhib_msg^-16 OR opc$_facility;
: 495      0489 2     RETURN .(mb_buffer [2]);
: 496      0490 2
: 497      0491 1 END;
```

! End of rqstmain_wait_reply

			0004 00000	.ENTRY	RQSTMAIN_WAIT_REPLY, Save R2	: 0424
			CF 9E 00002	MOVAB	MB_IOSB, R2	
	C2	52	0000'	CALLS	#0, RQSTMAIN_SETMODE_QIO	: 0463
			00 FB 00007	CLRQ	-(SP)	: 0467
			7E 7C 0000B	CLRQ	-(SP)	
			7E 7C 0000D	MOVZWL	#256, -(SP)	
		7E	0100	PUSHAB	MB_BUFFER	
			08	CLRQ	-(SP)	
			A2 9F 00014	PUSHL	R2	
			7E 7C 00017	PUSHL	#49	
			52 DD 00019	PUSHL	MB_CHAN	
			31 DD 0001B	PUSHL	#2	
			FC A2 DD 0001D	CALLS	#12, SYS\$QIOUW	
			02 DD 00020	BLBC	STATUS, 1\$	: 0468
	00000000G	00	0C FB 00022	MOVZWL	MB_IOSB, STATUS	: 0470
		06	50 E9 00029	BLBS	STATUS, 2\$	: 0471
		50	62 3C 0002C	PUSHL	STATUS	: 0473
		0A	50 E8 0002F	CALLS	#1, LIB\$STOP	
	00000000G	00	50 DD 00032 1\$:	RET		
			01 FB 00034	PUSHAB	MB_BUFFER+8	: 0477
			04 0003B	MOVZWL	MB_IOSB+2, -(SP)	
		10	A2 9F 0003C 2\$:	SUBL2	#8, (SP)	
		02	A2 3C 0003F	PUSHL	#2	
			08 C2 00043	PUSHL	#360593	
			02 DD 00046	CALLS	#4, LIB\$SIGNAL	
			8F DD 00048	CMPW	MB_BUFFER+2, #32801	: C481
	00000000G	00	04 FB 0004E	BEQL	3\$	: 0488
	8021	8F	A2 B1 00055	MOVW	#4101, MB_BUFFER+4	: 0489
			0B 13 0005B	MOVL	MB_BUFFER+2, R0	
			8F B0 0005D	RET		
	0C	A2	1005	CLR		
		50	0A	RET		
			A2 D0 00063			
			04 00067			
			50 D4 00068 3\$:			
			04 0006A			

; Routine Size: 107 bytes, Routine Base: \$CODE\$ + 0283

OPC\$RQSTMAIN
V04-000

REQUEST command main module
rqstmain_wait_reply routine

C 13
16-Sep-1984 01:47:25
14-Sep-1984 12:50:54

VAX-11 Bliss-32 V4.0-742
[OPCOM.SRC]RQSYMAIN.B32;1

Page 18
(8)

: 499
: 500
0492 1 END
0493 0 ELUDOM

! End of rqstmain

.EXTRN LIB\$SIGNAL

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	1000	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$PLITS	62	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODES	750	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	32	0	1000	00:01.9
_\$255\$DUA28:[OPCOM.OBJ]OPCOMLIB.L32;1	633	19	3	43	00:00.9

COMMAND QUALIFIERS

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

: Size: 750 code + 1062 data bytes
: Run Time: 00:17.1
: Elapsed Time: 01:03.5
: Lines/CPU Min: 1733
: Lexemes/CPU-Min: 22245
: Memory Used: 137 pages
: Compilation Complete

0291 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

REPLYBRO
LIS

SECURITY
LIS

OPRENABLE
LIS

REPLYMAIN
LIS

ROSTMAIN
LIS