

1

: R

.....

•
•
•
•
•
•
•

—

```

000000  PPPPPPP  CCCCCCCC  000000  MM      MM  IIIIII  NN      NN  IIIIII
000000  PPPPPPP  CCCCCCCC  000000  MM      MM  IIIIII  NN      NN  IIIIII
00      00  PP      PP  CC      00      00  MMMM  MMMM  II      NN      NN  II
00      00  PP      PP  CC      00      00  MMMM  MMMM  II      NN      NN  II
00      00  PP      PP  CC      00      00  MM  MM  MM  II      NNNN  NN  II
00      00  PP      PP  CC      00      00  MM  MM  MM  II      NNNN  NN  II
00      00  PPPPPPP  CC      00      00  MM  MM  MM  II      NN  NN  NN  II
00      00  PPPPPPP  CC      00      00  MM  MM  MM  II      NN  NN  NN  II
00      00  PP      CC      00      00  MM  MM  MM  II      NN      NNNN  II
00      00  PP      CC      00      00  MM  MM  MM  II      NN      NNNN  II
00      00  PP      CC      00      00  MM  MM  MM  II      NN      NN  NN  II
00      00  PP      CC      00      00  MM  MM  MM  II      NN      NN  NN  II
00      00  PP      CC      00      00  MM  MM  MM  II      NN      NN  NN  II
00      00  PP      CC      00      00  MM  MM  MM  II      NN      NN  NN  II
000000  PP      CCCCCCCC  000000  MM      MM  IIIIII  NN      NN  IIIIII
000000  PP      CCCCCCCC  000000  MM      MM  IIIIII  NN      NN  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      II      SS
LLLLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLLLL  IIIIII  SSSSSSSS

```



```
1 0001 0 MODULE OPCSOPCOMINIT (
2 0002 0 LANGUAGE (BLISS32),
3 0003 0 IDENT = 'V04-000'
4 0004 0 ) =
5 0005 0
6 0006 0 *****
7 0007 0 *
8 0008 0 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 0 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 0 * ALL RIGHTS RESERVED.
11 0011 0 *
12 0012 0 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 0 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 0 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 0 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 0 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 0 * TRANSFERRED.
18 0018 0 *
19 0019 0 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 0 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 0 * CORPORATION.
22 0022 0 *
23 0023 0 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 0 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 0 *
26 0026 0 *
27 0027 0 *****
28 0028 0
29 0029 0 ++
30 0030 0 FACILITY:
31 0031 0
32 0032 0 OPCOM
33 0033 0
34 0034 0 ABSTRACT:
35 0035 0
36 0036 0 This module contains the logic to start and initialize OPCOM.
37 0037 0
38 0038 0 Environment:
39 0039 0
40 0040 0 VAX/VMS operating system.
41 0041 0
42 0042 0 Author:
43 0043 0
44 0044 0 Steven T. Jeffreys
45 0045 0
46 0046 0 Creation date:
47 0047 0
48 0048 0 March 10, 1981
49 0049 0
50 0050 0 Revision history:
51 0051 0
52 0052 0 V03-009 CWH3169 CW Hobbs 5-May-1984
53 0053 0 Second pass for cluster-wide OPCOM:
54 0054 0 - When we come up, ask for ACKs from any nodes that we can
55 0055 0 see.
56 0056 0 - Use SYIS$NODENAME rather than SCSNODE so that a useful
57 0057 0 length is returned
```

```

: 58      0058 0  |
: 59      0059 0  |
: 60      0060 0  |
: 61      0061 0  |
: 62      0062 0  |
: 63      0063 0  |
: 64      0064 0  |
: 65      0065 0  |
: 66      0066 0  |
: 67      0067 0  |
: 68      0068 0  |
: 69      0069 0  |
: 70      0070 0  |
: 71      0071 0  |
: 72      0072 0  |
: 73      0073 0  |
: 74      0074 0  |
: 75      0075 1  |
: 76      0076 1  |
: 77      0077 1  |
: 78      0078 1  |
: 79      0079 1  |
: 80      0080 1  |
: 81      0081 1  |
: 82      0082 1  |
: 83      0083 1  |
: 84      0084 1  |
: 85      0085 1  |
: 86      0086 1  |

V03-003 CWH3003      CW Hobbs      14-Apr-1984
Change SCSNODE syi items to the single item SYIS_SCSNODE.

V03-002 CWH3002      CW Hobbs      16-Sep-1983
Use jacket routines for VM calls

V03-001 CWH3001      CW Hobbs      30-Jul-1983
Various and sundry things to make OPCOM distributed
across the cluster.

V02-002 STJ0162      Steven T. Jeffreys, 08-Feb-1982
Make references to library routines use general addressing mode.

--

BEGIN                                ! Start of OPCOMMAIN

LIBRARY 'SYSS$LIBRARY:LIB.L32';
LIBRARY 'LIB$:OPCOMLIB';

FORWARD ROUTINE
OPCOM_INIT      : NOVALUE,      ! Initialization routine
KERNEL_SETUP;   ! Kernel mode setup

BUILTIN
INSQUE,          ! Insert entry onto a queue
REMQUE;          ! Remove entry from a queue

```



```
88 0087 1 GLOBAL ROUTINE OPCOM_INIT : NOVALUE =
89 0088 1
90 0089 1 ++
91 0090 1 Functional description:
92 0091 1
93 0092 1 This routine will start OPCOM and perform all of
94 0093 1 the necessary initialization to ensure OPCOM's
95 0094 1 correct functioning.
96 0095 1
97 0096 1 Input:
98 0097 1
99 0098 1 None.
100 0099 1
101 0100 1 Implicit Input:
102 0101 1
103 0102 1 The operator mailbox exists and is available.
104 0103 1 The process has CHMKRNL privilege.
105 0104 1
106 0105 1 Output:
107 0106 1
108 0107 1 None.
109 0108 1
110 0109 1 Implicit output:
111 0110 1
112 0111 1 None.
113 0112 1
114 0113 1 Side effects:
115 0114 1
116 0115 1 The operator logfile will be created on the system disk.
117 0116 1
118 0117 1 Routine value:
119 0118 1
120 0119 1 SSS_NOPRIV - The process does not have CHKRNL privilege.
121 0120 1 SSS_DEVALLOC - The operator mailbox already has a listener.
122 0121 1 SSS_NORMAL - OPCOM successfully completed its shutdown sequence.
123 0122 1 --
124 0123 1
125 0124 2 BEGIN ! Start of OPCOM_INIT
126 0125 2
127 0126 2 EXTERNAL ROUTINE
128 0127 2 CLUSMSG_ACK PLEASE : NOVALUE, ! Request an ACK from another node
129 0128 2 CLUSUTIC_CONFIGURE, ! Cluster configuration
130 0129 2 CLUSUTIL_INIT : NOVALUE, ! Cluster initializations
131 0130 2 CLUSUTIL_NODE MESSAGE : NOVALUE, ! Tell about cluster initializations
132 0131 2 OPRENABLE_HANDLER: NOVALUE, ! Enable operator terminal
133 0132 2 LOGFILE_HANDLER : NOVALUE, ! Initialize system logfile
134 0133 2 SHARE_INIT_OPER_NAME : NOVALUE, ! Initialize the operator name vector
135 0134 2 TIME_STAMP : NOVALUE; ! Issue periodic timestamp
136 0135 2
137 0136 2 EXTERNAL LITERAL
138 0137 2 OCD_K_TYPE, ! OCD structure type
139 0138 2 MIN_SCOPE, ! Min scope value
140 0139 2 MAX_SCOPE, ! Max scope value
141 0140 2 MIN_DS_TYPE, ! Min DS_TYPE value
142 0141 2 MAX_DS_TYPE; ! Max DS_TYPE value
143 0142 2
144 0143 2 EXTERNAL
```

```
145 0144 2 SYI_SWPOUTPGCNT : LONG,
146 0145 222 GLOBAL_STATUS : BITVECTOR,      ! Global status flags
147 0146 222 OCD_VECTOR : VECTOR,          ! Pointers to OCD's
148 0147 222 SCB_TABLE : VECTOR,          ! Start of SCB table
149 0148 222 WAIT_DELTA : $bblock,        ! Quadword AST wait time
150 0149 222 OPAO_ENABLE_MSG : $bblock,   ! Descriptor of enable message
151 0150 222 LOGFILE_MSG : $bblock,      ! Descriptor of INIT message
152 0151 222 OPER_MBX_CHAN : WORD,        ! Channel to MBA2:
153 0152 222 OPER_MBX_NAME : $bblock,     ! Quadword descriptor of 'MBA2:'
154 0153 222 LCL_NODENAME : VECTOR,      ! Name of local node (DECnet or VAXcluster)
155 0154 222 LCL_NOD : $ref_bblock,      ! Local node block
156 0155 222 LCL_CSID : LONG,            ! CSID for local node
157 0156 222 NOD_HEAD : VECTOR,          ! Head of node queue
158 0157 222 LOGFILE_FAB : $ref_bblock,  ! RMS control structure
159 0158 222 LOGFILE_RAB : $ref_bblock,  ! RMS control structure
160 0159 222 LOGFILE_OPEN : LONG;        ! Global boolean
161 0160
162 0161 222 LOCAL
163 0162 222 IOSB : $bblock [8],          ! I/O status block
164 0163 222 SIZE : LONG,                ! Size of data structure
165 0164 222 TYPE : LONG,                ! Type of data structure
166 0165 222 BLOCK : $ref_bblock,        ! VM block
167 0166 222 SCB : $ref_bblock,          ! SCB block
168 0167 222 PTR : LONG;
169 0168 222 STATUS : LONG;
170 0169
171 0170 222 OWN
172 0171 222 NODENAME_BUF : VECTOR [16, BYTE],
173 0172 222 NODENAME_DESC : VECTOR [2, LONG] INITIAL (0, NODENAME_BUF),
174 0173 222 TRANLOG_DESC : VECTOR [2, LONG] INITIAL (16, NODENAME_BUF),
175 0174 222 SYI_ITEMS : VECTOR [7, LONG] PRESET (
176 0175 222 [0] = (SYI$NODENAME*16 OR 16),
177 0176 222 [1] = NODENAME_BUF,
178 0177 222 [2] = NODENAME_DESC [0],
179 0178 222 [3] = (SYI$SWPOUTPGCNT*16 OR 4),
180 0179 222 [4] = SYI_SWPOUTPGCNT,
181 0180 222 [5] = 0,
182 0181 222 [6] = 0);
183 0182
184 0183 222
185 0184 222 ! Assign a channel to the operator mailbox, MBA2:. The mailbox is
186 0185 222 ! defined by the EXEC module DEVICEDAT, and has an initial reference
187 0186 222 ! count of 1. This means that no one can allocate the device, no one
188 0187 222 ! can delete it, and that once we assign a channel to it, the reference
189 0188 222 ! count had better be 2. If the $ASSIGN fails, then give up.
190 0189 222
191 0190 222 IF NOT (STATUS = $ASSIGN (CHAN=OPER_MBX_CHAN, DEVNAM=OPER_MBX_NAME))
192 0191 222 THEN
193 0192 222 $signal_stop (.STATUS);
194 0193 222
195 0194 222 ! We now have a channel to the operator mailbox. Now we must go into kernel
196 0195 222 ! mode to do some further checking. We must guaranty that we have CHMKRNL
197 0196 222 ! privilege, and that the reference count on the operator mailbox is equal to
198 0197 222 ! 2. While in kernel mode, also set the OPR bit in the default operator UCB,
199 0198 222 ! _OPAO:.
200 0199 222
201 0200 3 IF NOT (STATUS = $CMKRNL (ROUTIN=KERNEL_SETUP))
```



```
202 0201 2 THEN
203 0202 2
204 0203 2
205 0204 2
206 0205 2
207 0206 2
208 0207 2
209 0208 2
210 0209 2
211 0210 2
212 0211 2
213 0212 2
214 0213 2
215 0214 2
216 0215 2
217 0216 2
218 0217 2
219 0218 2
220 0219 2
221 0220 2
222 0221 2
223 0222 2
224 0223 2
225 0224 2
226 0225 2
227 0226 2
228 0227 2
229 0228 2
230 0229 2
231 0230 2
232 0231 2
233 0232 2
234 0233 2
235 0234 2
236 0235 2
237 0236 2
238 0237 2
239 0238 2
240 0239 2
241 0240 2
242 0241 2
243 0242 2
244 0243 2
245 0244 2
246 0245 2
247 0246 2
248 0247 2
249 0248 2
250 0249 2
251 0250 2
252 0251 2
253 0252 2
254 0253 2
255 0254 2
256 0255 2
257 0256 2
258 0257 2

    THEN
    !
    ! Something is wrong. The only error that we can tolerate is the
    ! nonexistence of OPA0:, which is indicated by a status of SS$_NOSUCHDEV.
    ! Any other error condition will force us to abort.
    IF NOT (.STATUS EQL SS$_NOSUCHDEV)
    THEN
        $signal_stop (.STATUS);

    !
    ! Initialize the vector of operator names
    SHARE_INIT_OPER_NAME ();

    !
    ! Build the look-aside lists for all data structures.
    INCR J FROM MIN_DS_TYPE TO MAX_DS_TYPE DO
    BEGIN
        SCB = .SCB_TABLE [J-1];          ! Get the SCB address
        SIZE = .SCB [SCB_W_SIZE];        ! Get data structure size
        TYPE = J;                        ! Get data structure type
        INCR I FROM 1 TO .SCB [SCB_W_LAL_COUNT] DO
        BEGIN
            !
            ! Allocate a block of memory, initialize it, and
            ! put it on the appropriate look-aside list.
            IF NOT (STATUS = OPC$GET_VM (SIZE, BLOCK))
            THEN
                $signal_stop (.STATUS);
                CH$FILL (0, .SIZE, .BLOCK); ! Zero the block
                BLOCK [HDR_L_FLINK] = BLOCK [HDR_L_FLINK];
                BLOCK [HDR_L_BLINK] = BLOCK [HDR_L_FLINK];
                BLOCK [HDR_W_SIZE] = .SIZE; ! Set block size
                BLOCK [HDR_B_TYPE] = .TYPE; ! Set block type
                BLOCK [HDR_V_LAL] = TRUE;   ! Mark this as an LAL block
                INSQUE (BLOCK [HDR_L_FLINK], SCB [SCB_L_FLINK]);
            END;
        END;
    END;

    !
    ! Do a $GETSYI to get information about the current system
    IF NOT (STATUS = $GETSYI (ITMLST=SYI_ITEMS))
    THEN
        $signal_stop (.STATUS);

    !
    ! If the SCS nodename is null, try to translate SYSS$NODE to find the DECnet name. Remove the "_" and "':"
    ! from the translated name.
    IF .NODENAME_DESC [0] EQL 0
    THEN
        BEGIN
            IF NOT (STATUS = $TRNLOG (LOGNAM=%ASCII 'SYSS$NODE', RSLLEN=TRANLOG_DESC, RSLBUF=TRANLOG_DESC, DSBMSK=6))
            THEN
```

```
259 0258 3 $signal_stop (.STATUS);
260 0259 3 IF .STATUS EQL $$$_NORMAL ! If we translated, remove the underscores and colons
261 0260 3 THEN
262 0261 4 BEGIN
263 0262 4 PTR = CH$FIND_NOT_CH (.TRANLOG_DESC [0], .TRANLOG_DESC [1], %C '_');
264 0263 4 IF .PTR NEQ 0
265 0264 4 THEN
266 0265 5 BEGIN
267 0266 5 TRANLOG_DESC [0] = .TRANLOG_DESC [0] - (.PTR - .TRANLOG_DESC [1]);
268 0267 5 TRANLOG_DESC [1] = .PTR;
269 0268 4 END;
270 0269 4 PTR = CH$FIND_CH (.TRANLOG_DESC [0], .TRANLOG_DESC [1], %C ':');
271 0270 4 IF .PTR NEQ 0
272 0271 4 THEN
273 0272 4 TRANLOG_DESC [0] = .PTR - .TRANLOG_DESC [1];
274 0273 4 NODENAME_DESC [0] = .TRANLOG_DESC [0];
275 0274 4 NODENAME_DESC [1] = .TRANLOG_DESC [1];
276 0275 3 END;
277 0276 2 END;
278 0277 2
279 0278 2 | Set the global message
280 0279 2
281 0280 2 IF .NODENAME_DESC [0] NEQ 0
282 0281 2 THEN
283 0282 3 BEGIN
284 0283 3 LCL_NODENAME [0] = .NODENAME_DESC [0];
285 0284 3 LCL_NODENAME [1] = .NODENAME_DESC [1];
286 0285 3 END;
287 0286 2
288 0287 2 | Perform cluster initialization functions necessary before can enable
289 0288 2 | operators.
290 0289 2
291 0290 2 CLUSUTIL_INIT ();
292 0291 2
293 0292 2
294 0293 2 | Set up _OPA0: as the default operator. This is done by calling the enable handler with a the
295 0294 2 | address of a buffer descriptor that points to a predefined enable message.
296 0295 2
297 0296 2 OPRENABLE_HANDLER (OPA0_ENABLE_MSG);
298 0297 2
299 0298 2
300 0299 2 | Open the log file. This is done by calling the INIT message handler with the address of a buffer
301 0300 2 | descriptor that points to a predefined INIT message. Set logging enabled.
302 0301 2
303 0302 2 LOGFILE_HANDLER (LOGFILE_MSG);
304 0303 2 GLOBAL_STATUS [GBLSTS_K_LOGGING_ENABLED] = TRUE;
305 0304 2
306 0305 2
307 0306 2 | Configure the nodes into the cluster database, and notify any
308 0307 2 | operators. This should be done after OPA0 and the logfile are ready.
309 0308 2
310 0309 2 IF .GLOBAL_STATUS [GBLSTS_K_IN_VAXcluster]
311 0310 2 THEN
312 0311 3 BEGIN
313 0312 3 CLUSUTIL_NODE_MESSAGE (.LCL_NOD, OPCS_NODE_INIT, FALSE); ! Tell them we have initialized the local no
314 0313 3 IF CLUSUTIL_CONFIGURE () ! Now bring in everybody else, true if anyon
315 0314 3 THEN ! has been added to the club
```



```

316      BEGIN
317      LOCAL
318      NOD : $REF BBLOCK;
319      NOD = .NOD_HEAD [0];
320      WHILE .NOD-NEQ NOD_HEAD [0]
321      DO
322      BEGIN
323      IF .NOD [NOD_L_NODE_CSID] NEQ .LCL_CSID
324      THEN
325      CLUSMSG ACK_PLEASE (.NOD);
326      NOD = .NOD [NOD_L_FLINK];
327      END;
328      END;
329      END;
330      Set the operator mailbox protection to
331      allow OWNER reads and writes.
332      IF NOT (STATUS = $QIO (FUNC = (IOS$ SETMODE OR IOSM_SETPROT),
333      P  CHAN = .OPER_MBX_CHAN,
334      P  IOSB = IOSB,
335      P  P2 = READ_WRITE
336      P  ))
337      THEN
338      $signal_stop (.STATUS);
339      Check the function status.
340      IF NOT (STATUS = .IOSB [0,0,16,0])
341      THEN
342      $signal_stop (.STATUS);
343      Set the timer ast for the first time stamp message.
344      Also set the associated GLOBAL_STATUS bits.
345      GLOBAL_STATUS [GBLSTS_K_BUSY] = TRUE;
346      GLOBAL_STATUS [GBLSTS_K_TIMESTAMP_PENDING] = FALSE;
347      IF NOT (STATUS = $SETIMR (EFN = EFN_K_TIME_STAMP, DAYTIM = WAIT_DELTA, ASTADR = TIME_STAMP))
348      THEN
349      $signal_stop (.STATUS);
350      END;
351      ! End of OPCOM_INIT

```

```

.TITLE OPC$OPCOMINIT
.IDENT \V04-000\

.PSECT $PLITS$,NOWRT,NOEXE,2

.ASCII \SYS$NODE\
.LONG 17694728
.ADDRESS P.AAB

.PSECT $OWNS$,NOEXE,2

```

```

45 44 4F 4E 24 53 59 53 00000 P.AAB:
010E0008 00008 P.AAA:
00000000 0000C

```

```

00000 NODENAME_BUF:
00000000 00010 NODENAME_DESC:
00000000' 00014 .LONG 0
00000010 00018 TRANLOG_DESC:
00000000' 0001C .LONG 16
10D90010 00020 SYI_ITEMS:
00000000' 00000000' 00024 .ADDRESS NODENAME_BUF, NODENAME_DESC
103B0004 0002C .LONG 272302084
00000000G 00030 .ADDRESS SYI_SWPOUTPGCNT
00000000 00000000 00034 .LONG 0, 0

.EXTRN CLUSMSG ACK PLEASE
.EXTRN CLUSUTIC_CONFIGURE
.EXTRN CLUSUTIL_INIT, CLUSUTIL_NODE_MESSAGE
.EXTRN OPRENABLE_HANDLER
.EXTRN LOGFILE_HANDLER
.EXTRN SHARE_INIT_OPER_NAME
.EXTRN TIME_STAMP, OCD_K_TYPE
.EXTRN MIN_SCOPE, MAX_SCOPE
.EXTRN MIN_DS_TYPE, MAX_DS_TYPE
.EXTRN SYI_SWPOUTPGCNT
.EXTRN GLOBAL_STATUS, OCD_VECTOR
.EXTRN SCB_TABLE, WAIT_DELTA
.EXTRN OPAO_ENABLE_MSG
.EXTRN LOGFILE_MSG, OPER_MBX_CHAN
.EXTRN OPER_MBX_NAME, LCC_NODENAME
.EXTRN LCL_NOD, LCL_CSID
.EXTRN NOD_HEAD, LOGFILE_FAB
.EXTRN LOGFILE_RAB, LOGFILE_OPEN
.EXTRN SYSS$ASSIGN, LIB$STOP
.EXTRN SYSS$CMKRNL, OPC$GET_VM
.EXTRN SYSS$GETSYI, SYSS$TRNLOG
.EXTRN SYSS$QIO, SYSS$SETIMR

.PSECT $CODE$,NOWRT,2

OFFC 00000
SE 14 C2 00002
7E 7C 00005
0000G CF 9F 00007
0000G CF 9F 0000B
00000000G 00 04 FB 0000F
59 50 D0 00016
1C 59 E9 00019
7E D4 0001C
0000V CF 9F 0001E
02 FB 00022
00000000G 00 50 D0 00029
59 59 E8 0002C
OC 59 D1 0002F
00000908 8F 03 13 00036
01AF 31 00038 1$:

.SUBL2 #20, SP
CLRQ -(SP)
PUSHAB OPER_MBX_CHAN
PUSHAB OPER_MBX_NAME
CALLS #4, SYSS$ASSIGN
MOVL R0, STATUS
BLBC STATUS, 1$
CLRL -(SP)
PUSHAB KERNEL_SETUP
CALLS #2, SYSS$CMKRNL
MOVL R0, STATUS
BLBS STATUS, 2$
CMPL STATUS, #2312
BEQL 2$
BRW 18$

```


58	0000G	CF	00	FB	0003B	2\$:	CALLS	#0, SHARE_INIT_OPER_NAME	0214
	00000000G	8F	01	C3	00040		SUBL3	#1, #MIN_DS_TYPE, J	0219
			50	11	00048		BRB	6\$	
	08	56	0000G	CF	48	3\$:	MOVL	SCB TABLE-4[J], SCB	0221
		AE		66	3C		MOVZWL	(SCB), SIZE	0222
		6E		58	D0		MOVL	J, TYPE	0223
		5B	02	A6	3C		MOVZWL	2(SCB), R11	0224
				5A	D4		CLRL	I	
				37	11		BRB	5\$	
			04	AE	9F	4\$:	PUSHAB	BLOCK	0230
			0C	AE	9F		PUSHAB	SIZE	
	0000G	CF		02	FB		CALLS	#2, OPC\$GET_VM	
		59		50	D0		MOVL	R0, STATUS	
		C8		59	E9		BLBC	STATUS, 1\$	
08	AE	57	04	AE	D0		MOVL	BLOCK, R7	0233
		6E		00	2C		MOVC5	#0, (SP), #0, SIZE, (R7)	
				67					
		67		57	D0		MOVL	R7, (R7)	0234
	04	A7		57	D0		MOVL	R7, 4(R7)	0235
	08	A7	08	AE	B0		MOVW	SIZE, 8(R7)	0236
	0A	A7		6E	90		MOVB	TYPE, 10(R7)	0237
	28	A7		01	88		BISB2	#1, 40(R7)	0238
		50	08	A6	9E		MOVAB	8(R6), R0	0239
		60		67	0E		INSQUE	(R7), (R0)	
C5		5A		5B	F3	5\$:	AOBLEQ	R11, I, 4\$	0224
A8		58	00000000G	8F	F3	6\$:	AOBLEQ	#MAX_DS_TYPE, J, 3\$	0219
				7E	7C		CLRQ	-(SP)	0246
				7E	D4		CLRL	-(SP)	
			0000'	CF	9F		PUSHAB	SYI ITEMS	
				7E	7C		CLRQ	-(SP)	
				7E	D4		CLRL	-(SP)	
	00000000G	00		07	FB		CALLS	#7, SYS\$GETSYI	
		59		50	D0		MOVL	R0, STATUS	
		20		59	E9		BLBC	STATUS, 7\$	
			0000'	CF	D5		TSTL	NODENAME_DESC	0253
				65	12		BNEQ	13\$	
				06	DD		PUSHL	#6	0256
				7E	7C		CLRQ	-(SP)	
			0000'	CF	9F		PUSHAB	TRANLOG_DESC	
			0000'	CF	9F		PUSHAB	TRANLOG_DESC	
			0000'	CF	9F		PUSHAB	P.AAA	
	00000000G	00		06	FB		CALLS	#6, SYS\$TRNLOG	
		59		50	D0		MOVL	R0, STATUS	
		03		59	E8	7\$:	BLBS	STATUS, 8\$	
			01	0109	31		BRW	18\$	
				59	D1	8\$:	CMPL	STATUS, #1	0259
				40	12		BNEQ	13\$	
0000'	DF	0000'	CF	5F	8F		SKPC	#95, TRANLOG_DESC, @TRANLOG_DESC+4	0262
					02		BNEQ	9\$	
					51		CLRL	R1	
					51	9\$:	TSTL	PTR	0263
					10		BEQL	10\$	
	50	0000'	CF	51	C3		SUBL3	PTR, TRANLOG_DESC+4, R0	0266
		0000'	CF	50	C0		ADDL2	R0, TRANLOG_DESC	
		0000'	CF	51	D0		MOVL	PTR, TRANLOG_DESC+4	0267
0000'	DF	0000'	CF	3A	3A	10\$:	LOCC	#58, TRANLOG_DESC, @TRANLOG_DESC+4	0269
				02	12		BNEQ	11\$	

0000' CF	51	0000'	51	D4	00111	CLRL	R1		
			51	D5	00113	11\$: TSTL	PTR		0270
			08	13	00115	BEQL	12\$		
	0000' CF	0000'	CF	C3	00117	SUBL3	TRANLOG_DESC+4, PTR, TRANLOG_DESC		0272
	50	0000'	CF	7D	0011F	12\$: MOVQ	TRANLOG_DESC, NODENAME_DESC		0273
			CF	D0	00126	13\$: MOVL	NODENAME_DESC, R0		0280
			0C	13	00128	BEQL	14\$		
	0000G CF		50	D0	0012D	MOVL	R0, LCL_NODENAME		0283
	0000G CF	0000'	CF	D0	00132	MOVL	NODENAME_DESC+4, LCL_NODENAME+4		0284
	0000G CF		00	FB	00139	14\$: CALLS	#0, CLUSUTIL_INIT		0290
			CF	9F	0013E	PUSHAB	OPAO_ENABLE_MSG		0296
	0000G CF		01	FB	00142	CALLS	#1, OPENABLE_HANDLER		
			CF	9F	00147	PUSHAB	LOGFILE_MSG		0302
	0000G CF		01	FB	0014B	CALLS	#1, LOGFILE_HANDLER		
	0000G CF		02	88	00150	BISB2	#2, GLOBAL_STATUS		0303
			3C	0000G CF	E9	00155	BLBC	GLOBAL_STATUS+1, 17\$	0309
				7E	D4	0015A	CLRL	-(SP)	0312
		00058233		8F	DD	0015C	PUSHL	#361011	
		0000G		CF	DD	00162	PUSHL	LCL_NOD	
	0000G CF		03	FB	00166	CALLS	#3, CLUSUTIL_NODE_MESSAGE		
	0000G CF		00	FB	0016B	CALLS	#0, CLUSUTIL_CONFIGURE		0313
			23	50	E9	00170	BLBC	R0, 17\$	
			52	0000G CF	D0	00173	MOVL	NOD_HEAD, NOD	0318
			50	0000G CF	9E	00178	15\$: MOVAB	NOD_HEAD, R0	0319
			50	52	D1	0017D	CMPL	NOD, R0	
				14	13	00180	BEQL	17\$	
	0000G CF	2C	A2	D1	00182	CMPL	44(NOD), LCL_CSID		0322
			07	13	00188	BEQL	16\$		
			52	DD	0018A	PUSHL	NOD		0324
	0000G CF		01	FB	0018C	CALLS	#1, CLUSMSG_ACK_PLEASE		
			52	62	D0	00191	16\$: MOVL	(NOD), NOD	0325
				E2	11	00194	BRB	15\$	0319
				7E	7C	00196	17\$: CLRQ	-(SP)	0337
				7E	7C	00198	CLRQ	-(SP)	
			7E	8F	3C	0019A	MOVZWL	#65295, -(SP)	
				7E	7C	0019F	CLRQ	-(SP)	
				7E	D4	001A1	CLRL	-(SP)	
				AE	9F	001A3	PUSHAB	IOSB	
		2C		8F	3C	001A6	MOVZWL	#547, -(SP)	
		0223		CF	3C	001AB	MOVZWL	OPER_MBX_CHAN, -(SP)	
		0000G		7E	D4	001B0	CLRL	-(SP)	
	00000000G		00	0C	FB	001B2	CALLS	#12, SYSSQIO	
			59	50	D0	001B9	MOVL	R0, STATUS	
			2B	59	E9	001BC	BLBC	STATUS, 18\$	
			59	AE	3C	001BF	MOVZWL	IOSB, STATUS	0344
			24	59	E9	001C3	BLBC	STATUS, 18\$	
	0000G CF	40	8F	88	001C6	BISB2	#64, GLOBAL_STATUS		0352
	0000G CF		20	8A	001CC	BICB2	#32, GLOBAL_STATUS		0353
			7E	D4	001D1	CLRL	-(SP)		0354
				CF	9F	001D3	PUSHAB	TIME_STAMP	
		0000G		CF	9F	001D7	PUSHAB	WAIT_DELTA	
		0000G		04	DD	001DB	PUSHL	#4	
	00000000G		00	04	FB	001DD	CALLS	#4, SYSSSETIMR	
			59	50	D0	001E4	MOVL	R0, STATUS	
			09	59	E8	001E7	BLBS	STATUS, 19\$	
				59	DD	001EA	18\$: PUSHL	STATUS	0356
	00000000G		00	01	FB	001EC	CALLS	#1, LIB\$STOP	

OPC\$OPCOMINIT
V04-000

D 7
16-Sep-1984 01:32:26
14-Sep-1984 12:50:47

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

Page 11
(2)

04 001F3 19\$: RET

; 0358

; Routine Size: 500 bytes, Routine Base: \$CODE\$ + 0000

; 360 0359 1

```

362 0360 1 ROUTINE KERNEL_SETUP =
363 0361 1
364 0362 1 !++
365 0363 1 Functional description:
366 0364 1
367 0365 1 This routine executes in kernel mode, and performs some
368 0366 1 additional setup necessary for the correct operation of OPCOM.
369 0367 1
370 0368 1 Input:
371 0369 1
372 0370 1 None.
373 0371 1
374 0372 1 Implicit Input:
375 0373 1
376 0374 1 A channel has been assigned to _MBA2:.
377 0375 1
378 0376 1 Output:
379 0377 1
380 0378 1 None.
381 0379 1
382 0380 1 Implicit output:
383 0381 1
384 0382 1 If the _OPA0: exists, then the OPR bit in its UCB is set.
385 0383 1
386 0384 1 Side effects:
387 0385 1
388 0386 1 None.
389 0387 1
390 0388 1 Routine value:
391 0389 1
392 0390 1 SSS_NORMAL if everything worked.
393 0391 1 SSS_DEVALLOC if _MBA2: has a reference count greater than 2.
394 0392 1 SSS_NOSUCHDEV if _OPA0: does not exist.
395 0393 1 --
396 0394 1
397 0395 2 BEGIN ! Start of KERNEL_SETUP
398 0396 2
399 0397 2 EXTERNAL
400 0398 2 SYSSGL_OPRMBX : $bblock ADDRESSING_MODE (GENERAL); ! Address of MBA2 UCB
401 0399 2
402 0400 2 OWN
403 0401 2 DEF_OPER_DEV : $string_desc ('_OPA0:'); ! Default operator device name
404 0402 2
405 0403 2 LOCAL
406 0404 2 STATUS : LONG;
407 0405 2
408 0406 2
409 0407 2 Check the reference count in the MBA2: UCB. If it is greater than 2,
410 0408 2 it means that there is already a listener at the mailbox.
411 0409 2
412 0410 2 IF .SYSSGL_OPRMBX [UCB$W_REFC] GTR 2
413 0411 2 THEN
414 0412 2 RETURN (SSS_DEVALLOC);
415 0413 2
416 0414 2
417 0415 2 Set the OPR bit in the default operator UCB.
418 0416 2

```


: 419 0417 2 RETURN (EXES\$SETOPR (DEF_OPER_DEV,ON));
: 420 0418 1 END;

! End of KERNEL_SETUP

.PSECT \$SPLITS,NOWRT,NOEXE,2

3A 30 41 50 4F 5F 00010 P.AAC: .ASCII _OPA0:\

.PSECT \$OWNS\$,NOEXE,2

0006 0003C DEF_OPER_DEV:
01 0E 0003E .WORD 6
00000000' 00040 .BYTE 14, 1
.ADDRESS P.AAC

.EXTRN SYS\$GL_OPRMBX, EXES\$SETOPR

.PSECT \$CODE\$,NOWRT,2

0000 00000 KERNEL_SETUP:

02 00000000G 00 B1 00002 .WORD Save nothing
06 1B 00009 CMPW SYS\$GL_OPRMBX+92, #2
50 0840 8F 3C 0000B BLEQU 1\$
04 00010 MOVZWL #2112, R0
01 DD 00011 1\$: RET
CF 9F 00013 PUSHL #1
00000000G 00 0000' CF 9F 00013 PUSHAB DEF_OPER_DEV
02 FB 00017 CALLS #2, EXES\$SETOPR
04 0001E RET

: 0360
: 0410
: 0412
: 0417
: 0418

; Routine Size: 31 bytes, Routine Base: \$CODE\$ + 01F4

: 421 0419 1
: 422 0420 1 END
: 423 0421 0 ELUDOM

! End of OPCOMINIT

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS\$	68 NOVEC, WRT, RD	,NOEXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)
\$SPLITS	22 NOVEC,NOWRT, RD	,NOEXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)
\$CODE\$	531 NOVEC,NOWRT, RD	, EXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)

Library Statistics

----- Symbols ----- Pages Processing

OPCSOPCOMINIT
V04-000

6 7
16-Sep-1984 01:32:26
14-Sep-1984 12:50:47

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

Page 14
(3)

File	Total	Loaded	Percent	Mapped	Time
\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	25	0	1000	00:01.9
_\$255\$DUA28:[OPCOM.OBJ]OPCOMLIB.L32;1	633	24	3	43	00:00.9

COMMAND QUALIFIERS

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

; Size: 531 code + 90 data bytes
; Run Time: 00:14.5
; Elapsed Time: 00:48.4
; Lines/CPU Min: 1744
; Lexemes/CPU-Min: 19379
; Memory Used: 183 pages
; Compilation Complete

0290 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

