

[illegible]

```

LL      000000      GGGGGGGG      EEEEEEEEEEE      VV      VV      EEEEEEEEEEE      NN      NN      TTTTTTTTTT
LL      000000      GGGGGGGG      EEEEEEEEEEE      VV      VV      EEEEEEEEEEE      NN      NN      TTTTTTTTTT
LL      00      00      GG      EE      VV      VV      EE      NN      NN      TT
LL      00      00      GG      EE      VV      VV      EE      NN      NN      TT
LL      00      00      GG      EE      VV      VV      EE      NNNN      NN      TT
LL      00      00      GG      EE      VV      VV      EE      NNNN      NN      TT
LL      00      00      GG      EEEEEEEE      VV      VV      EEEEEEEE      NN      NN      TT
LL      00      00      GG      EEEEEEEE      VV      VV      EEEEEEEE      NN      NN      TT
LL      00      00      GG      GG      GG      VV      VV      EE      NN      NN      TT
LL      00      00      GG      GG      GG      VV      VV      EE      NN      NN      TT
LL      00      00      GG      GG      GG      VV      VV      EE      NN      NN      TT
LL      00      00      GG      GG      GG      VV      VV      EE      NN      NN      TT
LL      00      00      GG      GG      GG      VV      VV      EE      NN      NN      TT
LL      00      00      GG      GG      GG      VV      VV      EE      NN      NN      TT
LLLLLLLLLLLL      000000      GGGGGG      EEEEEEEEEEE      VV      VV      EEEEEEEEEEE      NN      NN      TT
LLLLLLLLLLLL      000000      GGGGGG      EEEEEEEEEEE      VV      VV      EEEEEEEEEEE      NN      NN      TT

```



```

LL      IIIIII      SSSSSSSS
LL      IIIIII      SSSSSSSS
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 OPC$LOGEVENT (
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 all the various and sundry general
37 0037 0 purpose utility routines used by OPCOM's request handlers.
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 Steven T. Jeffreys
46 0046 0
47 0047 0 Creation date:
48 0048 0
49 0049 0 March 10, 1981
50 0050 0
51 0051 0 Revision history:
52 0052 0
53 0053 0 V03-005 CWH3169 CW Hobbs 5-May-1984
54 0054 0 Second pass for cluster-wide OPCOM:
55 0055 0 - Add an explanation to DUMP_LOG_FILE and WRITE_LOG_FILE
56 0056 0 messages so that users won't bother us unless something
57 0057 0 is really wrong.
```



```

58 0058 0
59 0059 0
60 0060 0 V03-004 RSH0117 R. Scott Hanna 14-Mar-1983
61 0061 0 LOG_MESSAGE / Increase local buffer size to OPC$K_MAXMESSAGE.
62 0062 0
63 0063 0 V03-003 CWH3003 CW Hobbs 16-Sep-1983
64 0064 0 Increase size of local buffer, print blank line as separate
65 0065 0 records instead of using <CR><LF>.
66 0066 0
67 0067 0 V03-002 CWH3001 CW Hobbs 30-Jul-1983
68 0068 0 Various and sundry things to make OPCOM distributed
69 0069 0 across the cluster.
70 0070 0 V03-001 STJ3032 Steven T. Jeffreys, 05-Oct-1982
71 0071 0 Set GBLSTS_K_FLUSH_PENDIND when writing to the logfile.
72 0072 0
73 0073 0 V02-002 STJ0160 Steven T. Jeffreys, 08-Feb-1982
74 0074 0 Jiggle the message size and pointer so that the 'bell'
75 0075 0 character on front of each message is not sent to the logfile.
76 0076 0
77 0077 0 !--
78 0078 0
79 0079 1 BEGIN ! Start of LOGEVENT
80 0080 1
81 0081 1 LIBRARY 'SYS$LIBRARY:LIB.L32';
82 0082 1 LIBRARY 'LIB$:OPCOMLIB';
83 0083 1
84 0084 1 FORWARD ROUTINE
85 0085 1 DUMP_LOG_FILE, ! Print a formatted dump of a buffer
86 0086 1 LOG_MESSAGE, ! Write a message to a log file
87 0087 1 LOG_MESSAGE_PUT, ! $PUT a single record of a message
88 0088 1 LOGEVENT_FAO_BUFFER, ! Local fao buffer routine
89 0089 1 WRITE_LOG_FILE; ! Write a string desc to a log file
90 0090 1
91 0091 1 BIND
92 0092 1 EXP1 = %ASCID 'OPCOM has noticed a condition which might be due to an internal error.',
93 0093 1 EXP2 = %ASCID 'It might also be explained by normal events, especially if nodes have just',
94 0094 1 EXP3 = %ASCID 'crashed or rebooted in a VAXcluster. Please bring this message to Digital's',
95 0095 1 EXP4 = %ASCID 'attention only if you are having problems with operator communications.';
96 0096 1
97 0097 1 MACRO
98 0098 1 PUT (DESC) = BEGIN
99 0099 1 BIND DSC = (DESC) : $bblock;
100 0100 1 LOGFILE_RAB [RAB$W_RSZ] = .DSC [DSC$W_LENGTH];
101 0101 1 LOGFILE_RAB [RAB$L_RBF] = .DSC [DSC$A_POINTER];
102 0102 1 $PUT (RAB = LOGFILE_RAB) ! Status is value of the block
103 0103 1 END %,
104 0104 1 PUT_EXPLANATION =
105 0105 1 BEGIN PUT (EXP1); PUT (EXP2); PUT (EXP3); PUT (EXP4); END
106 0106 1 %;
```



```

: 108 0107 1 GLOBAL ROUTINE DUMP_LOG_FILE (BUFF_DESC : $ref_bblock, ID_DESC : $ref_bblock) =
: 109 0108 1
: 110 0109 1
: 111 0110 1 ++
: 112 0111 1 Functional description:
: 113 0112 1
: 114 0113 1 This routine will write a formatted hex dump to the operator log file.
: 115 0114 1 **
: 116 0115 1 ** This routine will be enhanced later to support mutliple log files.
: 117 0116 1 **
: 118 0117 1 Input:
: 119 0118 1
: 120 0119 1 BUFF_DESC : Address of a string desc for buffer to be dumped
: 121 0120 1 ID_DESC : Explanatory text for dump
: 122 0121 1
: 123 0122 1 Implicit Input:
: 124 0123 1
: 125 0124 1 None.
: 126 0125 1
: 127 0126 1 Output:
: 128 0127 1
: 129 0128 1 None.
: 130 0129 1
: 131 0130 1 Implicit output:
: 132 0131 1
: 133 0132 1 None.
: 134 0133 1
: 135 0134 1 Side effects:
: 136 0135 1
: 137 0136 1 None.
: 138 0137 1
: 139 0138 1 Routine value:
: 140 0139 1
: 141 0140 1 TRUE : If success
: 142 0141 1 <anything else> : If the log attempt failed
: 143 0142 1 !--
: 144 0143 1
: 145 0144 2 BEGIN ! Start of DUMP_LOG_FILE
: 146 0145 2
: 147 0146 2 EXTERNAL
: 148 0147 2 GLOBAL STATUS : BITVECTOR, ! OPCOM global status flags
: 149 0148 2 LOGFILE_RAB : $bblock, ! RMS control structure
: 150 0149 2 LOGFILE_FAB : $bblock; ! RMS control structure
: 151 0150 2
: 152 0151 2 LOCAL
: 153 0152 2 BASE,
: 154 0153 2 LEFT,
: 155 0154 2 PTR,
: 156 0155 2 LCL_DESC : $ref_bblock,
: 157 0156 2 INTER : VECTOR [8, LONG],
: 158 0157 2 STATUS : LONG;
: 159 0158 2
: 160 0159 2 See if logging is enabled.
: 161 0160 2
: 162 0161 3 IF (NOT .GLOBAL STATUS [GBLSTS_K_LOGGING_ENABLED])
: 163 0162 3 OR (.GLOBAL STATUS [GBLSTS_K_LOGFILE_CLOSED])
: 164 0163 2 THEN

```

```
165 0164 2 RETURN (TRUE);
166 0165 2
167 0166 2 Format and print the message header
168 0167 2
169 0168 2 INTER [0] = 0; ! A blank line
170 0169 2 PUT (INTER);
171 0170 2 PUT (LOGEVENT_FAO_BUFFER (%ASCID 'XXXXXXXXX OPCOM !XD XXXXXXXXXXXX DUMP_LOG_FILE', 0));
172 0171 2 PUT_EXPLANATION;
173 P 0172 2 PUT (LOGEVENT_FAO_BUFFER (%ASCID 'Buffer is !5UL (%X!XW) bytes -- '!AS'',
174 0173 2 .BUFF_DESC [DSC$W_LENGTH], .BUFF_DESC [DSC$W_LENGTH], .ID_DESC));
175 0174 2
176 0175 2 Format the buffer, 32 bytes at a time
177 0176 2
178 0177 2 LEFT = .BUFF_DESC [DSC$W_LENGTH];
179 0178 2 PTR = .BUFF_DESC [DSC$A_POINTER];
180 0179 2 BASE = 0;
181 0180 2 WHILE .LEFT GTR 0
182 0181 2 DO
183 0182 2 BEGIN
184 0183 2
185 0184 2 Move the next chunk of data to the intermediate buffer
186 0185 2
187 0186 2 CH$COPY (MINU (.LEFT, 32), .PTR, 0, 32, INTER [0]);
188 P 0187 2 PUT (LOGEVENT_FAO_BUFFER (%ASCID '!8(9XL) !32AF !XW',
189 PP 0188 2 .INTER [7], .INTER [6], .INTER [5], .INTER [4],
190 P 0189 2 .INTER [3], .INTER [2], .INTER [1], .INTER [0],
191 0190 2 32, INTER [0], .BASE));
192 0191 2
193 0192 2 Move to the next chunk
194 0193 2
195 0194 2 BASE = .BASE + 32;
196 0195 2 PTR = .PTR + 32;
197 0196 2 LEFT = .LEFT - 32;
198 0197 2 END;
199 0198 2
200 0199 2 $FLUSH (RAB = LOGFILE_RAB);
201 0200 2
202 0201 2 RETURN (TRUE);
203 0202 1 END;
```

! End of DUMP_LOG_FILE

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

.PSECT \$PLITS,NOWRT,NOEXE,2

63	69	74	6F	6E	20	73	61	68	20	4D	4F	43	50	4F	00000	P.AAB:	.ASCII	\OPCOM has noticed a condition which migh\	:
20	6E	6F	69	74	69	64	6E	6F	63	20	61	20	64	65	0000F				:
					68	67	69	6D	20	68	63	69	68	77	0001E				:
20	6E	61	20	6F	74	20	65	75	64	20	65	62	20	74	00028		.ASCII	\t be due to an internal error.\<0><0>	:
2E	72	6F	72	72	65	20	6C	61	6E	72	65	74	6E	69	00037				:
													00	00	00046				:
													010E0046	00048	P.AAA:	.LONG	17694790		:
													00000000	0004C		.ADDRESS	P.AAB		:
62	20	6F	73	6C	61	20	74	68	67	69	6D	20	74	49	00050	P.AAD:	.ASCII	\It might also be explained by normal eve\	:
20	79	62	20	64	65	6E	69	61	6C	70	78	65	20	65	0005F				:
					65	76	65	20	6C	61	6D	72	6F	6E	0006E				:


```

79 6C 6C 61 69 63 65 70 73 65 20 2C 73 74 6E 00078
20 65 76 61 68 20 73 65 64 6F 6E 20 66 69 20 00087
                                00 74 73 75 6A 00096
                                00 00098
                                010E004A 0009C P.AAC:
                                00000000 000A0 .LONG 17694794
6F 62 65 72 20 72 6F 20 64 65 68 73 61 72 63 000A4 P.AAF:
6C 63 58 41 56 20 61 20 6E 69 20 64 65 74 6F 000B3 .ADDRESS P.AAD
                                6C 50 20 20 2E 72 65 74 73 75 000C2
73 69 68 74 20 67 6E 69 72 62 20 65 73 61 65 000CC .ASCII \crashed or rebooted in a VAXcluster. PL\
67 69 44 20 6F 74 20 65 67 61 73 73 65 6D 20 000DB
                                73 27 6C 61 74 69 000EA
                                010E004C 000F0 P.AAE:
                                00000000 000F4 .LONG 17694796
20 79 6C 6E 6F 20 6E 6F 69 74 6E 65 74 74 61 000F8 P.AAH:
69 76 61 68 20 65 72 61 20 75 6F 79 20 66 69 00107 .ADDRESS P.AAF
                                6D 65 6C 62 6F 72 70 20 67 6E 00116 .ASCII \attention only if you are having problem\
72 6F 74 61 72 65 70 6F 20 68 74 69 77 20 73 00120
73 6E 6F 69 74 61 63 69 6E 75 6D 6D 6F 63 20 0012F
                                00 2E 0013E
                                010E0047 00140 P.AAG:
                                00000000 00144 .LONG 17694791
50 4F 20 20 25 25 25 25 25 25 25 25 25 25 25 00148 P.AAJ:
25 25 25 25 25 20 20 44 25 21 20 20 4D 4F 43 00157 .ADDRESS P.AAH
                                55 44 20 20 25 25 25 25 25 25 00166 .ASCII \XXXXXXXXXX OPCOM !%D XXXXXXXXXXXX DU\
                                00 45 4C 49 46 5F 47 4F 4C 5F 50 4D 00170
                                010E0033 0017C P.AAI:
                                00000000 00180 .LONG 1769477T
20 4C 55 35 21 20 73 69 20 72 65 66 66 75 42 00184 P.AAL:
2D 20 73 65 74 79 62 20 29 57 58 21 58 25 28 00193 .ADDRESS P.AAJ
                                00 22 53 41 21 22 20 2D 001A2 .ASCII \Buffer is !5UL (%X!XW) bytes -- '!AS'\<0>
                                00 00 001AA
                                010E0025 001AC P.AAK:
                                00000000 001B0 .LONG 17694757
21 20 46 41 32 33 21 20 29 4C 58 39 28 38 21 001B4 P.AAN:
                                00 00 00 57 58 001C3 .ADDRESS P.AAL
                                010E0011 001C8 P.AAM:
                                00000000 001CC .LONG 17694737
                                .ADDRESS P.AAN

```

EXP1=
EXP2=
EXP3=
EXP4=
DSC=
DSC=
DSC=
DSC=

P.AAA
P.AAC
P.AAE
P.AAG
P.AAA
P.AAC
P.AAE
P.AAG

.EXTRN GLOBAL STATUS, LOGFILE_RAB
.EXTRN LOGFILE_FAB, SYSSPUT
.EXTRN SYSSFLUSH

.PSECT \$CODE\$,NOWRT,2

.ENTRY DUMP_LOG_FILE, Save R2,R3,R4,R5,R6,R7,R8,-
R9,RT0,RT1
MOVAB P.AAI, R11
MOVAB SYSSPUT, R10

OFFC 00000

5B 0000 0000 CF 9E 00002
5A 00000000G 00 9E 00007

0107

		59	0000G	CF	9E	0000E	MOVAB	LOGFILE_RAB, R9		
		5E		20	C2	00013	SUBL2	#32, SP		
03	0000G	CF		01	E0	00016	BBS	#1, GLOBAL_STATUS, 2\$	0161	
				00F9	31	0001C	BRW	6\$		
F7	0000G	CF		03	E0	0001F	BBS	#3, GLOBAL_STATUS, 1\$	0162	
				6E	D4	00025	CLRL	INTER	0168	
	22	A9		6E	B0	00027	MOVW	DSC, LOGFILE_RAB+34	0169	
	28	A9	04	AE	D0	0002B	MOVL	DSC+4, LOGFILE_RAB+40		
				59	DD	00030	PUSHL	R9		
		6A		01	FB	00032	CALLS	#1, SYSS\$PUT		
				7E	D4	00035	CLRL	-(SP)	0170	
				5B	DD	00037	PUSHL	R11		
	0000V	CF		02	FB	00039	CALLS	#2, LOGEVENT FAO BUFFER		
	22	A9		60	B0	0003E	MOVW	(R0), LOGFILE_RAB+34		
	28	A9	04	A0	D0	00042	MOVL	4(R0), LOGFILE_RAB+40		
				59	DD	00047	PUSHL	R9		
		6A		01	FB	00049	CALLS	#1, SYSS\$PUT		
	22	A9	FECC	CB	B0	0004C	MOVW	DSC, LOGFILE_RAB+34		
	28	A9	FED0	CB	D0	00052	MOVL	DSC+4, LOGFILE_RAB+40		
				59	DD	00058	PUSHL	R9		
		6A		01	FB	0005A	CALLS	#1, SYSS\$PUT		
	22	A9	FF20	CB	B0	0005D	MOVW	DSC, LOGFILE_RAB+34		
	28	A9	FF24	CB	D0	00063	MOVL	DSC+4, LOGFILE_RAB+40		
				59	DD	00069	PUSHL	R9		
		6A		01	FB	0006B	CALLS	#1, SYSS\$PUT		
	22	A9	FF74	CB	B0	0006E	MOVW	DSC, LOGFILE_RAB+34		
	28	A9	FF78	CB	D0	00074	MOVL	DSC+4, LOGFILE_RAB+40		
				59	DD	0007A	PUSHL	R9		
		6A		01	FB	0007C	CALLS	#1, SYSS\$PUT		
	22	A9	C4	AB	B0	0007F	MOVW	DSC, LOGFILE_RAB+34		
	28	A9	C8	AB	D0	00084	MOVL	DSC+4, LOGFILE_RAB+40		
				59	DD	00089	PUSHL	R9		
		6A		01	FB	0008B	CALLS	#1, SYSS\$PUT		
			08	AC	DD	0008E	PUSHL	ID_DESC	0173	
		52	04	AC	D0	00091	MOVL	BUFF_DESC, R2		
		7E		62	3C	00095	MOVZWL	(R2), -(SP)		
		7E		62	3C	00098	MOVZWL	(R2), -(SP)		
			30	AB	9F	0009B	PUSHAB	P.AAK		
	0000V	CF		04	FB	0009E	CALLS	#4, LOGEVENT FAO BUFFER		
	22	A9		60	B0	000A3	MOVW	(R0), LOGFILE_RAB+34		
	28	A9	04	A0	D0	000A7	MOVL	4(R0), LOGFILE_RAB+40		
				59	DD	000AC	PUSHL	R9		
		6A		01	FB	000AE	CALLS	#1, SYSS\$PUT		
		56		62	3C	000B1	MOVZWL	(R2), LEFT	0177	
		57	04	A2	D0	000B4	MOVL	4(R2), PTR	0178	
				58	D4	000B8	CLRL	BASE	0179	
				56	D5	000BA	TSTL	LEFT	0180	
				51	15	000BC	BLEQ	5\$		
		50		56	D0	000BE	MOVL	LEFT, R0	0186	
		20		50	D1	000C1	CMPL	R0, #32		
				03	1B	000C4	BLEQU	4\$		
		50		20	D0	000C6	MOVL	#32, R0		
20				50	2C	000C9	MOVC5	R0, (PTR), #0, #32, INTER		
	00	67		6E		000CE				
				58	DD	000CF	PUSHL	BASE	0190	
			04	AE	9F	000D1	PUSHAB	INTER		
				20	DD	000D4	PUSHL	#32		

		0C	AE	DD	000D6	PUSHL	INTER	:	
		14	AE	DD	000D9	PUSHL	INTER+4	:	
		1C	AE	DD	000DC	PUSHL	INTER+8	:	
		24	AE	DD	000DF	PUSHL	INTER+12	:	
		2C	AE	DD	000E2	PUSHL	INTER+16	:	
		34	AE	DD	000E5	PUSHL	INTER+20	:	
		3C	AE	DD	000E8	PUSHL	INTER+24	:	
		44	AE	DD	000EB	PUSHL	INTER+28	:	
		4C	AB	9F	000EE	PUSHAB	P.AAM	:	
0000V	CF		0C	FB	000F1	CALLS	#12, LOGEVENT_FAO_BUFFER	:	
22	A9		60	B0	000F6	MOVW	(R0), LOGFILE_RAB+34	:	
28	A9	04	A0	D0	000FA	MOVL	4(R0), LOGFILE_RAB+40	:	
			59	DD	000FF	PUSHL	R9	:	
	6A		01	FB	00101	CALLS	#1, SYSSPUT	:	
	58		20	C0	00104	ADDL2	#32, BASE	:	0194
	57		20	C0	00107	ADDL2	#32, PTR	:	0195
	56		20	C2	0010A	SUBL2	#32, LEFT	:	0196
			AB	11	0010D	BRB	3\$	:	0180
			59	DD	0010F	PUSHL	R9	:	0199
00000000G	00		01	FB	00111	CALLS	#1, SYSSFLUSH	:	
	50		01	D0	00118	MOVL	#1, R0	:	0201
			04	0011B	RET			:	0202

; Routine Size: 284 bytes, Routine Base: \$CODE\$ + 0000


```
205 0203 1 GLOBAL ROUTINE LOG_MESSAGE (RQCB) =
206 0204 1
207 0205 1 ++
208 0206 1 Functional description:
209 0207 1
210 0208 1 This routine will write a message described by an MCB
211 0209 1 to the operator log file.
212 0210 1 **
213 0211 1 ** This routine will be enhanced later to support mutliple log files.
214 0212 1 **
215 0213 1
216 0214 1 Input:
217 0215 1
218 0216 1 RQCB : Address of an RQCB data structure
219 0217 1
220 0218 1 Implicit Input:
221 0219 1
222 0220 1 RQCB [RQCB_L_MCB] points to a valid MCB.
223 0221 1
224 0222 1 Output:
225 0223 1
226 0224 1 None.
227 0225 1
228 0226 1 Implicit output:
229 0227 1
230 0228 1 None.
231 0229 1
232 0230 1 Side effects:
233 0231 1
234 0232 1 None.
235 0233 1
236 0234 1 Routine value:
237 0235 1
238 0236 1 TRUE : If success
239 0237 1 <anything else> : If the log attempt failed
240 0238 1 --
241 0239 1
242 0240 2 BEGIN ! Start of LOG_MESSAGE
243 0241 2
244 0242 2 MAP
245 0243 2 RQCB : $ref_bblock;
246 0244 2
247 0245 2 EXTERNAL LITERAL
248 0246 2 MCB_K_TYPE; ! MCB structure type
249 0247 2
250 0248 2 EXTERNAL
251 0249 2 GLOBAL_STATUS : BITVECTOR; ! OPCOM global status flags
252 0250 2
253 0251 2 LOCAL
254 0252 2 ADR : REF VECTOR [, BYTE], ! Adjusted address of string
255 0253 2 LEN : LONG, ! Adjusted length of string
256 0254 2 RECLEN : LONG, ! Adjusted length of single record
257 0255 2 CHAR : BYTE,
258 0256 2 BUF : VECTOR [OPC$K_MAXMESSAGE, BYTE],
259 0257 2 BUFP : REF VECTOR [, BYTE],
260 0258 2 MCB : $ref_bblock, ! MCB data structure
261 0259 2 STATUS : LONG;
```



```
262 0260 2 |
263 0261 2 |
264 0262 2 | Check for a valid MCB.
265 0263 2 |
266 0264 2 MCB = .RQCB [RQCB_L_MCB];
267 0265 2 IF (.MCB EQL 0) OR (.MCB [MCB_B_TYPE] NEQ MCB_K_TYPE)
268 0266 2 THEN
269 0267 2 RETURN (FALSE);
270 0268 2 |
271 0269 2 | See if logging is enabled.
272 0270 2 |
273 0271 2 IF (NOT .GLOBAL_STATUS [GBLSTS_K_LOGGING_ENABLED])
274 0272 2 OR (.GLOBAL_STATUS [GBLSTS_K_LOGFILE_CLOSED])
275 0273 2 OR ((.Sbblock [RQCB [RQCB_L_OPTIONS], OPC$V_NOLOG]) AND
276 0274 2 (.Sbblock [RQCB [RQCB_L_PRIVMASK1], PRV$V_OPER]))
277 0275 2 THEN
278 0276 2 RETURN (TRUE);
279 0277 2 |
280 0278 2 | Adjust the string. Remove any trailing control and space characters
281 0279 2 |
282 0280 2 LEN = .MCB [MCB_L_TEXTLEN];
283 0281 2 ADR = .MCB [MCB_L_TEXTPTR];
284 0282 2 IF .LEN LEQ 0
285 0283 2 THEN
286 0284 2 RETURN (TRUE);
287 0285 2 DECR I FROM .LEN-1 TO 0 | Remove all trailing control and space characters
288 0286 2 DO
289 0287 2 SELECTONE .ADR [I] OF
290 0288 2 SET
291 0289 2 [0 TO 32] : LEN = .LEN - 1; | Bad one, remove it
292 0290 2 [OTHERWISE] : EXITLOOP; | Good one, we are done looking
293 0291 2 TES;
294 0292 2 IF .LEN LEQ 0 | If we got rid of the whole string, we are done
295 0293 2 THEN
296 0294 2 RETURN (TRUE);
297 0295 2 |
298 0296 2 | Now get ready to filter the string. We will copy it to a local buffer, making some changes.
299 0297 2 | Ignore all control characters except tab and line-feed. Pass tab through, if see a line-feed
300 0298 2 | then write the record found up to the line-feed. Do not print zero-length lines.
301 0299 2 |
302 0300 2 STATUS = LOG_MESSAGE_PUT (0, BUF); | Put a single blank line before the message
303 0301 2 RECLN = 0;
304 0302 2 BUFP = BUF;
305 0303 2 WHILE .LEN GTR 0
306 0304 2 DO
307 0305 2 BEGIN
308 0306 2 CHAR = CH$RCHAR_A (ADR);
309 0307 2 SELECTONE .CHAR OF
310 0308 2 SET
311 0309 2 |
312 0310 2 | Line feed, print the old record and start a new one
313 0311 2 |
314 0312 2 [10] : BEGIN
315 0313 2 IF .RECLN GTR 0
316 0314 2 THEN
317 0315 2 STATUS = LOG_MESSAGE_PUT (.RECLN, BUF);
318 0316 2 RECLN = 0;
```

```

319      0317      4      BUFP = BUF;
320      0318      END;
321      0319      :
322      0320      : Misc control char, ignore
323      0321      :
324      0322      : [0 TO 8, 11 TO 31] : ;
325      0323      :
326      0324      : Valid char, pass it to the new buffer
327      0325      :
328      0326      : [OTHERWISE] :
329      0327      : BEGIN
330      0328      : RECLN = .RECLN + 1;
331      0329      : CH$WCHAR_A (.CHAR, BUFP);
332      0330      : END;
333      0331      :
334      0332      : TES;
335      0333      : LEN = .LEN - 1;
336      0334      : END;
337      0335      : If we have some data in the new buffer, print it
338      0336      : IF .RECLN GTR 0
339      0337      : THEN
340      0338      : STATUS = LOG_MESSAGE_PUT (.RECLN, BUF);
341      0339      :
342      0340      : RETURN (.STATUS);
343      0341      : END;

```

! End of LOG_MESSAGE

				.EXTRN MCB_K_TYPE				
				.ENTRY LOG_MESSAGE, Save R2,R3,R4,R5,R6,R7				0203
				MOVAB LOG_MESSAGE_PUT, R7				
				MOVAB -2048(SP), SP				
				MOVL RQCB, R1				0264
				MOVL 108(R1), MCB				
				BNEQ 2\$				0265
				BRW 15\$				
				CMPZV #0, #8, 10(MCB), #MCB_K_TYPE				
				BNEQ 1\$				
				BBC #1, GLOBAL_STATUS, 7\$				0271
				BBS #3, GLOBAL_STATUS, 7\$				0272
				BLBC 84(R1), 3\$				0273
				BBS #2, 50(R1), 7\$				0274
				MOVL 48(MCB), LEN				0280
				MOVL 52(MCB), ADR				0281
				TSTL LEN				0282
				BLEQ 7\$				
				MOVL LEN, I				0287
				BRB 5\$				
				MOVZBL (I)[ADR], R1				
				CMPB R1, #32				0289
				BGTRU 6\$				
				DECL LEN				
				SOBGEQ I, 4\$				0287
				TSTL LEN				0292
				BGTR 8\$				
				MOVL #1, R0				0294

				00FC 00000				
57	0000V	CF	9E	00002				
5E	F800	CE	9E	00007				
51	04	AC	D0	0000C				
50	6C	A1	D0	00010				
		03	12	00014				
		0096	31	00016	1\$:			
		00	ED	00019	2\$:			
		F1	12	00023				
		01	E1	00025				
		03	E0	0002B				
		54	A1	E9	00031			
		02	E0	00035				
		52	A0	D0	0003A	3\$:		
		56	34	A0	D0	0003E		
			52	D5	00042			
			17	15	00044			
			52	D0	00046			
			0B	11	00049			
			6046	9A	0004B	4\$:		
			51	91	0004F			
			05	1A	00052			
			52	D7	00054			
			50	F4	00056	5\$:		
			52	D5	00059	6\$:		
			04	14	0005B			
			50	01	D0	0005D	7\$:	

		04	00060	RET		
	5E	DD	00061	8\$: PUSH	SP	0300
	7E	D4	00063	CLRL	-(SP)	
67	02	FB	00065	CALLS	#2, LOG_MESSAGE_PUT	
	53	D4	00068	CLRL	RECLN	0301
55	6E	9E	0006A	MOVAB	BUF, BUFP	0302
	52	D5	0006D	9\$: TSTL	LEN	0303
	32	15	0006F	BLEQ	14\$	
54	86	90	00071	MOVB	(ADR)+, CHAR	0306
0A	54	91	00074	CMPB	CHAR, #10	0312
	12	12	00077	BNEQ	11\$	
	53	D5	00079	TSTL	RECLN	0313
	07	15	0007B	BLEQ	10\$	
	8F	BB	0007D	PUSHR	#*M<R3,SP>	0315
67	02	FB	00081	CALLS	#2, LOG_MESSAGE_PUT	
	53	D4	00084	10\$: CLRL	RECLN	0316
55	6E	9E	00086	MOVAB	BUF, BUFP	0317
	14	11	00089	BRB	13\$	0307
08	54	91	0008B	11\$: CMPB	CHAR, #8	0322
	0F	1B	0008E	BLEQU	13\$	
0B	54	91	00090	CMPB	CHAR, #11	
	05	1F	00093	BLSSU	12\$	
1F	54	91	00095	CMPB	CHAR, #31	
	05	1B	00098	BLEQU	13\$	
	53	D6	0009A	12\$: INCL	RECLN	0327
85	54	90	0009C	MOVB	CHAR, (BUFP)+	0328
	52	D7	0009F	13\$: DECL	LEN	0331
	CA	11	000A1	BRB	9\$	0303
	53	D5	000A3	14\$: TSTL	RECLN	0336
	0A	15	000A5	BLEQ	16\$	
	8F	BB	000A7	PUSHR	#*M<R3,SP>	0338
67	02	FB	000AB	CALLS	#2, LOG_MESSAGE_PUT	
		04	000AE	RET		0340
	50	D4	000AF	15\$: CLRL	R0	0341
		04	000B1	16\$: RET		

; Routine Size: 178 bytes, Routine Base: \$CODE\$ + 011C

```
345 0342 1 GLOBAL ROUTINE LOG_MESSAGE_PUT (LEN, ADR) =
346 0343 1
347 0344 1 **
348 0345 1 Functional description:
349 0346 1
350 0347 1 Place the record in the log file.
351 0348 1 **
352 0349 1 ** This routine will be enhanced later to support multiple log files.
353 0350 1 **
354 0351 1
355 0352 1 Input:
356 0353 1
357 0354 1 LEN - Length of record
358 0355 1 ADR - Address of record
359 0356 1
360 0357 1 Implicit Input:
361 0358 1
362 0359 1 None.
363 0360 1
364 0361 1 Output:
365 0362 1
366 0363 1 None.
367 0364 1
368 0365 1 Implicit output:
369 0366 1
370 0367 1 None.
371 0368 1
372 0369 1 Side effects:
373 0370 1
374 0371 1 None.
375 0372 1
376 0373 1 Routine value:
377 0374 1
378 0375 1 TRUE : If success
379 0376 1 <anything else> : If the log attempt failed
380 0377 1 --
381 0378 1
382 0379 2 BEGIN ! Start of LOG_MESSAGE_PUT
383 0380 2
384 0381 2 EXTERNAL
385 0382 2 GLOBAL STATUS : BITVECTOR, ! OPCOM global status flags
386 0383 2 LOGFILE_RAB : $bblock, ! RMS control structure
387 0384 2 LOGFILE_FAB : $bblock; ! RMS control structure
388 0385 2
389 0386 2 LOCAL
390 0387 2 MESSAGE : LONG, ! Error message code
391 0388 2 STATUS : LONG;
392 0389 2
393 0390 2 Write the message to the logfile.
394 0391 2
395 0392 2 LOGFILE_RAB [RAB$W_RSZ] = .LEN;
396 0393 2 LOGFILE_RAB [RAB$L_RBF] = .ADR;
397 0394 2 GLOBAL STATUS [GBLSTS_K_FLUSH_PENDING] = TRUE;
398 0395 2 IF NOT (STATUS = $PUT (RAB = LOGFILE_RAB))
399 0396 2 THEN
400 0397 3 BEGIN
401 0398 3 !
```



```

: 402      0399 3      ! The log attempt failed.  Complain if appropriate.
: 403      0400 3      !
: 404      0401 3      IF NOT .GLOBAL_STATUS [GBLSTS_K_LAST_LOG_FAILED]
: 405      0402 3      THEN
: 406      0403 4      BEGIN
: 407      0404 4      !
: 408      0405 4      ! Complain to the appropriate operators.
: 409      0406 4      !
: 410      0407 4      MESSAGE = OPC$ LOGFAIL;
: 411      0408 4      ! *** the remainder will be supplied later ***
: 412      0409 3      END;
: 413      0410 3      GLOBAL_STATUS [GBLSTS_K_LAST_LOG_FAILED] = TRUE;
: 414      0411 2      END;
: 415      0412 2
: 416      0413 2 RETURN (.STATUS);
: 417      0414 2
: 418      0415 1 END;
! End of LOG_MESSAGE_PUT

```

			0004 00000	.ENTRY LOG MESSAGE PUT, Save R2	: 0342
	52	0000G CF	9E 00002	MOVAB GLOBAL_STATUS, R2	: 0392
	0000G CF	04 AC	B0 00007	MOVW LEN, LOGFILE_RAB+34	: 0393
	0000G CF	08 AC	D0 0000D	MOVL ADR, LOGFILE_RAB+40	: 0394
	62	80 8F	88 00013	BISB2 #128, GLOBAL_STATUS	: 0395
		0000G CF	9F 00017	PUSHAB LOGFILE_RAB	
	00000000G 00	01 FB	0001B	CALLS #1, SYS\$PUT	
	0E	50 E8	00022	BLBS STATUS, 2\$	
07	62	02 E0	00025	BBS #2, GLOBAL_STATUS, 1\$	: 0401
	51 00058034	8F D0	00029	MOVL #360500, MESSAGE	: 0407
	62	04 88	00030 1\$:	BISB2 #4, GLOBAL_STATUS	: 0410
			04 00033 2\$:	RET	: 0415

; Routine Size: 52 bytes, Routine Base: \$CODE\$ + 01CE

```

: 420 0416 1 ROUTINE LOGEVENT_FAO_BUFFER (CTRSTR : REF VECTOR[2], ARGS : VECTOR [4]) = %SBTTL 'LOGEVENT_FAO_BUFFER'
: 421 0417 2 BEGIN
: 422 0418 2 ++
: 423 0419 2
: 424 0420 2 FUNCTIONAL DESCRIPTION:
: 425 0421 2
: 426 0422 2 This routine passes an ascii string through the FAO system service with any number of specified para
: 427 0423 2
: 428 0424 2 INPUTS:
: 429 0425 2
: 430 0426 2 ctrstr Address of FAO control string descriptor
: 431 0427 2 args Any number of additional arguments
: 432 0428 2
: 433 0429 2 IMPLICIT INPUTS:
: 434 0430 2
: 435 0431 2 none
: 436 0432 2
: 437 0433 2 OUTPUTS:
: 438 0434 2
: 439 0435 2 none
: 440 0436 2
: 441 0437 2 IMPLICIT OUTPUTS:
: 442 0438 2
: 443 0439 2 none
: 444 0440 2
: 445 0441 2 ROUTINE VALUE:
: 446 0442 2
: 447 0443 2 Address of formatted descriptor
: 448 0444 2
: 449 0445 2 SIDE EFFECTS:
: 450 0446 2
: 451 0447 2 none
: 452 0448 2 --
: 453 0449 2
: 454 0450 2 OWN
: 455 0451 2 desc : VECTOR [2, LONG],
: 456 0452 2 faobuf : VECTOR [512, BYTE]
: 457 0453 2 ;
: 458 0454 2
: 459 0455 2 desc [0] = 512; ! Set up result descriptor
: 460 0456 2 desc [1] = faobuf;
: 461 0457 2
: 462 0458 2 $faol (ctrstr=.ctrstr, outlen=desc, outbuf=desc, prmlst=args);
: 463 0459 2
: 464 0460 2 RETURN desc;
: 465 0461 1 END;

```

```

.PSECT $OWNS$,NOEXE,2
00000 DESC: .BLKB 8
00008 FAOBUF: .BLKB 512
.EXTRN SYSS$FAOL
.PSECT $CODE$,NOWRT,2

```


		0004 00000	LOGEVENT_FAO_BUFFER:		
	52	0000'	CF 9E 00002	WORD Save R2	: 0416
	62	0200	8F 3C 00007	MOVAB DESC, R2	: 0455
04	A2	08	A2 9E 0000C	MOVZWL #512, DESC	: 0456
		08	AC 9F 00011	MOVAB FAOBUF, DESC+4	: 0458
			52 DD 00014	PUSHAB ARGS	: 0460
			52 DD 00016	PUSHL R2	: 0461
		04	AC DD 00018	PUSHL R2	
00000000G	00		04 FB 0001B	PUSHL CTRSTR	
	50		62 9E 00022	CALLS #4, SYSSFAOL	
			04 00025	MOVAB DESC, R0	
				RET	

; Routine Size: 38 bytes, Routine Base: \$CODE\$ + 0202

```

467 0462 1 GLOBAL ROUTINE WRITE_LOG_FILE (DESC : $ref_bblock) =
468 0463 1
469 0464 1 ++
470 0465 1 Functional description:
471 0466 1
472 0467 1 This routine will write a message described by simple string desc
473 0468 1 to the operator log file.
474 0469 1 **
475 0470 1 ** This routine will be enhanced later to support mutiple log files.
476 0471 1 **
477 0472 1
478 0473 1 Input:
479 0474 1
480 0475 1 DESC : Address of a string desc
481 0476 1
482 0477 1 Implicit Input:
483 0478 1
484 0479 1 RQCB [RQCB_L_MCB] points to a valid MCB.
485 0480 1
486 0481 1 Output:
487 0482 1
488 0483 1 None.
489 0484 1
490 0485 1 Implicit output:
491 0486 1
492 0487 1 None.
493 0488 1
494 0489 1 Side effects:
495 0490 1
496 0491 1 None.
497 0492 1
498 0493 1 Routine value:
499 0494 1
500 0495 1 TRUE : If success
501 0496 1 <anything else> : If the log attempt failed
502 0497 1 --
503 0498 1
504 0499 2 BEGIN ! Start of WRITE_LOG_FILE
505 0500 2
506 0501 2 EXTERNAL
507 0502 2 GLOBAL STATUS : BITVECTOR, ! OPCOM global status flags
508 0503 2 LOGFILE_RAB : $bblock, ! RMS control structure
509 0504 2 LOGFILE_FAB : $bblock; ! RMS control structure
510 0505 2
511 0506 2 LOCAL
512 0507 2 NULLDESC : LONG; ! Only need length word
513 0508 2
514 0509 2
515 0510 2 See if logging is enabled.
516 0511 2
517 0512 3 IF (NOT .GLOBAL STATUS [GBLSTS_K_LOGGING_ENABLED])
518 0513 3 OR (.GLOBAL STATUS [GBLSTS_K_LOGFILE_CLOSED])
519 0514 2 THEN
520 0515 2 RETURN (TRUE);
521 0516 2
522 0517 2 Format and print the message header
523 0518 2

```



```
: 524      0519 2 NULLDESC = 0;
: 525      0520 2 PUT (NULLDESC);
: 526      0521 2 PUT (LOGEVENT_FA0_BUFFER (%ASCID 'XXXXXXXXXX OPCOM !XD XXXXXXXXXXXX WRITE_LOG_FILE', 0));
: 527      0522 2 PUT_EXPLANATION;
: 528      0523 2 PUT (LOGEVENT_FA0_BUFFER (%ASCID '!'AS'', .DESC));
: 529      0524 2 $FLUSH (RAB = LOGFILE_RAB);
: 530      0525 2
: 531      0526 2 RETURN (TRUE);
: 532      0527 1 END;
```

! A blank line

! End of WRITE_LOG_FILE

```
50 4F 20 20 25 25 25 25 25 25 25 25 25 25 001D0 P.AAP: .PSECT $PLITS,NOWRT,NOEXE,2
25 25 25 25 25 20 20 44 25 21 20 20 4D 4F 43 001DF .ASCII \XXXXXXXXXX OPCOM !XD XXXXXXXXXXXX WR\
45 4C 49 46 5F 47 4F 4C 5F 45 54 49 001EE
010E0034 00204 P.AAO: .ASCII \ITE_LOG_FILE\
00000000' 00208 .LONG 17694772
00 00 00 22 53 41 21 22 0020C P.AAR: .ADDRESS P.AAP
010E0005 00214 P.AAQ: .ASCII \!'AS'\<0><0><0>
00000000' 00218 .LONG 17694725
                                .ADDRESS P.AAR
                                DSC= P.AAA
                                DSC= P.AAC
                                DSC= P.AAE
                                DSC= P.AAG
```

```
03 0000G CF 0094 31 00017 1$:
F7 0000G CF 03 E0 0001A 2$:
22 A2 6E D4 00020
28 A2 6E B0 00022
63 52 DD 0002A
0000' 01 FB 0002C
7E D4 0002F
A1 AF 0000' CF 9F 00031
22 A2 02 FB 00035
28 A2 04 B0 00039
63 52 DD 00042
22 A2 0000' 01 FB 00044
28 A2 0000' CF B0 00047
63 52 DD 00053
22 A2 0000' 01 FB 00055
28 A2 0000' CF B0 00058
0000' CF D0 0005E
```

```
.ENTRY WRITE_LOG_FILE, Save R2,R3
MOVAB SYSSPUT, R3
MOVAB LOGFILE_RAB, R2
SUBL2 #4, SP
BBS #1, GLOBAL_STATUS, 2$
BRW 3$
BBS #3, GLOBAL_STATUS, 1$
CLRL NULLDESC
MOVW DSC, LOGFILE_RAB+34
MOVL DSC+4, LOGFILE_RAB+40
PUSHL R2
CALLS #1, SYSSPUT
CLRL -(SP)
PUSHAB P.AAO
CALLS #2, LOGEVENT_FA0_BUFFER
MOVW (R0), LOGFILE_RAB+34
MOVL 4(R0), LOGFILE_RAB+40
PUSHL R2
CALLS #1, SYSSPUT
MOVW DSC, LOGFILE_RAB+34
MOVL DSC+4, LOGFILE_RAB+40
PUSHL R2
CALLS #1, SYSSPUT
MOVW DSC, LOGFILE_RAB+34
MOVL DSC+4, LOGFILE_RAB+40
```

0462
0512
0513
0519
0520
0521

```

      22 63      0000' 52 DD 00064      PUSHL R2
      28 A2      0000' 01 FB 00066      CALLS #1, SYSS$PUT
      28 A2      0000' CF B0 00069      MOVW DSC, LOGFILE_RAB+34
      28 A2      0000' CF D0 0006F      MOVL DSC+4, LOGFILE_RAB+40
      22 63      0000' 52 DD 00075      PUSHL R2
      28 A2      0000' 01 FB 00077      CALLS #1, SYSS$PUT
      28 A2      0090' CF B0 0007A      MOVW DSC, LOGFILE_RAB+34
      28 A2      0090' CF D0 00080      MOVL DSC+4, LOGFILE_RAB+40
      28 A2      0090' 52 DD 00086      PUSHL R2
      28 A2      0090' 01 FB 00088      CALLS #1, SYSS$PUT
      28 A2      0090' AC DD 0008B      PUSHL DESC
      28 A2      0090' CF 9F 0008E      PUSHAB P.AAQ
      FF43 CF      0000' 02 FB 00092      CALLS #2, LOGEVENT_FAO_BUFFER
      22 A2      0000' 60 B0 00097      MOVW (R0), LOGFILE_RAB+34
      28 A2      0000' A0 D0 0009B      MOVL 4(R0), LOGFILE_RAB+40
      28 A2      0000' 52 DD 000A0      PUSHL R2
      28 A2      0000' 01 FB 000A2      CALLS #1, SYSS$PUT
      28 A2      0000' 52 DD 000A5      PUSHL R2
      00000000G 00      01 FB 000A7      CALLS #1, SYSS$FLUSH
      28 A2      0000' 01 D0 000AE 3$: MOVL #1, R0
      28 A2      0000' 04 000B1      RET
```

0523

0524

0526

0527

; Routine Size: 178 bytes, Routine Base: \$CODE\$ + 0228

```

: 533      0528 1
: 534      0529 1 END
: 535      0530 0 ELUDOM
```

! End of LOGEVENT

PSECT SUMMARY

Name	Bytes	Attributes
\$SPLITS	540	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	730	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$OWNS	520	NOVEC, WRT, RD, NOEXE, 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	13	0	1000	00:01.8
\$255\$DUA28:[OPCOM.OBJ]OPCOMLIB.L32;1	633	15	2	43	00:00.8

OPC\$LOGEVENT
V04-000

LOGEVENT_FAO_BUFFER

N 1
16-Sep-1984 01:29:21
14-Sep-1984 12:50:43

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

Page 19
(6)

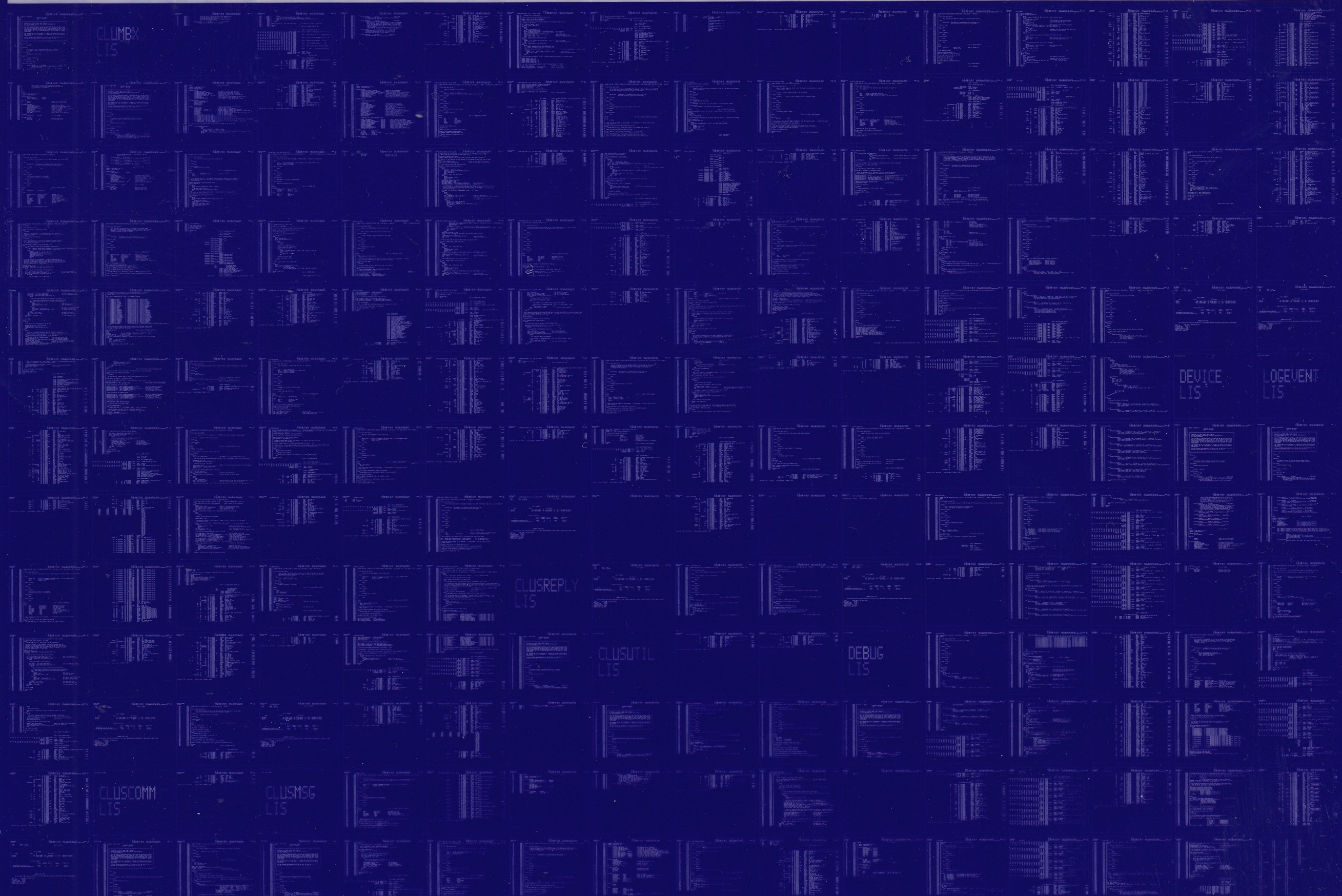
COMMAND QUALIFIERS

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

; Size: 730 code + 1060 data bytes
; Run Time: 00:18.2
; Elapsed Time: 01:01.1
; Lines/CPU Min: 1751
; Lexemes/CPU-Min: 23511
; Memory Used: 135 pages
; Compilation Complete

0289 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



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

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

