



```

SSSSSSSS  NN      NN  DDDDDDD  EEEEEEEEE  RRRRRRR  LL
SSSSSSSS  NN      NN  DDDDDDD  EEEEEEEEE  RRRRRRR  LL
SS         NN      NN  DD        DD  EE         RR      RR  LL
SS         NN      NN  DD        DD  EE         RR      RR  LL
SS         NNNN    NN  DD        DD  EE         RR      RR  LL
SS         NNNN    NN  DD        DD  EE         RR      RR  LL
          SSSSSS  NN  NN  NN  DD  DD  EEEEEEEEE  RRRRRRR  LL
          SSSSSS  NN  NN  NN  DD  DD  EEEEEEEEE  RRRRRRR  LL
                   SS  NN      NNNN  DD  DD  EE         RR  RR  LL
                   SS  NN      NNNN  DD  DD  EE         RR  RR  LL
                   SS  NN      NN     DD  DD  EE         RR      RR  LL
                   SS  NN      NN     DD  DD  EE         RR      RR  LL
SSSSSSSS  NN      NN  DDDDDDD  EEEEEEEEE  RR      RR  LLLLLLLLLL
SSSSSSSS  NN      NN  DDDDDDD  EEEEEEEEE  RR      RR  LLLLLLLLLL

```

```

LL         IIIIII  SSSSSSS
LL         IIIIII  SSSSSSS
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
LLLLLLLLLL IIIIII  SSSSSSS
LLLLLLLLLL IIIIII  SSSSSSS

```

1  
2  
:001 :EAD0210  
4-1  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
:001 :EAD0210  
:002 :EAD0210  
:003 :EAD0210  
52  
53  
54

0001 0  
0002 0  
0003 0  
0004 0  
0005 1  
0006 1  
0007 1  
0008 1  
0009 1  
0010 1  
0011 1  
0012 1  
0013 1  
0014 1  
0015 1  
0016 1  
0017 1  
0018 1  
0019 1  
0020 1  
0021 1  
0022 1  
0023 1  
0024 1  
0025 1  
0026 1  
0027 1  
0028 1  
0029 1  
0030 1  
0031 1  
0032 1  
0033 1  
0034 1  
0035 1  
0036 1  
0037 1  
0038 1  
0039 1  
0040 1  
0041 1  
0042 1  
0043 1  
0044 1  
0045 1  
0046 1  
0047 1  
0048 1  
0049 1  
0050 1  
0051 1  
0052 1  
0053 1  
0054 1  
0055 1  
0056 1  
0057 1

```
MODULE SNDRERL (  
    LANGUAGE (BLISS32),  
    IDENT = 'V04-001'  
    ) =  
  
BEGIN  
  
*****  
*  
* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY  
* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.  
* ALL RIGHTS RESERVED.  
*  
* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  
* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE  
* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER  
* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  
* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY  
* TRANSFERRED.  
*  
* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE  
* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT  
* CORPORATION.  
*  
* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS  
* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.  
*  
*****  
  
**  
  
FACILITY: F11ACP Structure Level 1  
  
ABSTRACT:  
  
    This routine sends a message to the error logger to inform it of a  
    volume mount or dismount.  
  
ENVIRONMENT:  
  
    STANLET operating system, including privileged system services  
    and internal exec routines.  
  
--  
  
AUTHOR: Andrew C. Goldstein, CREATION DATE: 23-Jun-1978 18:47  
  
MODIFIED BY:  
  
    V04-001 EAD0210 Elliott A. Drayton 16-Nov-1984  
    Add code to log sysnode name.  
  
    V03-001 LMP0221 L. Mark Pilant, 27-Mar-1984 14:46  
    Change UCBSL_OWNUIC to ORBSL_OWNER and UCBSW_VPROT to  
    ORBSW_PROT.
```

```
.. 55      0058 1  
.. 56      0059 1 : A0101 ACG0113 Andrew C. Goldstein, 15-Jan-1980 22:58  
.. 57      0060 1 : Fill in volume set data in error log message  
.. 58      0061 1 :  
.. 59      0062 1 : A0100 ACG00001 Andrew C. Goldstein, 10-Oct-1978 20:03  
.. 60      0063 1 : Previous revision history moved to F11A.REV  
.. 61      0064 1 : ..  
.. 62      0065 1 :  
.. 63      0066 1 :  
.. 64      0067 1 LIBRARY 'SYSS$LIBRARY:LIB.L32';  
.. 65      0068 1 REQUIRE 'SRC$:FCPDEF.B32';  
.. 66      1059 1 :  
.. 67      1060 1 :  
.. 68      1061 1 : This routine is called at raised IPL and must be locked into the working set.  
.. 69      1062 1 :  
.. 70      1063 1 :  
.. 71      1064 1 LOCK_CODE;
```

```

: 73      1065 1 GLOBAL ROUTINE SEND_ERRLOG (MODE, UCB) =
: 74      1066 1
: 75      1067 1  +-
: 76      1068 1
: 77      1069 1  FUNCTIONAL DESCRIPTION:
: 78      1070 1
: 79      1071 1      This routine sends a message to the error logger to inform it of a
: 80      1072 1      volume mount or dismount.
: 81      1073 1
: 82      1074 1
: 83      1075 1  CALLING SEQUENCE:
: 84      1076 1      SEND_ERRLOG (ARG1, ARG2)
: 85      1077 1
: 86      1078 1  INPUT PARAMETERS:
: 87      1079 1      ARG1: 1 to signal mount
: 88      1080 1      0 to signal dismount
: 89      1081 1      ARG3 address of UCB
: 90      1082 1
: 91      1083 1  IMPLICIT INPUTS:
: 92      1084 1      NONE
: 93      1085 1
: 94      1086 1  OUTPUT PARAMETERS:
: 95      1087 1      NONE
: 96      1088 1
: 97      1089 1  IMPLICIT OUTPUTS:
: 98      1090 1      NONE
: 99      1091 1
: 100     1092 1  ROUTINE VALUE:
: 101     1093 1      1
: 102     1094 1
: 103     1095 1  SIDE EFFECTS:
: 104     1096 1      Message sent to error logger
: 105     1097 1
: 106     1098 1  --
: 107     1099 1
: 108     1100 2 BEGIN
: 109     1101 2
: 110     1102 2 MAP
: 111     1103 2      UCB          : REF BBLOCK;      ! UCB argument
: 112     1104 2
: 113     1105 2 LINKAGE
: 114     1106 2      L_ERL_ALLOC = JSB (REGISTER = 1) :
: 115     1107 2      GLOBAL (ADDRESS = 2)
: 116     1108 2      NOTUSED (3, 4, 5, 6, 7, 8, 9, 10, 11);
: 117     1109 2
: 118     1110 2      L_ERL_RELEASE = JSB (REGISTER = 2) :
: 119     1111 2      NOTUSED (3, 4, 5, 6, 7, 8, 9, 10, 11);
: 120     1112 2
: 121     1113 2 LOCAL
: 122     1114 2      ORB          : REF BBLOCK,      ! local address of ORB
: 001 :EAD0210 1115 2      MSG_BUFFER : REF BBLOCK,      ! other buffer pointer to dodge MOVC
: 002 :EAD0210 1116 2      STRING       : REF VECTOR [,BYTE], ! Used for byte access to a string
: 003 :EAD0210 1117 2      STRING_START : REF VECTOR [,BYTE], ! Address pointer into msg_buffer
: 004 :EAD0210 1118 2      LENGTH_OFFSET : BYTE,          ! Used to add offset to string length
: 005 :EAD0210 1119 2      NAME_OFFSET  : LONG,          ! Used to add offset to device name addr
: 006 :EAD0210 1120 2      DDB          : REF BBLOCK;      ! Used to access DDB
: 124-1  1121 2

```

```
125 1122 2 EXTERNAL ROUTINE
126 1123 2 ERL$ALLOCEMB : L_ERL_ALLOC ADDRESSING MODE (GENERAL),
127 1124 2 : allocate error log buffer
128 1125 2 ERL$RELEASEMB : L_ERL_RELEASE ADDRESSING MODE (GENERAL);
129 1126 2 : release error log buffer
130 1127 2
131 1128 2
132 1129 2 ! Allocate an error log buffer. If this fails, forget it.
133 1130 2
134 1131 2
135 1132 2 BEGIN
136 1133 2 GLOBAL REGISTER
137 1134 2 ADDRESS = 2 : REF BBLOCK; ! pointer to error log buffer
138 1135 2
139 1136 2 IF NOT ERL$ALLOCEMB (EMBSK_VM_LENGTH)
140 1137 2 THEN RETURN 1;
141 1138 2 MSG_BUFFER = .ADDRESS;
142 1139 2 END;
143 1140 2
144 1141 2 ! Now fill in the message buffer.
145 1142 2
146 1143 2
147 1144 2 IF .MODE
148 1145 2 THEN MSG_BUFFER[EMBSW_VM_ENTRY] = EMBSK_VM
149 1146 2 ELSE MSG_BUFFER[EMBSW_VM_ENTRY] = EMBSK_VD; ! log entry type
150 1147 2
151 1148 2 ORB = .UCB[UCBSL_ORB];
152 1149 2 MSG_BUFFER[EMBSL_VM_OWNUIC] = .ORB[ORB$OWNER];
153 1150 2 MSG_BUFFER[EMBSL_VM_ERRCNT] = .UCB[UCBSW_ERRCNT];
154 1151 2 MSG_BUFFER[EMBSL_VM_OPRCNT] = .UCB[UCBSL_OPCNT];
155 1152 2 MSG_BUFFER[EMBSW_VM_UNIT] = .UCB[UCBSW_UNIT];
156 1153 2
157 1154 2 MSG_BUFFER[EMBSW_VM_VOLNUM] = 0;
158 1155 2 MSG_BUFFER[EMBSW_VM_NUMSET] = 0;
159 1156 2
:001 EAD0210 1157 2 STRING = MSG_BUFFER[EMBSB_VM_NAMLNG];
:002 EAD0210 1158 2 STRING [0] = 0;
:003 EAD0210 1159 2 DDB = .UCB[UCBSL_DDB];
:004 EAD0210 1160 2
:005 EAD0210 1161 2 IF .DDB[DDB$SB] NEQ 0 ! If there is a system block
:006 EAD0210 1162 2 THEN ! then if the node name string is NEQ 0
:007 EAD0210 1163 2 IF .(BBLOCK [.DDB[DDB$SB], SB$T_NODENAME])<0,8> NEQ 0
:008 EAD0210 1164 2 THEN
:009 EAD0210 1165 2 BEGIN ! Copy the string
:010 EAD0210 1166 2 CH$MOVE (.(BBLOCK [.DDB[DDB$SB], SB$T_NODENAME])<0,8> + 1,
:011 EAD0210 1167 2 BBLOCK [.DDB[DDB$SB], SB$T_NODENAME],
:012 EAD0210 1168 2 STRING [0]);
:013 EAD0210 1169 2 STRING [0] = .STRING [0] + 1; ! Bump the string count by 1
:014 EAD0210 1170 2 STRING [ .STRING [0] ] = %C'$'; ! Append a '$'
:015 EAD0210 1171 2 END;
:016 EAD0210 1172 2
:017 EAD0210 1173 2 IF .STRING[0] GTR 0
:018 EAD0210 1174 2 THEN
:019 EAD0210 1175 2 BEGIN
:020 EAD0210 1176 2 STRING_START = STRING[ .STRING[0] + 1 ];
:021 EAD0210 1177 2 LENGTH_OFFSET = 0;
:022 EAD0210 1178 2 NAME_OFFSET = 1;
```

```

:023 :EAD0210 1179 3 END
:024 :EAD0210 1180 2 ELSE
:025 :EAD0210 1181 3 BEGIN
:026 :EAD0210 1182 3 STRING_START = MSG_BUFFER[EMBSB_VM_NAMLANG];
:027 :EAD0210 1183 3 LENGTH_OFFSET = 1;
:028 :EAD0210 1184 3 NAME_OFFSET = 0;
:029 :EAD0210 1185 2 END;
:030 :EAD0210 1186 2 CH$MOVE (. (BBLOCK [UCB[UCBSL_DDB], DDB$T_NAME])<0,8> + .LENGTH_OFFSET,
:031 :EAD0210 1187 2 BBLOCK [UCB[UCBSL_DDB], DDB$T_NAME] + .NAME_OFFSET,
:032 :EAD0210 1188 2 STRING_START[0]);
:033 :EAD0210 1189 2
:034 :EAD0210 1190 2 STRING [0] = .STRING [0] + .(BBLOCK [UCB[UCBSL_DDB], DDB$T_NAME])<0,8>;
:035 :EAD0210 1191 2
163-3 1192 2
164 1193 2 IF .BBLOCK[UCB[UCBSL_DEVCHAR],DEV$V FOR]
165 1194 2 OR NOT .BBLOCK[UCB[UCBSL_DEVCHAR],DEV$V_SQD]
166 1195 2 THEN
167 1196 3 BEGIN
168 1197 3 LOCAL
169 1198 3 VCB : REF BBLOCK, ! address of volume control block
170 1199 3 RVT : REF BBLOCK; ! address of relative volume table
171 1200 3
172 1201 3 VCB = .UCB[UCBSL_VCB];
173 1202 3 IF .VCB[VCBSW_RVN] NEQ 0
174 1203 3 THEN
175 1204 4 BEGIN
176 1205 4 RVT = .VCB[VCBSL_RVT];
177 1206 4 MSG_BUFFER[EMBSW_VM_VOLNUM] = .VCB[VCBSW_RVN];
178 1207 4 MSG_BUFFER[EMBSW_VM_NUMSET] = .RVT[RVT$B_NVOLS];
179 1208 3 END;
180 1209 3 CH$MOVE (VCB$S_VOLNAME,
181 1210 3 BBLOCK [.UCB[UCBSL_VCB], VCB$T_VOLNAME],
182 1211 3 MSG_BUFFER[EMB$T_VM_LABEL]);
183 1212 3 END
184 1213 2 ELSE
185 1214 3 BEGIN
186 1215 3 LOCAL
187 1216 3 MVL : REF BBLOCK, ! magtape volume labels
188 1217 3 MVL_ENTRY : REF BBLOCK, ! address of label entry
189 1218 3 RUN, ! relative unit number
190 1219 3 RVT : REF BBLOCK, ! relative volume table
191 1220 3 UCBLIST : REF VECTOR, ! address of UCB list
192 1221 3 VCB : REF BBLOCK; ! volume control block
193 1222 3 VCB = .UCB[UCBSL_VCB];
194 1223 3 RVT = .VCB[VCBSL_RVT];
195 1224 3 UCBLIST = RVT[RVT$S_UCBLST];
196 1225 3 MVL = .VCB[VCBSL_MV];
197 1226 3 MSG_BUFFER[EMBSW_VM_NUMSET] = .MVL[MVL$B_NVOLS]; ! no of volumes in vol set known
198 1227 3 CH$FILL(' ',VCB$S_VOLNAME,MSG_BUFFER[EMB$T_VM_LABEL]);
199 1228 3 INCR I FROM 0 TO .RVT[RVT$B_NVOLS] - 1 DO
200 1229 4 BEGIN
201 1230 4 RUN = .I;
202 1231 4 I UCBLIST[I] EQL .UCB THEN EXITLOOP;
203 1232 3 END;
204 1233 3 MVL_ENTRY = .MVL + MVL$K_FIXLEN;
205 1234 3 INCR I FROM 0 TO .MVL[MV$B_NVOLS] - 1 DO
206 1235 4 BEGIN

```

```

: 207      1236 4      IF .MVL_ENTRY[MVL$B RVN] EQL .RUN
: 208      1237 4      AND .MVL_ENTRY[MVL$V_MOUNTED]
: 209      1238 4      THEN
: 210      1239 5      BEGIN
: 211      1240 5      MSG_BUFFER[EMBSW VM VOLNUM] = I + 1;
: 212      1241 5      CH$COPY(MVL$S VO[LBC,MVL_ENTRY[MVL$T VOLLBL], ' ',
: 213      1242 5      VCB$S_VOLNAME,MSG_BUF[RCEMBST_VM_LABEL]);
: 214      1243 5      EXITLOOP;
: 215      1244 4      END;
: 216      1245 4      MVL_ENTRY = .MVL_ENTRY + MVL$K_LENGTH;
: 217      1246 3      END;
: 218      1247 2      END;
: 219      1248 2      ! Finally release the buffer and make the entry.
: 220      1249 2      !
: 221      1250 2      ERL$RELEASEMB (.MSG_BUFFER);
: 222      1251 2      !
: 223      1252 2      RETURN 1;
: 224      1253 2      !
: 225      1254 2      !
: 226      1255 2      !
: 227      1256 1      END;

```

! end of routine SEND\_ERRLOG

					.TITLE	SNDERL	
					.IDENT	\V04-001\	
					.EXTRN	ERL\$ALLOCEMB, ERL\$RELEASEMB	
					.PSECT	\$LOCKEDC1\$,NOWRT,2	
			07FC 00000		.ENTRY	SEND_ERRLOG, Save R2,R3,R4,R5,R6,R7,R8,R9,-	1065
						R10	
	51		3E D0 00002		MOVL	#62, R1	1136
		00000000G	00 16 00005		JSB	ERL\$ALLOCEMB	
	03		50 E8 0000B		BLBS	R0, 1\$	
			0151 31 0000E		BRW	17\$	
	59		52 D0 00011	1\$:	MOVL	ADDRESS, MSG_BUFFER	1138
	07	04	AC E9 00014		BLBC	MODE, 2\$	1144
	A9	40	8F 9B 00018		MOVZBW	#64, 4(MSG_BUFFER)	1145
			05 11 0001D		BRB	3\$	
	04	A9	8F 9B 0001F	2\$:	MOVZBW	#65, 4(MSG_BUFFER)	1146
		50	AC D0 00024	3\$:	MOVL	UCB, R0	1148
		50	A0 D0 00028		MOVL	28(R0), ORB	
	10	A9	60 D0 0002C		MOVL	(ORB), 16(MSG_BUFFER)	1149
		50	AC D0 00030		MOVL	UCB, R0	1150
	14	A9	0082 C0 3C 00034		MOVZWL	130(R0), 20(MSG_BUFFER)	
		50	08 AC D0 0003A		MOVL	UCB, R0	1151
	18	A9	70 A0 D0 0003E		MOVL	112(R0), 24(MSG_BUFFER)	
		50	08 AC D0 00043		MOVL	UCB, R0	1152
	1C	A9	54 A0 B0 00047		MOVW	84(R0), 28(MSG_BUFFER)	
			2E A9 D4 0004C		CLRL	46(MSG_BUFFER)	1154
	56		1E A9 9E 0004F		MOVAB	30(R9), STRING	1157
			66 94 00053		CLRB	(STRING)	1158
		50	08 AC D0 00055		MOVL	UCB, R0	1159
		50	28 A0 D0 00059		MOVL	40(R0), DDB	
		50	34 A0 D0 0005D		MOVL	52(DDB), R0	1161
			19 13 00061		BEQL	4\$	



			44	A0	95	00063	TSTB	68(R0)	1163
			14	13	00066	BEQL	4\$		
		51	44	A0	9A	00068	MOVZBL	68(R0), R1	1166
				51	D6	0006C	INCL	R1	
66		44	A0	51	28	0006E	MOVCS	R1, 68(R0), (STRING)	1168
				66	96	00073	INCB	(STRING)	1169
		50		66	9A	00075	MOVZBL	(STRING), R0	1170
		6046		24	90	00078	MOVB	#36, (R0)[STRING]	
				66	95	0007C	TSTB	(STRING)	1173
				11	13	0007E	BEQL	5\$	
		50		66	9A	00080	MOVZBL	(STRING), R0	1176
		50		56	C0	00083	ADDL2	STRING, R0	
		52		01	A0	9E	MOVAB	1(R0), STRING_START	
				53	94	0008A	CLRB	LENGTH_OFFSET	1177
		51		01	D0	0008C	MOVL	#1, NAME_OFFSET	1178
				09	11	0008F	BRB	6\$	1173
		52		1E	A9	9E	MOVAB	30(R9), STRING_START	1182
		53		01	90	00095	MOVB	#1, LENGTH_OFFSET	1183
				51	D4	00098	CLRL	NAME_OFFSET	1184
		50		08	AC	D0	MOVL	UCB, R0	1187
		50			28	C0	ADDL2	#40, R0	
		50			60	D0	MOVL	(R0), R0	
		54		14	A0	9A	MOVZBL	20(R0), R4	
		53			53	9A	MOVZBL	LENGTH_OFFSET, R3	
		53			54	C0	ADDL2	R4, R3	
62		14	A140	53	28	000AE	MOVCS	R3, 20(NAME_OFFSET)[R0], (STRING_START)	1189
				08	AC	D0	MOVL	UCB, R0	1191
		50		28	A0	D0	MOVL	40(R0), R0	
		66		14	A0	80	ADDB2	20(R0), (STRING)	
		51		08	AC	D0	MOVL	UCB, R1	1201
		50		08	AC	D0	MOVL	UCB, R0	1193
		05		3B	A0	E8	BLBS	59(R0), 7\$	
27		38	A0	05	F0	000CC	BBS	#5, 56(R0), 9\$	1194
		50		34	A1	D0	MOVL	52(R1), VCB	1201
				0E	A0	B5	TSTW	14(VCB)	1202
				0E	13	000D8	BEQL	8\$	
		51		20	A0	D0	MOVL	32(VCB), RVT	1205
		2E	A9	0E	A0	B0	MOVW	14(VCB), 46(MSG_BUFFER)	1206
		30	A9	0B	A1	9B	MOVZBW	11(RVT), 48(MSG_BUFFER)	1207
		50		08	AC	D0	MOVL	UCB, R0	1210
		50		34	A0	D0	MOVL	52(R0), R0	
32	A9	14	A0	0C	28	000F0	MOVCS	#12, 20(R0), 50(MSG_BUFFER)	1211
				61	11	000F6	BRB	16\$	1193
		50		34	A1	D0	MOVL	52(R1), VCB	1222
		56		20	A0	D0	MOVL	32(VCB), RVT	1223
		58		44	A6	9E	MOVAB	68(R6), UCBLIST	1224
		57		34	A0	D0	MOVL	52(VCB), MVL	1225
		30	A9	0B	A7	9B	MOVZBW	11(MVL), 48(MSG_BUFFER)	1226
OC		20	6E	00	2C	0010D	MOVCS	#0, (SP), #32, #12, 50(MSG_BUFFER)	1227
				32	A9	00112			
		51		0B	A6	9A	MOVZBL	11(RVT), R1	1228
		50			01	CE	MNEGL	#1, I	1231
					0A	11	BRB	11\$	
		5A		50	D0	0011D	MOVL	I, RUN	1230
		08	AC	6840	D1	00120	CMPL	(UCBLIST)[I], UCB	1231
				04	13	00125	BEQL	12\$	
F2		50		51	F2	00127	AOBLS	R1, I, 10\$	1228

			58	24	A7	9E	0012B	12\$:	MOVAB	36(R7), MVL_ENTRY	:	1233
			57	0B	A7	9A	0012F		MOVZBL	11(MVL), R7-	:	1234
			56		01	CE	00133		MNEGL	#1, I	:	
SA	06	A8	08		1D	11	00136		BRB	15\$	:	
					00	ED	00138	13\$:	CMPZV	#0, #8, 6(MVL_ENTRY), RUN	:	1236
					12	12	0013E		BNEQ	14\$	:	
			0E	07	A8	E9	00140		BLBC	7(MVL_ENTRY), 14\$	:	1237
OC	2E	A9	56		01	A1	00144		ADDW3	#1, I, 46(MSG_BUFFER)	:	1240
		20	68		06	2C	00149		MOVCS	#6, (MVL_ENTRY), #32, #12, 50(MSG_BUFFER)	:	1242
				32	A9		0014E				:	
					07	11	00150		BRB	16\$	:	1239
			58		08	C0	00152	14\$:	ADDL2	#8, MVL_ENTRY	:	1245
		DF	56		57	F2	00155	15\$:	AOBLSS	R7, I, T3\$	:	1234
			52		59	D0	00159	16\$:	MOVL	MSG_BUFFER, R2	:	1252
					00	16	0015C		JSB	ERL\$RELEASEMB	:	
			50	00000000G	01	D0	00162	17\$:	MOVL	#1, R0	:	1254
					04	00	00165		RET		:	1256

: Routine Size: 358 bytes, Routine Base: \$LOCKEDC1\$ + 0000

```

: 228      1257  1
: 229      1258  1 END
: 230      1259  0 ELUDOM

```

PSECT SUMMARY

Name	Bytes	Attributes
\$LOCKEDC1\$	358	NOVEC,NOWRT, RD, EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)

Library Statistics

File	----- Symbols -----		Pages Mapped	Processing Time
	Total	Loaded Percent		
\$_\$255\$DUA18:[SYSLIB]LIB.L32;1	18619	56 0	1000	00:02.0

COMMAND QUALIFIERS

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

: Size: 358 code + 0 data bytes

SNDRL  
V04-001

G 11  
8-Jan-1985 18:39:15 VAX-11 Bliss-32 V4.0-742

Page 9

: Run Time: 00:15.8  
: Elapsed Time: 00:29.1  
: Lines/CPU Min: 4774  
: Lexemes/CPU-Min: 24553  
: Memory Used: 214 pages  
: Compilation Complete

0444 AH-EF71A-SE  
VAX/VMS V4.1 SRC LST MCRF UPD

Grid of source code listings for VAX/VMS V4.1, including files like LTRIVER.MAP, LTRIVER.LIS, TRUNC.LIS, RWUB.LIS, SMALOC.LIS, SNDER.LIS, and LAT.