

CCCCCCCCCCCC	LLL	IIIIIIII	UUU	UUU	TTTTTTTTTTTTTTTT	LLL
CCCCCCCCCCCC	LLL	IIIIIIII	UUU	UUU	TTTTTTTTTTTTTTTT	LLL
CCCCCCCCCCCC	LLL	IIIIIIII	UUU	UUU	TTTTTTTTTTTTTTTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCC	LLL	III	UUU	UUU	TTT	LLL
CCCCCCCCCCCC	LLLLLLLLLLLLLLLL	IIIIIIII	UUUUUUUUUUUUUU	UUUUUUUUUUUUUU	TTTT	LLLLLLLLLLLLLLLL
CCCCCCCCCCCC	LLLLLLLLLLLLLLLL	IIIIIIII	UUUUUUUUUUUUUU	UUUUUUUUUUUUUU	TTTT	LLLLLLLLLLLLLLLL
CCCCCCCCCCCC	LLLLLLLLLLLLLLLL	IIIIIIII	UUUUUUUUUUUUUU	UUUUUUUUUUUUUU	TTTT	LLLLLLLLLLLLLLLL

```

SSSSSSSS EEEEEEEEE TTTT TTTT MM MM AAAAAA IIIIII NN NN
SSSSSSSS EEEEEEEEE TTTT TTTT MM MM AAAAAA IIIIII NN NN
SS      EE      TT      MMMM MMMM AA      AA      IIIIII NN NN
SS      EE      TT      MMMM MMMM AA      AA      IIIIII NN NN
SS      EE      TT      MM MM MM AA      AA      IIIIII NN NN
      SSSSSS EEEEEEEE TT      MM MM AA      AA      IIIIII NN NN
      SSSSSS EEEEEEEE TT      MM MM AA      AA      IIIIII NN NN
      SS      EE      TT      MM MM AA      AA      IIIIII NN NN
      SS      EE      TT      MM MM AAAAAAAAAA IIIIII NN NN
      SS      EE      TT      MM MM AAAAAAAAAA IIIIII NN NN
      SS      EE      TT      MM MM AA      AA      IIIIII NN NN
      SS      EE      TT      MM MM AA      AA      IIIIII NN NN
SSSSSSSS EEEEEEEEE TT      MM MM AA      AA      IIIIII NN NN
SSSSSSSS EEEEEEEEE TT      MM MM AA      AA      IIIIII NN NN

```

```

....
....
....
....

```

```

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
LL      II      SS
LLLLLLLLLL IIIIII SSSSSSS
LLLLLLLLLL IIIIII SSSSSSS

```

```
1 0001 0 MODULE setmain (IDENT='V04-000',
2 0002 0     MAIN=set$start,
3 0003 0     ADDRESSING MODE (EXTERNAL=GENERAL,
4 0004 0     NONEXTERNAL=LONG_RELATIVE)
5 0005 0 ) =
6 0006 1 BEGIN
7 0007 1
8 0008 1
9 0009 1 *****
10 0010 1 *
11 0011 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY *
12 0012 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS. *
13 0013 1 * ALL RIGHTS RESERVED. *
14 0014 1 *
15 0015 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
16 0016 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE *
17 0017 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER *
18 0018 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
19 0019 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY *
20 0020 1 * TRANSFERRED. *
21 0021 1 *
22 0022 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE *
23 0023 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT *
24 0024 1 * CORPORATION. *
25 0025 1 *
26 0026 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *
27 0027 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL. *
28 0028 1 *
29 0029 1 *
30 0030 1 *****
31 0031 1
32 0032 1
33 0033 1 **
34 0034 1
35 0035 1 FACILITY: SET utility
36 0036 1
37 0037 1 ABSTRACT:
38 0038 1     This module contains the command processing and dispatch routines.
39 0039 1
40 0040 1 ENVIRONMENT:
41 0041 1     VAX native, user mode.
42 0042 1
43 0043 1 AUTHOR: Gerry Smith          CREATION DATE: 10-Mar-1983
44 0044 1
45 0045 1 MODIFIED BY:
46 0046 1
47 0047 1     V03-006 JLV0344          Jake VanNoy          8-APR-1984
48 0048 1     Fix problem that option=keyword caused in not matching
49 0049 1     option.
50 0050 1
51 0051 1     V03-005 BLS0291          Benn Schreiber          24-MAR-1984
52 0052 1     Move SET PASSWORD into SETPO
53 0053 1
54 0054 1     V03-004 PRB0320          Paul Beck          11-Mar-1984 12:56
55 0055 1     Add entry for SET CLUSTER
56 0056 1
57 0057 1     V03-003 GAS0175          Gerry Smith          25-Aug-1983
```

SETMAIN
V04-000

B 10
16-Sep-1984 00:43:16 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:09:10 [CLIUTL.SRC]SETMAIN.B32;1

Page 2
(1)

.. 58 0058 1 |
: 59 0059 1 |
: 60 0060 1 |
: 61 0061 1 |
: 62 0062 1 |
: 63 0063 1 |
: 64 0064 1 |
: 65 0065 1 |
: 66 0066 1 |

Add SHOW BROADCAST.

V03-002 GAS0153 Gerry Smith 7-Jul-1983
Add SET AUDIT.

V03-001 GAS0115 Gerry Smith 4-Apr-1983
Include SET LOGINS, which was inadvertently left out.

```

: 68      0067 1 LIBRARY 'SYSS$LIBRARY:STARLET';          ! VAX/VMS common definitions
: 69      0068 1
: 70      0069 1
: 71      0070 1
: 72      0071 1 Macro to set up two associated tables. The first table is a list of
: 73      0072 1 descriptor addresses. These descriptors contain the option names.
: 74      0073 1 The second table is a corresponding list of addresses of option routines.
: 75      0074 1
: 76      0075 1 If a new option is added to SET, all that is required in this
: 77      0076 1 module is to add one line of code, the option name, e.g. WORKING SET.
: 78      0077 1 Then, the name of the global routine that is dispatched to from this
: 79      0078 1 routine will be named SET$WORKING_SET.
: 80      0079 1 MACRO
: 81      0080 1
: 82      0081 1     option_name [option] = %EXACTSTRING(4, 0, option)%,
: 83      0082 1
: 84      0083 1     option_address [option] = %NAME(%STRING('set$', %STRING(option)))%,
: 85      0084 1
: 86      0085 1     option_declare [option] = %NAME(%STRING('set$', %STRING(option))) : NOVALUE%,
: 87      0086 1
: 88      M 0087 1     make_table (name) =
: 89      M 0088 1         [LITERAL %NAME(%STRING(name, 'table length')) = %LENGTH - 1;
: 90      M 0089 1         EXTERNAL ROUTINE option_declare(%REMAINING);
: 91      M 0090 1         OWN
: 92      M 0091 1             %NAME(%STRING(name, 'option')) : VECTOR[%LENGTH - 1]
: 93      M 0092 1             INITIAL (option_name(%REMAINING)),
: 94      M 0093 1
: 95      M 0094 1             %NAME(%STRING(name, 'routine')) : VECTOR[%LENGTH - 1]
: 96      0095 1             INITIAL (option_address(%REMAINING));%,
: 97      0096 1

```

```

99      0097 1 FORWARD ROUTINE
100     0098 1     set$start,           ! Main routine for SET
101     0099 1     handler;         ! Condition handler for SET
102     0100 1
103     0101 1 EXTERNAL ROUTINE
104     0102 1     lib$put_output,
105     0103 1     cli$get_value,   ! Get qualifier value
106     0104 1     cli$present;    ! Test if qualifier present
107     0105 1
108     0106 1 GLOBAL set$exit_status : $BBLOCK[4]
109     0107 1     INITIAL(1);
110     0108 1
111     0109 1
112     P 0110 1 $SHR_MSGDEF      (SET, 119, LOCAL,
113     0111 1     (badlogic, error));
114     0112 1 !
115     0113 1 ! Set up a table of all options, and another table pointing to the address
116     0114 1 ! of the routine for each option.
117     0115 1 !
118     0116 1
119     P 0117 1 make_table (set,
120     P 0118 1     accounting,
121     P 0119 1     audit,
122     P 0120 1     broadcast,
123     P 0121 1     card_reader,
124     P 0122 1     cluster,
125     P 0123 1     day,
126     P 0124 1     device,
127     P 0125 1     directory,
128     P 0126 1     file,
129     P 0127 1     login,
130     P 0128 1     magtape,
131     P 0129 1     printer,
132     P 0130 1     process,
133     P 0131 1     protection,
134     P 0132 1     rms_default,
135     P 0133 1     terminal,
136     P 0134 1     time,
137     P 0135 1     volume,
138     0136 1     working_set);
139     0137 1
140     0138 1

```

```

142 0139 1 ROUTINE set$start =
143 0140 2 BEGIN
144 0141 2
145 0142 2 :---
146 0143 2 :
147 0144 2 This is the main program. It gathers all the command inputs, and then
148 0145 2 dispatches to the appropriate routines.
149 0146 2 :
150 0147 2 :---
151 0148 2
152 0149 2 LOCAL
153 0150 2     loc_char,
154 0151 2     status,
155 0152 2     option : $BBLOCK[dsc$c_s_bln];
156 0153 2
157 0154 2 ENABLE handler;                               ! Enable the condition handler
158 0155 2
159 0156 2 :
160 0157 2 : Interrogate the CLI to determine which option one was requested, and
161 0158 2 : dispatch to the appropriate routine.
162 0159 2 :
163 0160 2 $init_dyndesc(option);
164 0161 2
165 0162 2 IF NOT (status = cli$get_value(%ASCID 'OPTION', option))
166 0163 2 THEN
167 0164 2     BEGIN
168 0165 2     SIGNAL_STOP(.status);
169 0166 2     END;
170 0167 2
171 0168 2 option[dsc$w_length] = MINU (.option[dsc$w_length], 4);
172 0169 2
173 0170 2 :
174 0171 2 : look for xxx=yyy to trim length of search string.
175 0172 2 :
176 0173 2
177 0174 2 loc_char=CH$FIND_CH (.option[dsc$w_length], .option[dsc$a_pointer], %ASCII '=');
178 0175 2 If .loc_char NEQ 0 THEN
179 0176 2     option[dsc$w_length] = .loc_char - .option[dsc$a_pointer];
180 0177 2
181 0178 2 loc_char=CH$FIND_CH (.option[dsc$w_length], .option[dsc$a_pointer], %ASCII ':');
182 0179 2 If .loc_char NEQ 0 THEN
183 0180 2     option[dsc$w_length] = .loc_char - .option[dsc$a_pointer];
184 0181 2
185 0182 2 set$exit_status = set$_badlogic;           ! if loop fails, exit with error
186 0183 2
187 0184 2 INCR index FROM 0 TO set_table_length - 1 DO
188 0185 2     BEGIN
189 0186 2     IF CH$EQL(.option[dsc$w_length], .option[dsc$a_pointer],
190 0187 2         .option[dsc$w_length], set_option[.index])
191 0188 2     THEN
192 0189 2         BEGIN
193 0190 2         set$exit_status = 1;
194 0191 2         (.set_routine[.index])();
195 0192 2         EXITLOOP
196 0193 2         END;
197 0194 2     END;
198 0195 2

```

```

: 199      0196 2 IF .set$exit_status NEO set$_badlogic
: 200      0197 2     THEN
: 201      0198 2         set$exit_status = .set$exit_status OR sts$m_inhib_msg;
: 202      0199 2
: 203      0200 2 RETURN .set$exit_status;      ! Exit
: 204      0201 1 END;

```

```

.TITLE SETMAIN
.IDENT \V04-000\
.PSECT $SPLITS$,NOWRT,NOEXE,2

```

```

00 00 4E 4F 49 54 50 4F 00000 P.AAB: .ASCII \OPTION\<0><0>
010E0006 00008 P.AAA: .LONG 17694726
00000000' 0000C .ADDRESS P.AAB

```

```

.PSECT $OWNS$,NOEXE,2

```

```

4F 43 43 41 00000 SET_OPTION:
49 44 55 41 00004 .ASCII \ACCO\
41 4F 52 42 00008 .ASCII \AUDI\
44 52 41 43 0000C .ASCII \BROA\
53 55 4C 43 00010 .ASCII \CARD\
00 59 41 44 00014 .ASCII \CLUS\
49 56 45 44 00018 .ASCII \DAY\<0>
45 52 49 44 0001C .ASCII \DEVI\
45 4C 49 46 00020 .ASCII \DIRE\
49 47 4F 4C 00024 .ASCII \FILE\
54 47 41 4D 00028 .ASCII \LOGI\
4E 49 52 50 0002C .ASCII \MAGT\
43 4F 52 50 00030 .ASCII \PRIN\
54 4F 52 50 00034 .ASCII \PROG\
5F 53 4D 52 00038 .ASCII \PROT\
4D 52 45 54 0003C .ASCII \RMS \
45 4D 49 54 00040 .ASCII \TERM\
55 4C 4F 56 00044 .ASCII \TIME\
48 52 4F 57 00048 .ASCII \VOLU\
                     .ASCII \WORK\

```

```

00000000G 00000000G 00000000G 00000000G 00000000G 00000000G 0004C SET_ROUTINE:
00000000G 00000000G 00000000G 00000000G 00000000G 00000000G 00064
00000000G 00000000G 00000000G 00000000G 00000000G 00000000G 0007C
                     00000000G 00094

```

```

.ADDRESS SET$ACCOUNTING, SET$AUDIT, -
          SET$BROADCAST, SET$CARD READER, -
          SET$CLUSTER, SET$DAY, SET$DEVICE, -
          SET$DIRECTORY, SET$FILE, SET$LOGIN, -
          SET$MAGTAPE, SET$PRINTER, SET$PROCESS, -
          SET$PROTECTION, SET$RMS_DEFAULT, -
          SET$TERMINAL, SET$TIME, SET$VOLUME, -
          SET$WORKING_SET

```

```

.PSECT $GLOBAL$,NOEXE,2

```

```

00000001 00000 SET$EXIT_STATUS:;
.LONG 1

```

```

.EXTRN LIB$PUT OUTPUT, CLISGET VALUE
.EXTRN CLISPRESENT, SET$ACCOUNTING
.EXTRN SET$AUDIT, SET$BROADCAST

```



```
.EXTRN SET$CARD READER
.EXTRN SET$CLUSTER, SET$DAY
.EXTRN SET$DEVICE, SET$DIRECTORY
.EXTRN SET$FILE, SET$LOGIN
.EXTRN SET$MAGTAPE, SET$PRINTER
.EXTRN SET$PROCESS, SET$PROTECTION
.EXTRN SET$RMS DEFAULT
.EXTRN SET$TERMINAL, SET$TIME
.EXTRN SET$VOLUME, SET$WORKING_SET
```

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

```
003C 00000 SET$START:
```

					.WORD	Save R2,R3,R4,R5	: 0139
	55	00000000'	EF	9E 00002	MOVAB	SET\$EXIT_STATUS, R5	
	5E		04	C2 00009	SUBL2	#4, SP	
	6D	0094	CF	DE 0000C	MOVAL	11\$, (FP)	: 0140
		020E0000	8F	DD 00011	PUSHL	#34471936	: 0160
		04	AE	D4 00017	CLRL	OPTION+4	
			5E	DD 0001A	PUSHL	SP	: 0162
		00000000'	EF	9F 0001C	PUSHAB	P.AAA	
	00000000G	00	02	FB 00022	CALLS	#2, CLISGET_VALUE	
		09	50	EB 00029	BLBS	STATUS, 1\$	
			50	DD 0002C	PUSHL	STATUS	: 0165
	00000000G	00	01	FB 0002E	CALLS	#1, LIB\$STOP	
		50	6E	3C 00035	MOVZWL	OPTION, R0	: 0168
		04	50	B1 00038	CMPW	R0, #4	
			03	1B 0003B	BLEQU	2\$	
		50	04	D0 0003D	MOVL	#4, R0	
		6E	50	B0 00040	MOVW	R0, OPTION	
04	BE	6E	3D	3A 00043	LOCC	#61, OPTION, @OPTION+4	: 0174
			C2	12 00048	BNEQ	3\$	
			51	D4 0004A	CLRL	R1	
			51	D5 0004C	TSTL	LOC_CHAR	: 0175
			05	13 0004E	BEQL	4\$	
	04	6E	51	A3 00050	SUBW3	OPTION+4, LOC_CHAR, OPTION	: 0176
	BE	6E	3A	3A 00055	LOCC	#58, OPTION, @OPTION+4	: 0178
			02	12 0005A	BNEQ	5\$	
			51	D4 0005C	CLRL	R1	
			51	D5 0005E	TSTL	LOC_CHAR	: 0179
			05	13 00060	BEQL	6\$	
	6E	51	AE	A3 00062	SUBW3	OPTION+4, LOC_CHAR, OPTION	: 0180
		65	8F	D0 00067	MOVL	#7803170, SET\$EXIT_STATUS	: 0182
			54	D4 0006E	CLRL	INDEX	: 0186
		00000000'	EF	44 DF 00070	PUSHAL	SET OPTION[INDEX]	: 0187
	9E	08	BE	AE 29 00077	CMPC3	OPTION, @OPTION+4, @(SP)+	
			10	12 0007D	BNEQ	8\$	
		65	01	D0 0007F	MOVL	#1, SET\$EXIT_STATUS	: 0190
		50	EF	44 D0 00082	MOVL	SET_ROUTINE[INDEX], R0	: 0191
		60	00	FB 0008A	CALLS	#0, -(R0)	
			04	11 0008D	BRB	9\$: 0189
	DD	00771122	54	12 F3 0008F	AOBLEQ	#18, INDEX, 7\$: 0184
			8F	65 D1 00093	CMPL	SET\$EXIT_STATUS, #7803170	: 0196
			04	13 0009A	BEQL	10\$	
		03	A5	10 88 0009C	BISB2	#16, SET\$EXIT_STATUS+3	: 0198
			50	65 D0 000A0	MOVL	SET\$EXIT_STATUS, R0	: 0200
			04	000A3	RET		: 0201


```

: 206      0202 1 ROUTINE handler (sigargs, mechargs) =
: 207      0203 2 BEGIN
: 208      0204 2
: 209      0205 2 |---
: 210      0206 2
: 211      0207 2 | This routine is a condition handler established by the main
: 212      0208 2 | routine. It saves the most severe condition for the exit status.
: 213      0209 2
: 214      0210 2 |---
: 215      0211 2
: 216      0212 2 MAP
: 217      0213 2     sigargs : REF $BBLOCK,
: 218      0214 2     mechargs : REF $BBLOCK;
: 219      0215 2 BIND
: 220      0216 2     signame = sigargs[chf$_sig_name] : $BBLOCK;           ! Name of signal
: 221      0217 2
: 222      0218 2
: 223      0219 2 IF .set$exit_status EQL 1                                ! If no errors yet, use
: 224      0220 2 THEN set$exit_status = .signame;                       ! this one.
: 225      0221 2
: 226      0222 2 IF NOT .signame                                         ! If an error signal
: 227      0223 2 AND .signame[sts$v_severity]                             ! and severity is worse
: 228      0224 2 GTRU . $BBLOCK[set$exit_status, sts$v_severity] ! than current saved severity
: 229      0225 2 THEN set$exit_status = .signame;                       ! then save it for exit
: 230      0226 2
: 231      0227 2 RETURN ss$_resignal;                                     ! Resignal to get message
: 232      0228 1 END;

```

				0004	C0000	HANDLER:	.WORD	Save R2		: 0202
		52	00000000'	EF	9E	00002	MOVAB	SET\$EXIT_STATUS, R2		: 0216
	50	04		04	C1	00009	ADDL3	#4, SIGARGS, R0		: 0219
				62	D1	00C0E	CMPL	SET\$EXIT_STATUS, #1		: 0220
				03	12	00011	BNEQ	1\$: 0222
		62		60	D0	00013	MOVL	(R0), SET\$EXIT_STATUS		: 0224
		0F		60	E8	00016	1\$:	BLBS	(R0), 2\$: 0225
51	62	03		00	EF	00019	EXTZV	#0, #3, SET\$EXIT_STATUS, R1		: 0227
51	60	03		00	ED	0001E	CMPL	#0, #3, (R0), R1		: 0228
				03	1B	00023	BLEQU	2\$: 0228
		62		60	D0	00025	MOVL	(R0), SET\$EXIT_STATUS		: 0227
		50	0918	8F	3C	00028	2\$:	MOVZWL	#2328, R0	: 0228
				04	00	0002D	RET			

; Routine Size: 46 bytes, Routine Base: \$CODE\$ + 00B6

: 234
: 235 0229 1 END
 0230 0 ELUDOM

.EXTRN LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes
\$GLOBALS	4	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$OWNS	152	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$SPLITS	16	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODES	228	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Tota	----- Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	97 6	18	0	581	00:01.0

COMMAND QUALIFIERS

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:SETMAIN/OBJ=OBJ\$:SETMAIN MSRCS:SETMAIN/UPDATE=(ENHS:SETMAIN)

: Size: 228 code + 172 data bytes
: Run Time: 00:06.9
: Elapsed Time: 00:23.6
: Lines/CPU Min: 2000
: Lexemes/CPU-Min: 23956
: Memory Used: 79 pages
: Compilation Complete

0053 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

The image displays a grid of 140 terminal windows, arranged in 10 rows and 14 columns. Each window shows a different terminal session, likely from a VAX/VMS system. The sessions contain various system commands and their outputs, including:

- SETFILE LIS**: Located in the top-left window.
- SETPOMESS LIS**: Located in the 5th row, 13th column window.
- SETP001SP LIS**: Located in the 7th row, 13th column window.
- SETMISC LIS**: Located in the 8th row, 11th column window.
- SETPRO LIS**: Located in the 9th row, 13th column window.
- SETMAIN LIS**: Located in the 10th row, 11th column window.

Other windows show various system messages, command prompts, and data listings. The text is small and dense, typical of a terminal window output.