

```

PPPPPPPPPPPP    HHH        HHH        000000000    NNN        NNN    EEEEEEEEEEEEEEEEE
PPPPPPPPPPPPPP    HHH        HHH        000000000    NNN        NNN    EEEEEEEEEEEEEEEEE
PPPPPPPPPPPPPP    HHH        HHH        000000000    NNN        NNN    EEEEEEEEEEEEEEEEE
PPP                PPP    HHH        HHH        000        000    NNN        NNN    EEE
PPP                PPP    HHH        HHH        000        000    NNN        NNN    EEE
PPP                PPP    HHH        HHH        000        000    NNN        NNN    EEE
PPP                PPP    HHH        HHH        000        000    NNNNNN    NNN    EEE
PPP                PPP    HHH        HHH        000        000    NNNNNN    NNN    EEE
PPP                PPP    HHH        HHH        000        000    NNNNNN    NNN    EEE
PPPPPPPPPPPPPP    HHHHHHHHHHHHHHHHHHH    000        000    NNN        NNN    EEEEEEEEEEEEE
PPPPPPPPPPPPPP    HHHHHHHHHHHHHHHHHHH    000        000    NNN        NNN    EEEEEEEEEEEEE
PPPPPPPPPPPPPP    HHHHHHHHHHHHHHHHHHH    000        000    NNN        NNN    EEEEEEEEEEEEE
PPP                HHH        HHH        000        000    NNN        NNNNNN    EEE
PPP                HHH        HHH        000        000    NNN        NNNNNN    EEE
PPP                HHH        HHH        000        000    NNN        NNNNNN    EEE
PPP                HHH        HHH        000        000    NNN        NNN        EEE
PPP                HHH        HHH        000        000    NNN        NNN        EEE
PPP                HHH        HHH        000        000    NNN        NNN        EEE
PPP                HHH        HHH        000        000    NNN        NNN        EEE
PPP                HHH        HHH        000000000    NNN        NNN    EEEEEEEEEEEEEEEEE
PPP                HHH        HHH        000000000    NNN        NNN    EEEEEEEEEEEEEEEEE
PPP                HHH        HHH        000000000    NNN        NNN    EEEEEEEEEEEEEEEEE

```

[illegible]

```

MM      MM      IIIIII      SSSSSSSS      CCCCCCCC      CCCCCCCC      MM      MM      DDDDDDDD      SSSSSSSS
MM      MM      IIIIII      SSSSSSSS      CCCCCCCC      CCCCCCCC      MM      MM      DDDDDDDD      SSSSSSSS
MMMM    MMMM      II      SS      CC      CC      MMMM    MMMM      DD      DD      SS
MMMM    MMMM      II      SS      CC      CC      MMMM    MMMM      DD      DD      SS
MM      MM      II      SS      CC      CC      MM      MM      DD      DD      SS
MM      MM      II      SSSSSS      CC      CC      MM      MM      DD      DD      SSSSSS
MM      MM      II      SSSSSS      CC      CC      MM      MM      DD      DD      SSSSSS
MM      MM      II      SS      CC      CC      MM      MM      DD      DD      SS
MM      MM      II      SS      CC      CC      MM      MM      DD      DD      SS
MM      MM      II      SS      CC      CC      MM      MM      DD      DD      SS
MM      MM      IIIIII      SSSSSSSS      CCCCCCCC      CCCCCCCC      MM      MM      DDDDDDDD      SSSSSSSS
MM      MM      IIIIII      SSSSSSSS      CCCCCCCC      CCCCCCCC      MM      MM      DDDDDDDD      SSSSSSSS

```

....


```

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

```



```
0001 0 %title 'MISCCMDS - Miscellaneous Phone Commands'
0002 0
0003 1 module misccmds (
0004 1 ident='V04-000') = begin
0005 1
0006 1 *****
0007 1 *
0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0010 1 * ALL RIGHTS RESERVED.
0011 1 *
0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0017 1 * TRANSFERRED.
0018 1 *
0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0021 1 * CORPORATION.
0022 1 *
0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0025 1 *
0026 1 *
0027 1 *****
0028 1
0029 1
0030 1 ++
0031 1 Facility: VAX/VMS Telephone Facility, Miscellaneous Phone Commands
0032 1
0033 1 Abstract: This module handles the following user commands:
0034 1 EXIT Exit the facility.
0035 1 HELP Provide help information.
0036 1 DIRECTORY Provide a list of people to call.
0037 1 MAIL Send phone message to someone.
0038 1
0039 1
0040 1 Environment:
0041 1
0042 1 Author: Paul C. Anagnostopoulos, Creation Date: 20 November 1980
0043 1
0044 1 Modified By:
0045 1
0046 1 V03-007 ROP0034 Robert Posniak 08-Aug-1984
0047 1 Pass a null input file to MAIL when sending
0048 1 a phone message to someone.
0049 1
0050 1 V03-006 BLS0335 Benn Schreiber 27-JUL-1984
0051 1 Include SYSNAM in privs dropped.
0052 1
0053 1 V03-005 BLS0241 Benn Schreiber 8-Dec-1983
0054 1 Drop privs before spawning MAIL.
0055 1
0056 1 V03-004 MHB0085 Mark Bramhall 17-Jan-1983
0057 1 PHN$ESTAB_LINK now always displays its status via
```

MISCCMDS
V04-000

MISCCMDS - Miscellaneous Phone Commands

1 6
16-Sep-1984 02:13:32
14-Sep-1984 12:53:27

VAX-11 Bliss-32 V4.0-742
[PHONE.SRC]MISCCMDS.B32;1

Page 2
(1)

```

: 58      0058 1 | PHN$INFORM; its callers just check the returned status.
: 59      0059 1 |
: 60      0060 1 | V03-003 BLS0169      Benn Schreiber      12-Apr-1982
: 61      0061 1 |      Fix error handling from lib$spawn call
: 62      0062 1 |
: 63      0063 1 | V03-002 PCA0043      Paul Anagnostopoulos  26-Mar-1982
: 64      0064 1 |      Remove DIRECTORY feature that shows if user is running
: 65      0065 1 |      PHONE. This required inswap of every process.
: 66      0066 1 |
: 67      0067 1 | V03-001 PCA0042      Paul Anagnostopoulos  26-Mar-1982
: 68      0068 1 |      Use LIB$SPAWN to fire up MAIL subprocess.
: 69      0069 1 | --

```



```
: 71 0070 1 %sbttl 'Module Declarations'
: 72 0071 1
: 73 0072 1 | Libraries and Requires:
: 74 0073 1 |
: 75 0074 1 |
: 76 0075 1 library 'sys$library:starlet.l32';
: 77 0076 1 library 'sys$library:tpamac.l32';
: 78 0077 1 require 'phonereq';
: 79 0406 1
: 80 0407 1 |
: 81 0408 1 | Table of Contents:
: 82 0409 1 |
: 83 0410 1 |
: 84 0411 1 forward routine
: 85 0412 1     phn$exit_cmd: novalue,
: 86 0413 1     phn$exit_handler: novalue,
: 87 0414 1     phn$help_cmd: novalue,
: 88 0415 1     phn$help2: novalue,
: 89 0416 1     phn$directory_cmd: novalue,
: 90 0417 1     phn$directory2: novalue,
: 91 0418 1     phn$directory_line,
: 92 0419 1     phn$mail_cmd: novalue;
: 93 0420 1
: 94 0421 1 |
: 95 0422 1 | External References:
: 96 0423 1 |
: 97 0424 1 |
: 98 0425 1 external routine
: 99 0426 1     lbr$close: addressing_mode(general),
100 0427 1     lbr$get_help: addressing_mode(general),
101 0428 1     lbr$ini_control: addressing_mode(general),
102 0429 1     lbr$open: addressing_mode(general),
103 0430 1     lib$spawn: addressing_mode(general),
104 0431 1     lib$tparse: addressing_mode(general),
105 0432 1     phn$break_link,
106 0433 1     phn$term_characteristic,
107 0434 1     phn$establish_link,
108 0435 1     phn$inform,
109 0436 1     phn$kill_ctl,
110 0437 1     phn$make_ctl,
111 0438 1     phn$make_tsb,
112 0439 1     phn$prepare_users_target,
113 0440 1     phn$queue_smb,
114 0441 1     phn$read_slave,
115 0442 1     phn$scroll_line,
116 0443 1     phn$scroll_prep,
117 0444 1     phn$send_smb,
118 0445 1     scr$erase_page: addressing_mode(general);
119 0446 1
120 0447 1 external literal
121 0448 1     lib$_syntaxerr;
122 0449 1
123 0450 1 |
124 0451 1 | Own Variables:
125 0452 1 |
126 0453 1 | The following is the head of the help save queue. It is used by
: 127 0454 1 | PHN$HELP_CMD and PHN$HELP2.
```

MISCCMDS
V04-000

MISCCMDS - Miscellaneous Phone Commands
Module Declarations

K C
16-Sep-1984 02:13:32
14-Sep-1984 12:53:27

VAX-11 Bliss-32 V4.0-742
[PHONE.SRC]MISCCMDS.B32;1

Page 4
(2)

```
: 128      0455 1
: 129      0456 1 own
: 130      0457 1      help_save_head: vector[2,long]
: 131      0458 1      initial(rep 2 of (help_save_head));
: 132      0459 1
```


MISCCMDS
V04-000

MISCCMDS - Miscellaneous Phone Commands
PHN\$EXIT_CMD - Handle EXIT Command

L 6
16-Sep-1984 02:13:32
14-Sep-1984 12:53:27

VAX-11 Bliss-32 V4.0-742
[PHONE.SRC]MISCCMDS.B32;1

Page 5
(3)

```
: 134      0460 1 %sbttl 'PHN$EXIT_CMD - Handle EXIT Command'
: 135      0461 1 ++
: 136      0462 1 Functional Description:
: 137      0463 1 This routine handles the EXIT command, which is used to exit
: 138      0464 1 the facility.
: 139      0465 1
: 140      0466 1 Formal Parameters:
: 141      0467 1 none
: 142      0468 1
: 143      0469 1 Implicit Inputs:
: 144      0470 1 global data
: 145      0471 1
: 146      0472 1 Implicit Outputs:
: 147      0473 1 global data
: 148      0474 1
: 149      0475 1 Returned Value:
: 150      0476 1 none
: 151      0477 1
: 152      0478 1 Side Effects:
: 153      0479 1
: 154      0480 1 --
: 155      0481 1
: 156      0482 1
: 157      0483 2 global routine phn$exit_cmd: novalue = begin
: 158      0484 2
: 159      0485 2
: 160      0486 2 $exit(code=ss$_normal);
: 161      0487 2
: 162      0488 1 end;
```

```
.TITLE MISCCMDS MISCCMDS - Miscellaneous Phone Command
.IDENT \V04-000\
.PSECT $OWN$,NOEXE,2
```

00000000' 00000 HELP_SAVE_HEAD:

00000000' 00004 .ADDRESS HELP_SAVE_HEAD

```
.EXTRN PHN$_OK, PHN$_ANSWERED
.EXTRN PHN$_BUSYCALL, PHN$_CANCALL
.EXTRN PHN$_CANTREACH, PHN$_CONFCALL
.EXTRN PHN$_DEAD, PHN$_DECNETLINK
.EXTRN PHN$_DIRCAN, PHN$_FACSCAN
.EXTRN PHN$_HELPCAN, PHN$_HUNGUP
.EXTRN PHN$_JUSTRANG, PHN$_LOGGEDOFF
.EXTRN PHN$_REJECTED, PHN$_RING
.EXTRN PHN$_REJECTJUNK
.EXTRN PHN$_SENDINGMAIL
.EXTRN PHN$_BADCMD, PHN$_BADHELP
.EXTRN PHN$_BADMAILCMD
.EXTRN PHN$_BADSMB, PHN$_BADSPEC
.EXTRN PHN$_HELPMISSING
.EXTRN PHN$_IVREDUNANS
.EXTRN PHN$_IVREDUNCALL
```

MISCCMDS
V04-000

MISCCMDS - Miscellaneous Phone Commands
PHN\$EXIT_CMD - Handle EXIT Command

M 6
16-Sep-1984 02:13:32
14-Sep-1984 12:53:27

VAX-11 Bliss-32 V4.0-742
[PHONE.SRC]MISCCMDS.B32;1

Page 6
(3)

```
.EXTRN PHN$_LINKERROR, PHN$_NEEDUSER
.EXTRN PHN$_NOCALL, PHN$_NOROLDS
.EXTRN PHN$_NOPTS, PHN$_NOPRIV
.EXTRN PHN$_NOPROC, PHN$_NOTCONV
.EXTRN PHN$_ONLYNODE, PHN$_PHONEBUSY
.EXTRN PHN$_REMOTEERROR
.EXTRN PHN$_TARGTERM, PHN$_UNPLUGGED
.EXTRN PHN$_BADTERM, PHN$_SHAREDMBX
.EXTRN PHN$_INPUTTERM, PHN$_GQ_NODE_NAME
.EXTRN PHN$_GQ_SWITCH_HOOK
.EXTRN PHN$_GQ_VIEWPORT_SIZE
.EXTRN PHN$_GB_SCROLL, PHN$_GQ_PUBHEAD
.EXTRN PHN$_GB_FLAGS, LBR$_CLOSE
.EXTRN LBR$_GET_HELP, LBR$_INI_CONTROL
.EXTRN LBR$_OPEN, LIB$_SPAWN
.EXTRN LIB$_PARSE, PHN$_BREAK_LINK
.EXTRN PHN$_TERM_CHARACTERISTIC
.EXTRN PHN$_ESTAB_LINK, PHN$_INFORM
.EXTRN PHN$_KILL_CTL, PHN$_MAKE_CTL
.EXTRN PHN$_MAKE_TSB, PHN$_PREPARE_USERS_TARGET
.EXTRN PHN$_QUEUE_SMB, PHN$_READ_SCAVE
.EXTRN PHN$_SCROLL_LINE
.EXTRN PHN$_SCROLL_PREP
.EXTRN PHN$_SEND_SMB, SCR$_ERASE_PAGE
.EXTRN LIB$_SYNTAXERR, SYS$EXIT
```

.PSECT \$CODE\$,NOWRT,2

```
.ENTRY PHN$EXIT_CMD, Save nothing
PUSHL #1
CALLS #1, SYS$EXIT
RET
```

: 0483
: 0486
: 0488

00000000G 00

0000 00000
01 DD 00002
01 FB 00004
04 0000B

; Routine Size: 12 bytes, Routine Base: \$CODE\$ + 0000


```
164 0489 1 %sbttl 'PHN$EXIT_HANDLER - Exit the Facility'
165 0490 1 ++
166 0491 1 Functional Description:
167 0492 1 This routine is the exit handler established by the initialization
168 0493 1 routine. It is called whenever the user exits, be it by the EXIT
169 0494 1 command, CTRL/C, or CTRL/Y. We have to hang up the phone, so
170 0495 1 everyone we're talking to knows what's happening.
171 0496 1
172 0497 1 Formal Parameters:
173 0498 1 none
174 0499 1
175 0500 1 Implicit Inputs:
176 0501 1 global data
177 0502 1
178 0503 1 Implicit Outputs:
179 0504 1 global data
180 0505 1
181 0506 1 Returned Value:
182 0507 1 none
183 0508 1
184 0509 1 Side Effects:
185 0510 1
186 0511 1 --
187 0512 1
188 0513 1
189 0514 2 global routine phn$exit_handler: novalue = begin
190 0515 2
191 0516 2 local
192 0517 2 p: ref pub, p2: ref pub;
193 0518 2
194 0519 2
195 0520 2 ! We begin by scanning the entire PUB chain (except our own) and hanging
196 0521 2 ! up on everyone.
197 0522 2
198 0523 2 p = ..phn$gq_pubhead[0];
199 0524 2 until .p eqla phn$gq_pubhead do (
200 0525 2 p2 = .p[pub_l_flink];
201 0526 2
202 0527 2 phn$break_link(.p,smb__hungup);
203 0528 2
204 0529 2 p = .p2;
205 0530 2 );
206 0531 2
207 0532 2 ! Finally, we clear the screen so the user can start fresh in DCL.
208 0533 2
209 0534 2 scr$erase_page(1,1);
210 0535 2 return;
211 0536 2
212 0537 1 end;
```

```
52 0000G DF D0 00002
50 0000G CF 9E 00007 1$:
```

```
.ENTRY PHN$EXIT_HANDLER, Save R2,R3
MOVL @PHN$GQ_PUBHEAD, P
MOVAB PHN$GQ_PUBHEAD, R0
```

```
: 0514
: 0523
: 0524
```

MISCCMDS
V04-000

MISCCMDS - Miscellaneous Phone Commands
PHN\$EXIT_HANDLER - Exit the Facility

B 7
16-Sep-1984 02:13:32
14-Sep-1984 12:53:27

VAX-11 Bliss-32 V4.0-742
[PHONE.SRC]MISCCMDS.B32;1

Page 8
(4)

50	52	D1	0000C	CMPL	P, R0	:
	11	13	0000F	BEQL	2\$	:
53	62	D0	00011	MOVL	(P), P2	: 0525
	09	DD	00014	PUSHL	#9	: 0527
	52	DD	00016	PUSHL	P	:
0000G CF	02	FB	00018	CALLS	#2, PHN\$BREAK_LINK	:
52	53	D0	0001D	MOVL	P2, P	: 0529
	E5	11	00020	BRB	1\$	: 0524
	01	DD	00022	PUSHL	#1	: 0534
	01	DD	00024	PUSHL	#1	:
00000000G 00	02	FB	00026	CALLS	#2, SCR\$ERASE_PAGE	:
	04	0002D	RET			: 0537

; Routine Size: 46 bytes, Routine Base: \$CODE\$ + 000C

MIS
V04

; R


```
214 0538 1 %sbtcl 'PHN$HELP_CMD - Handle HELP Command'
215 0539 1 ++
216 0540 1 Functional Description:
217 0541 1 This routine handles the HELP command. We get all the information
218 0542 1 lines from the help library and store them in memory. Then we
219 0543 1 queue a steering message that will make us display the first
220 0544 1 line. Each line displayed then queues a message for the next
221 0545 1 line.
222 0546 1
223 0547 1 Formal Parameters:
224 0548 1 parameters Address of descriptor of the rest of the help
225 0549 1 command as entered by the user.
226 0550 1
227 0551 1 Implicit Inputs:
228 0552 1 global data
229 0553 1
230 0554 1 Implicit Outputs:
231 0555 1 global data
232 0556 1
233 0557 1 Returned Value:
234 0558 1 none
235 0559 1
236 0560 1 Side Effects:
237 0561 1
238 0562 1 --
239 0563 1
240 0564 1
241 0565 2 global routine phn$help_cmd(parameters): novalue = begin
242 0566 2
243 0567 2 bind
244 0568 2 parameters_dsc = .parameters: descriptor;
245 0569 2
246 0570 2 own
247 0571 2 topic_dsc: descriptor,
248 0572 2 subtopic_dsc: descriptor,
249 0573 2 tparse_block: block[tpa$k_length0,byte]
250 0574 2 initial(tpa$k_count0,
251 0575 2 tpa$m_abbrev);
252 0576 2
253 0577 2 own
254 0578 2 library_index: long,
255 0579 2 library_nam: $nam();
256 0580 2
257 0581 2 local
258 0582 2 status: long;
```

```

: 260      0583 2 ! The following is the parsing table used to parse the help parameters.
: 261      0584 2 ! We accept all of the standard VMS help parameter formats.
: 262      0585 2
: 263      0586 2 $init_state(help_state,help_key);
: 264      0587 2
: 265      P 0588 2 $state (,
: 266      P 0589 2      (tpa$_eos,tpa$_exit),
: 267      P 0590 2      ((topic),,,,topic_dsc)
: 268      0591 2 );
: 269      P 0592 2 $state (,
: 270      P 0593 2      (tpa$_eos,tpa$_exit),
: 271      P 0594 2      ((topic),tpa$_exit,,,subtopic_dsc)
: 272      0595 2 );
: 273      0596 2
: 274      P 0597 2 $state (topic,
: 275      P 0598 2      ('/'),
: 276      P 0599 2      (tpa$_lambda)
: 277      0600 2 );
: 278      P 0601 2 $state (,
: 279      P 0602 2      ('*'),
: 280      P 0603 2      (tpa$_symbol)
: 281      0604 2 );
: 282      P 0605 2 $state (,
: 283      P 0606 2      ('.'),
: 284      P 0607 2      (tpa$_lambda,tpa$_exit)
: 285      0608 2 );
: 286      P 0609 2 $state (,
: 287      P 0610 2      ('.')
: 288      0611 2 );
: 289      P 0612 2 $state (,
: 290      P 0613 2      ('.',tpa$_exit)
: 291      0614 2 );

```



```
293 0615 2 : This little routine is called by the librarian for each line it extracts
294 0616 2 : from the help library. We make a queue of these lines, using CTL buffers,
295 0617 2 : so that we can display them later.
296 0618 2
297 0619 2 routine help_save(line) = begin
298 0620 2
299 0621 2 local
300 0622 2     c: ref ctl;
301 0623 2
302 0624 2 phn$make_ctl(.line,c);
303 0625 2
304 0626 2 insque(.c,.help_save_head[1]);
305 0627 2 return ss$_normal;
306 0628 2
307 0629 2 end;
```

```
                .PSECT  _LIB$STATES$,NOWRT,  SHR,  PIC,1
00000 HELP_STATE::
11F7 00000 :TPA$TYPE      .BLKB      0
      U.2:      .WORD      4599
FFFF 00002 :TPA$TARGET
      U.3:      .WORD      -1
4DF8 00004 :TPA$TYPE      .WORD      19960
0000* 00006 :TPA$SUBEXP
      U.6:      .WORD      <<U.5-U.6>-2>
00000000* 00008 :TPA$ADDR
      U.7:      .LONG      <<TOPIC_DSC-U.7>-4>
11F7 0000C :TPA$TYPE      .WORD      4599
FFFF 0000E :TPA$TARGET
      U.9:      .WORD      -1
5DF8 00010 :TPA$TYPE      .WORD      24056
0000* 00012 :TPA$SUBEXP
      U.11:     .WORD      <<U.5-U.11>-2>
00000000* 00014 :TPA$ADDR
      U.12:     .LONG      <<SUBTOPIC_DSC-U.12>-4>
FFFF 00018 :TPA$TARGET
      U.13:     .WORD      -1
0001A :TOPIC
      U.5:      .BLKB      0
002F 0001A :TPA$TYPE      .WORD      47
05F6 0001C :TPA$TYPE      .WORD      1526
002A 0001E :TPA$TYPE      .WORD      42
05F1 00020 :TPA$TYPE      .WORD      1521
002E 00022 :TPA$TYPE      .WORD      46
15F6 00024 :TPA$TYPE
```

MISCCMDS - Miscellaneous Phone Commands
PHN\$HELP_CMD - Handle HELP Command

F 7
16-Sep-1984 02:13:32
14-Sep-1984 12:53:27

VAX-11 Bliss-32 V4.0-742
[PHONE.SRC]MISCCMDS.B32;1

Page 12
(7)

MI
VO

```

      FFFF 00026 U.19: .WORD 5622
             ;TPAS$TARGET
      042E 00028 U.20: .WORD -1
             ;TPAS$TYPE
      142E 0002A U.21: .WORD 1070
             ;TPAS$TYPE
      FFFF 0002C U.22: .WORD 5166
             ;TPAS$TARGET
             U.23: .WORD -1

             .PSECT _LIB$KEY0$,NOWRT, SHR, PIC,1

      00000 HELP_KEY:
             .BLKB 0
      00000 ;TPAS$KEY0
             U.1: .BLKB 0

             .PSECT $OWN$,NOEXE,2

      00008 TOPIC_DSC:
             .BLKB 8
      00010 SUBTOPIC_DSC:
             .BLKB 8
00000002 00000008 00018 TPARSE_BLOCK:
             .LONG 8, 2
      00020 .BLKB 28
      0003C LIBRARY_INDEX:
             .BLKB 4
      02 00040 LIBRARY_NAM:
             .BYTE 2
      60 00041 .BYTE 96
      00 00042 .BYTE 0
      00 00043 .BYTE 0
00000000 00044 .LONG 0
      00 00048 .BYTE 0
      00 00049 .BYTE 0
      00 0004A .BYTE 0
      00 0004B .BYTE 0
00000000 0004C .LONG 0
00000000 00050 .LONG 0
      0000# 00054 .WORD 0[8]
      0000# 00064 .WORD 0[3]
      0000# 0006A .WORD 0[3]
00000000 00070 .LONG 0
00000000 00074 .LONG 0
      00 00078 .BYTE 0
      00 00079 .BYTE 0
      00 0007A .BYTE 0
      00 0007B .BYTE 0
      00 0007C .BYTE 0
      00 0007D .BYTE 0
      00# 0007E .BYTE 0[2]
00000000 00080 .LONG 0
00000000 00084 .LONG 0
00000000 00088 .LONG 0
00000000 0008C .LONG 0
00000000 00090 .LONG 0

```


MISCCMDS
V04-000

MISCCMDS - Miscellaneous Phone Commands
PHN\$HELP_CMD - Handle HELP Command

G 7
16-Sep-1984 02:13:32
14-Sep-1984 12:53:27

VAX-11 Bliss-32 V4.0-742
[PHONE.SRC]MISCCMDS.B32;1

Page 13
(7)

00000000 00094
00000000# 00098

.LONG 0
.LONG 0[2]

:

.PSECT \$CODE\$,NOWRT,2

0000 00000 HELP_SAVE:

SE 04 C2 00002
04 SE DD 00005
00 AC DD 00007
0000G CF 02 FB 0000A
0000 DF 00 BE 0E 0000F
50 01 D0 00015
04 00018

.WORD Save nothing
SUBL2 #4, SP
PUSHL SP
PUSHL LINE
CALLS #2, PHN\$MAKE_CTL
INSQUE @C, @HELP_SAVE_HEAD+4
MOVL #1, R0
RET

: 0619
: 0624
: 0626
: 0627
: 0629

; Routine Size: 25 bytes, Routine Base: \$CODE\$ + 003A

```
309 0630 2 ! We begin by parsing the help parameters. If we get an error, just
310 0631 2 ! tell the user and quit.
311 0632 2
312 0633 2 ch$move(8,,parameters,tparse_block[tpa$l_stringcnt]);
313 0634 2 ch$fill(0,8,topic_dsc);
314 0635 2 ch$fill(0,8,subtopic_dsc);
315 0636 2 status = lib$tparse(tparse_block,help_state,help_key);
316 0637 2 if .status eglu lib$ syntaxerr then (
317 0638 3   phn$inform(phn$_badhelp);
318 0639 3   return;
319 0640 2 );
320 0641 2 check (.status);
321 0642 2
322 0643 2 ! Now we initialize the librarian control and open the help library.
323 0644 2
324 0645 2 status = lbr$ini_control(library_index,%ref(lbr$_read),%ref(lbr$_typ_hlp),library_nam);
325 0646 2 check (.status);
326 0647 2 status = lbr$open(library_index,describe('PHONEHELP'),0,describe('SYS$HELP:.HLB'));
327 0648 3 if not .status then (
328 0649 3   phn$inform(phn$_helpmissing);
329 0650 3   return;
330 0651 2 );
331 0652 2
332 0653 2 ! Now we read in the help information requested by the user. For each
333 0654 2 ! line, help_save will be called to save it in memory.
334 0655 2
335 0656 2 status = lbr$get_help(library_index,0,help_save,0,topic_dsc,subtopic_dsc);
336 0657 2 check (.status);
337 0658 2 status = lbr$close(library_index);
338 0659 2 check (.status);
339 0660 2
340 0661 2 ! Now tell the user how to cancel the help information.
341 0662 2
342 0663 2 phn$inform(phn$_helpcan);
343 0664 2
344 0665 2 ! Set the "scrolling in progress" and "must prepare for scrolling" flags
345 0666 2 ! so the steering message routine will know what's happening.
346 0667 2
347 0668 2 phn$gv_scroller = phn$gv_scrollprep = true;
348 0669 2
349 0670 2 ! Finally, queue a steering message to display the first help line.
350 0671 2
351 0672 2 phn$queue_smb(smb__help2);
352 0673 2 return;
353 0674 2
354 0675 1 end;
```

```
                                .PSECT $PLITS,NOWRT,NOEXE,2
                                50 4C 45 48 45 4E 4F 48 50 00000 P.AAB: .ASCII \PHONEHELP\
                                00009
                                00000009 0000C P.AAA: .BLKB 3
                                00000000 00010 .LONG 9
                                42 4C 48 2E 3A 50 4C 45 48 24 53 59 53 00014 P.AAD: .ADDRESS P.AAB
                                00021 .ASCII \SYS$HELP:.HLB\
                                .BLKB 3
```


MISCCMDS
V04-000

MISCCMDS - Miscellaneous Phone Commands
PHN\$HELP_CMD - Handle HELP Command

J 7
16-Sep-1984 02:13:32
14-Sep-1984 12:53:27

VAX-11 Bliss-32 V4.0-742
[PHONE.SRC]MISCCMDS.B32;1

Page 16
(8)

05	52	E8	000B7	BLBS	STATUS, 6\$	: 0657
	52	DD	000BA	PUSHL	STATUS	:
67	01	FB	000BC	CALLS	#1, LIB\$SIGNAL	:
	56	DD	000BF	PJSHL	R6	: 0658
00000000G 00	01	FB	000C1	CALLS	#1, LBR\$CLOSE	:
52	50	DD	000C8	MOVL	R0, STATUS	:
05	52	E8	000CB	BLBS	STATUS, 7\$	: 0659
	52	DD	000CE	PUSHL	STATUS	:
67	01	FB	000D0	CALLS	#1, LIB\$SIGNAL	:
	8F	DD	000D3	PUSHL	#PHN\$ HELPCAN	: 0663
0000G CF	01	FB	000D9	CALLS	#1, PHN\$INFORM	:
0000G CF	06	88	000DE	BISB2	#6, PHN\$GB_FLAGS	: 0668
	05	DD	000E3	PUSHL	#5	: 0672
0000G CF	01	FB	000E5	CALLS	#1, PHN\$QUEUE_SMB	:
	04	000EA	RET			: 0675

; Routine Size: 235 bytes, Routine Base: \$CODE\$ + 0053


```
: 356      0676 1 %sbttl 'PHN$HELP2 - Display a Help Line'
: 357      0677 1 ++
: 358      0678 1 | Functional Description:
: 359      0679 1 |     This steering message routine is called to display the next
: 360      0680 1 |     help line from the help queue. We do each line with a steering
: 361      0681 1 |     message so that other things can happen in between each line.
: 362      0682 1 |
: 363      0683 1 | Formal Parameters:
: 364      0684 1 |     none
: 365      0685 1 |
: 366      0686 1 | Implicit Inputs:
: 367      0687 1 |     global data
: 368      0688 1 |
: 369      0689 1 | Implicit Outputs:
: 370      0690 1 |     global data
: 371      0691 1 |
: 372      0692 1 | Returned Value:
: 373      0693 1 |     none
: 374      0694 1 |
: 375      0695 1 | Side Effects:
: 376      0696 1 |
: 377      0697 1 | --
: 378      0698 1 |
: 379      0699 1 |
: 380      0700 2 global routine phn$help2: novalue = begin
: 381      0701 2
: 382      0702 2 local
: 383      0703 2     c: ref ctl;
: 384      0704 2
: 385      0705 2
: 386      0706 2 ! First we check to see if the help command has been cancelled. If so
: 387      0707 2 ! we remove all remaining help lines from the help queue and throw
: 388      0708 2 ! them away.
: 389      0709 2
: 390      0710 3 if not .phn$gv_scroller then (
: 391      0711 3     while not remque(.help_save_head[0],c) do
: 392      0712 3         phn$kill_ctl(.c);
: 393      0713 3     return;
: 394      0714 2 );
: 395      0715 2
: 396      0716 2 ! If this is the first help line displayed, we need to prepare the
: 397      0717 2 ! screen by clearing most of it.
: 398      0718 2
: 399      0719 3 if .phn$gv_scrollprep then (
: 400      0720 3     phn$scroll_prep();
: 401      0721 3     phn$gv_scrollprep = false;
: 402      0722 2 );
: 403      0723 2
: 404      0724 2 ! Now we remove the next help line from the queue and display it on the
: 405      0725 2 ! screen. If there are more lines, we need to queue another steering message.
: 406      0726 2
: 407      0727 2 if remque(.help_save_head[0],c) eqlu 0 then
: 408      0728 2     phn$queue_smb(smb_help2);
: 409      0729 2 phn$scroll_line(c[ctl_q_line]);
: 410      0730 2 phn$kill_ctl(.c);
: 411      0731 2
: 412      0732 2 return;
```

MISCCMDS
V04-000

MISCCMDS - Miscellaneous Phone Commands
PHN\$HELP2 - Display a Help Line

L 7
16-Sep-1984 02:13:32
14-Sep-1984 12:53:27

VAX-11 Bliss-32 V4.0-742
[PHONE.SRC]MISCCMDS.B32;1

Page 18
(9)

: 413
: 414

0733 2
0734 1 end;

				000C 00000	.ENTRY PHN\$HELP2, Save R2,R3		
		53	0000G	CF 9E 00002	MOVAB PHN\$GB_FLAGS, R3		: 0700
10		63		01 E0 00007	BBS #1, PHN\$GB_FLAGS, 2\$		: 0710
		52	0000'	DF 0F 0000B 1\$:	REMQUE @HELP_SAVE_HEAD, C		: 0711
				39 1D 00010	BVS 5\$		
				52 DD 00012	PUSHL C		: 0712
	0000G	CF		01 FB 00014	CALLS #1, PHN\$KILL_CTL		
				F0 11 00019	BRB 1\$		
08		63		02 E1 0001B 2\$:	BBC #2, PHN\$GB_FLAGS, 3\$		: 0719
	0000G	CF		00 FB 0001F	CALLS #0, PHN\$SCROLL_PREP		: 0720
		63		04 8A 00024	BICB2 #4, PHN\$GB_FLAGS		: 0721
		52	0000'	DF 0F 00027 3\$:	REMQUE @HELP_SAVE_HEAD, C		: 0727
				50 DC 0002C	MOVPSL R0		
50	50	02		01 EF 0002E	EXTZV #1, #2, R0, R0		
				07 12 00033	BNEQ 4\$		
				05 DD 00035	PUSHL #5		: 0728
	0000G	CF		01 FB 00037	CALLS #1, PHN\$QUEUE_SMB		
			10	A2 9F 0003C 4\$:	PUSHAB 16(C)		: 0729
	0000G	CF		01 FB 0003F	CALLS #1, PHN\$SCROLL_LINE		
				52 DD 00044	PUSHL C		: 0730
	0000G	CF		01 FB 00046	CALLS #1, PHN\$KILL_CTL		
				04 0004B 5\$:	RET		: 0734

; Routine Size: 76 bytes, Routine Base: \$CODE\$ + 013E


```
: 416 0735 1 %sbttl 'PHN$DIRECTORY_CMD - Provide a List of Users'
: 417 0736 1 ++
: 418 0737 1 Functional Description:
: 419 0738 1 This routine handles the DIRECTORY command, which provides a list of
: 420 0739 1 users that we could call, not unlike the phone book.
: 421 0740 1
: 422 0741 1 Formal Parameters:
: 423 0742 1 node Address of descriptor of the node we want to list.
: 424 0743 1
: 425 0744 1 Implicit Inputs:
: 426 0745 1 global data
: 427 0746 1
: 428 0747 1 Implicit Outputs:
: 429 0748 1 global data
: 430 0749 1
: 431 0750 1 Returned Value:
: 432 0751 1 none
: 433 0752 1
: 434 0753 1 Side Effects:
: 435 0754 1
: 436 0755 1 --
: 437 0756 1
: 438 0757 1
: 439 0758 2 global routine phn$directory_cmd(node): novalue = begin
: 440 0759 2
: 441 0760 2 bind
: 442 0761 2 node_dsc = .node: descriptor;
: 443 0762 2
: 444 0763 2 local
: 445 0764 2 status: long,
: 446 0765 2 node_tsb: tsb;
: 447 0766 2 local
: 448 0767 2 local_described_buffer(prepared_node,nam$cm_maxrss);
: 449 0768 2
: 450 0769 2
: 451 0770 2 ! We begin by preparing the node name specified by the user. This gets
: 452 0771 2 ! it ready for use in generating the directory.
: 453 0772 2
: 454 0773 2 phn$prepare_users_target(node_dsc,true,prepared_node);
: 455 0774 2
: 456 0775 2 ! Now we make a TSB in order to parse the node name. If we get a syntax
: 457 0776 2 ! error, or if the user included a user name with the node name, we
: 458 0777 2 ! tell the user and quit.
: 459 0778 2
: 460 0779 2 status = phn$make_tsb(prepared_node,node_tsb);
: 461 0780 3 if .status nequ phn$ok then (
: 462 0781 3 phn$inform(.status);
: 463 0782 3 return;
: 464 0783 2 );
: 465 0784 3 if .node_tsb[tsb_v_user] then (
: 466 0785 3 phn$inform(phn$onlynode);
: 467 0786 3 return;
: 468 0787 2 );
: 469 0788 2
: 470 0789 2 ! Now we have to set up so that we can display the first directory line.
: 471 0790 2 ! This is dependent upon whether it's a local or remote directory.
: 472 0791 2
```

```

: 473 0792 3 begin
: 474 0793 3 local
: 475 0794 3     local_described_buffer(pub_address,4);
: 476 0795 3
: 477 0796 3 if not .node_tsb[tsb_v_remote] then
: 478 0797 3
: 479 0798 3     ! It's a local directory. Queue a steering message to do the first
: 480 0799 3     ! line. We pass a pub address of zero, meaning "local directory".
: 481 0800 3
: 482 0801 3     .pub_address[ptr] = 0
: 483 0802 4 else (
: 484 0803 4
: 485 0804 4     ! It's a remote directory. Queue a steering message to do the first
: 486 0805 4     ! line, containing the address of a temporary PUB describing the node.
: 487 0806 4
: 488 0807 4     status = phn$estab_link(prepared_node,.pub_address[ptr]);
: 489 0808 4     if .status nequ phn$_ok then
: 490 0809 4         return;
: 491 0810 3 );
: 492 0811 3 phn$queue_smb(smb__directory2,pub_address);
: 493 0812 2 end;
: 494 0813 2
: 495 0814 2
: 496 0815 2
: 497 0816 2 ! Ok, we're pretty much done. We need to tell the user how to cancel
: 498 0817 2 ! the directory. We also need to set the "scrolling in progress" and
: 499 0818 2 ! "must prepare scrolling" flags so the steering routine will know what's up.
: 500 0819 2
: 501 0820 2 phn$inform(phn$_dircan);
: 502 0821 2
: 503 0822 2 phn$gv_scroller = phn$gv_scrollprep = true;
: 504 0823 2
: 505 0824 2 return;
: 506 0825 2
: 507 0826 1 end;
```

```

                                000C 00000
                                8F DO 00002
                                53 00000000G 8F DO 00002
                                5E FE08 CE 9E 00009
                                OC AE FF 8F 9A 0000E
                                10 AE 14 AE 9E 00013
                                OC AE 9F 00018
                                04 AC DD 0001D
                                0000G CF 03 FB 00020
                                FF1C CD 9F 00025
                                10 AE 9F 00029
                                0000G CF 02 FB 0002C
                                52 50 DO 00031
                                53 52 D1 00034
                                04 13 00037
                                52 DD 00039
                                OC 11 0003B
```

```

.ENTRY PHN$DIRECTORY_CMD, Save R2,R3
MOVL #PHN$_OK, R3
MOVAB -504(SP), SP
MOVZBL #255, PREPARED_NODE
MOVAB PREPARED_NODE+8, PREPARED_NODE+4
PUSHAB PREPARED_NODE
PUSHL #1
PUSHL NODE
CALLS #3, PHN$PREPARE_USERS_TARGET
PUSHAB NODE_TSB
PUSHAB PREPARED_NODE
CALLS #2, PHN$MAKE_TSB
MOVL R0, STATUS
CML STATUS, R3
BEQL 1$
PUSHL STATUS
BRB 2$
```

```

: 0758
:
:
: 0767
:
: 0773
:
:
: 0779
:
:
: 0780
:
: 0781
:
```


MISCCMDS
V04-000

MISCCMDS - Miscellaneous Phone Commands
PHN\$DIRECTORY_CMD - Provide a List of Users

B 8
16-Sep-1984 02:13:32
14-Sep-1984 12:53:27

VAX-11 Bliss-32 V4.0-742
[PHONE.SRC]MISCCMDS.B32;1

Page 21
(10)

OC	FF1C	CD	00000000G	01	E1	0003D	1\$:	BBC	#1, NODE_TSB, 3\$	:	0784
				8F	DD	00043		PUSHL	#PHN\$ ONLYNODE	:	0785
	0000G	CF		01	FB	00049	2\$:	CALLS	#1, PHN\$INFORM	:	
					04	0004E		RET		:	0784
		6E		04	D0	0004F	3\$:	MOVL	#4, PUB_ADDRESS	:	0794
	04	AE	08	AE	9E	00052		MOVAB	PUB_ADDRESS+8, PUB_ADDRESS+4	:	
		05	FF1C	CD	E8	00057		BLBS	NODE_TSB, 4\$	:	0796
			04	BE	D4	0005C		CLRL	@PUB_ADDRESS+4	:	0801
				13	11	0005F		BRB	5\$	:	
			04	AE	DD	00061	4\$:	PUSHL	PUB_ADDRESS+4	:	0807
			10	AE	9F	00064		PUSHAB	PREPARED_NODE	:	
	0000G	CF		02	FB	00067		CALLS	#2, PHN\$ESTAB_LINK	:	
		52		50	D0	0006C		MOVL	R0, STATUS	:	
		53		52	D1	0006F		CMPL	STATUS, R3	:	0808
				19	12	00072		BNEQ	6\$	:	
				5E	DD	00074	5\$:	PUSHL	SP	:	0811
				0F	DD	00076		PUSHL	#15	:	
	0000G	CF		02	FB	00078		CALLS	#2, PHN\$QUEUE_SMB	:	
			00000000G	8F	DD	0007D		PUSHL	#PHN\$ DIRCAN	:	0820
	0000G	CF		01	FB	00083		CALLS	#1, PHN\$INFORM	:	
	0000G	CF		06	88	00088		BISB2	#6, PHN\$GB_FLAGS	:	0822
				04	0008D	6\$:		RET		:	0826

; Routine Size: 142 bytes, Routine Base: \$CODE\$ + 018A


```
509 0827 1 %sbttl 'PHN$DIRECTORY2 - Display Directory Line'
510 0828 1 ++
511 0829 1 Functional Description:
512 0830 1 This steering message routine is called to display the next line
513 0831 1 of directory information. This is done one at a time with steering
514 0832 1 messages so that other things can happen in between.
515 0833 1
516 0834 1 Formal Parameters:
517 0835 1 pub_address If it's a local directory, this is zero. If it's
518 0836 1 remote, it's the address of a PUB describing the node.
519 0837 1
520 0838 1 Implicit Inputs:
521 0839 1 global data
522 0840 1
523 0841 1 Implicit Outputs:
524 0842 1 global data
525 0843 1
526 0844 1 Returned Value:
527 0845 1 none
528 0846 1
529 0847 1 Side Effects:
530 0848 1
531 0849 1 --
532 0850 1
533 0851 1
534 0852 2 global routine phn$directory2(pub_address): novalue = begin
535 0853 2
536 0854 2 bind
537 0855 2 pub_address_dsc = .pub_address: descriptor,
538 0856 2 np = .pub_address_dsc[ptr]: ref pub;
539 0857 2
540 0858 2 own
541 0859 2 wild_pid: long,
542 0860 2 person_count: long;
543 0861 2
544 0862 2 local
545 0863 2 status: long,
546 0864 2 decnet_iosb: block[8,byte];
547 0865 2 local
548 0866 2 local_described_buffer(display_line,79);
549 0867 2
550 0868 2
551 0869 2 ! First we check to see if the directory has been cancelled. If so, quit.
552 0870 2 ! If it's a remote directory, though, make sure we break the link.
553 0871 2
554 0872 3 if not .phn$gv_scroller then (
555 0873 3 if .np neqa 0 then
556 0874 3 phn$break_link(.np,smb__slave_done);
557 0875 3 return;
558 0876 3 );
559 0877 2
560 0878 2 ! If this is the first line of the directory, we need to prepare the screen
561 0879 2 ! by clearing most of it and displaying some headings. Zero person count.
562 0880 2
563 0881 3 if .phn$gv_scrollprep then (
564 P 0882 3 phn$scroll_prep(describe(
565 0883 3 'Process Name User Name Terminal Phone Status'));
```



```

: 566      0884      3      wild_pid = -1;
: 567      0885      3      person_count = 0;
: 568      0886      3      phn$gv_scrollprep = false;
: 569      0887      2      );
: 570      0888      2
: 571      0889      2      ! Now we split up depending on whether it's a local or remote directory.
: 572      0890      2
: 573      0891      3      if .np eqa 0 then (
: 574      0892      3
: 575      0893      3          ! It's a local directory. Get the next directory line, if it exists.
: 576      0894      3
: 577      0895      3          status = phn$directory_line(wild_pid,display_line);
: 578      0896      3          status = .status eqa phn$ok;
: 579      0897      3      ) else (
: 580      0898      3
: 581      0899      3          ! It's a remote directory. Send a steering message to the network
: 582      0900      3          ! slave asking it for a line. Get the line. If its length is zero,
: 583      0901      3          ! we're done with the directory.
: 584      0902      3
: 585      0903      3          phn$send_smb(.np,smb__directory2);
: 586      0904      3          status = phn$read_slave(.np[pub_w_channel],display_line,false);
: 587      0905      3          status = .status eqa phn$ok and .display_line[len] gtru 0;
: 588      0906      3          if not .status then
: 589      0907      3              phn$break_link(.np,smb__slave_done);
: 590      0908      2      );
: 591      0909      2
: 592      0910      2      ! Let's see what happened with that directory line.
: 593      0911      2
: 594      0912      3      if .status then (
: 595      0913      3
: 596      0914      3          ! We got one. Scroll it onto the screen and queue a steering message
: 597      0915      3          ! so we'll be recalled to do the next line.
: 598      0916      3
: 599      0917      3          phn$scroll_line(display_line);
: 600      0918      3          inc (person_count);
: 601      0919      3          phn$queue_smb(smb__directory2,pub_address_dsc);
: 602      0920      3
: 603      0921      3      ) else (
: 604      0922      3
: 605      0923      3          ! There wasn't another line. Tell the user how many people were
: 606      0924      3          ! listed and quit.
: 607      0925      3
: 608      0926      3          phn$scroll_line(describe(''));
: 609      0927      3          display_line[len] = 79;
: 610      0928      3          status = $fao(describe('!UL person!%S listed.'),
: 611      0929      3              display_line,
: 612      0930      3              display_line,
: 613      0931      3              .person_count);
: 614      0932      3          check (.status);
: 615      0933      3          phn$scroll_line(display_line);
: 616      0934      2      );
: 617      0935      2
: 618      0936      2      return;
: 619      0937      2
: 620      0938      1      end;
```



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

20 20 20 65 6D 61 4E 20 73 73 65 63 6F 72 50 0002C P.AAF: .ASCII \Process Name      User Name      Terminal\
20 20 20 20 20 65 6D 61 4E 20 72 65 73 55 20 0003B
53 20 65 6E 6F 68 50 20 20 20 20 20 20 20 20 0004A
                                     .ASCII \          Phone Status\
                                     00054
                                     00063
                                     0000003C 00068 P.AAE: .LONG 60
                                     00000000 0006C .ADDRESS P.AAF
                                     00070 P.AAH: .BLKB 0
                                     00070 P.AAG: .LONG 0
                                     00000000 00074 .ADDRESS P.AAH
6C 20 53 25 21 6E 6F 73 72 65 70 20 4C 55 21 00078 P.AAJ: .ASCII \!UL person!%S listed.\
2E 64 65 74 73 69 00087
                                     0008D
                                     00000015 00090 P.AAI: .BLKB 3
                                     00000000 00094 .LONG 21
                                     .ADDRESS P.AAJ

.PSECT $OWN$,NOEXE,2

000A0 WILD_PID:
                                     .BLKB 4
000A4 PERSON_COUNT:
                                     .BLKB 4

.EXTRN SYS$FAO

.PSECT $CODE$,NOWRT,2

01FC 00000 .ENTRY PHN$DIRECTORY2, Save R2,R3,R4,R5,R6,R7,R8 : 0852
58 0000G CF 9E 00002 MOVAB PHN$GB_FLAGS, R8
57 0000G CF 9E 00007 MOVAB PHN$SCROLL_LINE, R7
56 00000000G 8F D0 0000C MOVL #PHN$ OK, R6
55 0000' CF 9E 00013 MOVAB PERSON_COUNT, R5
5E A4 AE 9E 00018 MOVAB -92(SP), SP
53 04 AC D0 0001C MOVL PUB_ADDRESS, R3 : 0855
7E 4F 8F 9A 00020 MOVZBL #79, DISPLAY_LINE : 0866
04 AE 08 AE 9E 00024 MOVAB DISPLAY_LINE+8, DISPLAY_LINE+4
11 68 01 E0 00029 BBS #1, PHN$GB_FLAGS, 2$ : 0872
04 B3 D5 0002D TSTL @4(R3) : 0873
01 12 00030 BNEQ 1$
04 00032 RET
0D DD 00033 1$: PUSHL #13 : 0874
04 B3 DD 00035 PUSHL @4(R3)
0000G CF 02 FB 00038 CALLS #2, PHN$BREAK_LINK : 0872
04 0003D RET : 0881
12 68 02 E1 0003E 2$: BBC #2, PHN$GB_FLAGS, 3$ : 0883
0000G CF 01 FB 00042 PUSHAB P.AAE
FC A5 01 CE 0004B CALLS #1, PHN$SCROLL_PREP : 0884
65 D4 0004F MNEGL #1, WILD_PID : 0885
04 0A 00051 CLRL PERSON_COUNT : 0886
52 04 B3 D0 00054 3$: MOVL @4(R3), R2 : 0891
1B 12 00058 BNEQ 5$
5E DD 0005A PUSHL SP : 0895
FC A5 9F 0005C PUSHAB WILD_PID
```


0000V	CF	02	FB	0005F	CALLS	#2, PHN\$DIRECTORY_LINE	:
	54	50	D0	00064	MOVL	R0, STATUS	:
		50	D4	00067	CLRL	R0	: 0896
	56	54	D1	00069	CMPL	STATUS, R6	:
		02	12	0006C	BNEQ	4\$	:
		50	D6	0006E	INCL	R0	:
	54	50	D0	00070	MOVL	R0, STATUS	:
		3F	11	00073	BRB	8\$	: 0891
		0F	DD	00075	PUSHL	#15	: 0903
		52	DD	00077	PUSHL	R2	:
0000G	CF	02	FB	00079	CALLS	#2, PHN\$SEND_SMB	:
		7E	D4	0007E	CLRL	-(SP)	: 0904
		AE	9F	00080	PUSHAB	DISPLAY_LINE	:
	7E	C2	3C	00083	MOVZWL	244(R2), -(SP)	:
0000G	CF	03	FB	00088	CALLS	#3, PHN\$READ_SLAVE	:
	54	50	D0	0008D	MOVL	R0, STATUS	:
		51	D4	00090	CLRL	R1	: 0905
	56	54	D1	00092	CMPL	STATUS, R6	:
		02	12	00095	BNEQ	6\$	:
		51	D6	00097	INCL	R1	:
		50	D4	00099	CLRL	R0	:
		6E	B5	0009B	TSTW	DISPLAY_LINE	:
		02	13	0009D	BEQL	7\$	:
		50	D6	0009F	INCL	R0	:
	54	51	D2	000A1	MCOML	R1, STATUS	:
54	50	54	CB	000A4	BICL3	STATUS, R0, STATUS	:
	0C	54	E8	000A8	BLBS	STATUS, 9\$	: 0906
		0D	DD	000AB	PUSHL	#13	: 0907
		52	DD	000AD	PUSHL	R2	:
0000G	CF	02	FB	000AF	CALLS	#2, PHN\$BREAK_LINK	:
	11	54	E9	000B4	BLBC	STATUS, 10\$	: 0912
		5E	DD	000B7	PUSHL	SP	: 0917
	67	01	FB	000B9	CALLS	#1, PHN\$SCROLL_LINE	:
		65	D6	000BC	INCL	PERSON_COUNT	: 0918
		53	DD	000BE	PUSHL	R3	: 0919
		0F	DD	000C0	PUSHL	#15	:
0000G	CF	02	FB	000C2	CALLS	#2, PHN\$QUEUE_SMB	:
		04	000C7	RET			: 0912
		CF	9F	000C8	PUSHAB	P.AAG	: 0926
	67	01	FB	000CC	CALLS	#1, PHN\$SCROLL_LINE	:
	6E	8F	9B	000CF	MOVZBW	#79, DISPLAY_LINE	: 0927
		65	DD	000D3	PUSHL	PERSON_COUNT	: 0931
		AE	9F	000D5	PUSHAB	DISPLAY_LINE	:
		AE	9F	000D8	PUSHAB	DISPLAY_LINE	:
		CF	9F	000DB	PUSHAB	P.AAI	:
00000000G	00	04	FB	000DF	CALLS	#4, SYSSFA0	:
	54	50	D0	000E6	MOVL	R0, STATUS	:
	09	54	E8	000E9	BLBS	STATUS, 11\$	: 0932
		54	DD	000EC	PUSHL	STATUS	:
00000000G	00	01	FB	000EE	CALLS	#1, LIB\$SIGNAL	:
		5E	DD	000F5	PUSHL	SP	: 0933
	67	01	FB	000F7	CALLS	#1, PHN\$SCROLL_LINE	:
		04	000FA	RET			: 0938

; Routine Size: 251 bytes, Routine Base: \$CODE\$ + 0218

```

: 622 0939 1 %sbttl 'PHN$DIRECTORY_LINE - Build a Directory Line'
: 623 0940 1 ++
: 624 0941 1 Functional Description:
: 625 0942 1 This routine is called to build a directory line for displaying.
: 626 0943 1 It uses the wild PID to obtain the next available process, and
: 627 0944 1 then obtains information from the process for the line.
: 628 0945 1
: 629 0946 1 Formal Parameters:
: 630 0947 1 wild_pid Address of a longword wild PID. Must be initialized
: 631 0948 1 to -1 at the beginning.
: 632 0949 1 line_buf Address of a descriptor of a display line buffer.
: 633 0950 1 We fill in the length.
: 634 0951 1
: 635 0952 1 Implicit Inputs:
: 636 0953 1 global data
: 637 0954 1
: 638 0955 1 Implicit Outputs:
: 639 0956 1 global data
: 640 0957 1
: 641 0958 1 Returned Value:
: 642 0959 1 phn$_noprocs No more processes to display.
: 643 0960 1
: 644 0961 1 Side Effects:
: 645 0962 1
: 646 0963 1 --
: 647 0964 1
: 648 0965 1
: 649 0966 2 global routine phn$directory_line(wild_pid,line_buf) = begin
: 650 0967 2
: 651 0968 2 own
: 652 0969 2 own_described_buffer(process_name,16),
: 653 0970 2 own_described_buffer(user_name,12),
: 654 0971 2 own_described_buffer(terminal_number,7),
: 655 0972 2 parent_pid: long;
: 656 0973 2
: 657 0974 2 bind
: 658 0975 2 get_info = uplit(word(16),word(jpi$prcnam),
: 659 0976 2 long(process_name+8),
: 660 0977 2 long(process_name),
: 661 0978 2 word(12),word(jpi$_username),
: 662 0979 2 long(user_name+8),
: 663 0980 2 long(user_name),
: 664 0981 2 word(7),word(jpi$terminal),
: 665 0982 2 long(terminal_number+8),
: 666 0983 2 long(terminal_number),
: 667 0984 2 word(4),word(jpi$_owner),
: 668 0985 2 long(parent_pid),
: 669 0986 2 long(0),
: 670 0987 2 long(0));
: 671 0988 2
: 672 0989 2 local
: 673 0990 2 status: long,
: 674 0991 2 terminal_ptr: ref descriptor,
: 675 0992 2 phone_status_ptr: ref descriptor;
: 676 0993 2
: 677 0994 2
: 678 0995 2 ! We obtain information about the next process using the wildcard $GETJPI
```



```
: 679      0996 2 ! facility. If we get a bad status, or if the process does not own a terminal
: 680      0997 2 ! (isn't interactive), then just try the next one. Also, we don't display
: 681      0998 2 ! subprocesses.
: 682      0999 2
: 683      1000 2 do (
: 684      P 1001 2     status = $getjpi(efn=phn$k_getjpiefn,
: 685      P 1002 2         pidadr=.wild_pid,
: 686      1003 2         itmlst=get_info);
: 687      1004 2     if .status eqlu ss$_nomoreproc then
: 688      1005 2         return phn$_noprocs;
: 689      1006 2
: 690      1007 2 ) until (.status eqlu ss$_normal) and (.terminal_number[len] nequ 0) and (.parent_pid eqlu 0);
: 691      1008 2
: 692      1009 2 $waitfr(efn=phn$k_getjpiefn);
: 693      1010 2
: 694      1011 2 ! OK, now we split up depending upon whether the process' terminal can
: 695      1012 2 ! be used as a telephone.
: 696      1013 2
: 697      1014 2 if phn$term_characteristic(terminal_number,tt$m_scope) then (
: 698      1015 2
: 699      1016 2     ! This process has a nice terminal. We will display its number.
: 700      1017 2
: 701      1018 2     terminal_ptr = terminal_number;
: 702      1019 2
: 703      1020 2     ! The phone status is determined by whether or not the person
: 704      1021 2     ! has set /NOBROADCAST to unplug the phone.
: 705      1022 2
: 706      1023 2     phone_status_ptr = (if phn$term_characteristic(terminal_number,tt$m_nobrdcst) then
: 707      1024 2         describe('/nobroadcast')
: 708      1025 2     else
: 709      1026 2         describe('available'));
: 710      1027 2
: 711      1028 2 ) else (
: 712      1029 2
: 713      1030 2     ! As it turns out, the guy's terminal is useless. Set up to
: 714      1031 2     ! display that.
: 715      1032 2
: 716      1033 2     terminal_ptr = describe('unusable');
: 717      1034 2     phone_status_ptr = describe('---');
: 718      1035 2 );
: 719      1036 2
: 720      1037 2 ! Now we use $FA0 to build the directory line to be displayed. It contains
: 721      1038 2 ! the process name, user name, terminal info, and phone status info.
: 722      1039 2 ! Display the line.
: 723      1040 2
: 724      P 1041 2 status = $fao(describe('!16<!AF!>!16<!AS!>!16<!AS!>!AS'),
: 725      P 1042 2     .line_buf,
: 726      P 1043 2     .line_buf,
: 727      P 1044 2     .process_name[len],
: 728      P 1045 2     .process_name[ptr],
: 729      P 1046 2     user_name,
: 730      P 1047 2     .terminal_ptr,
: 731      1048 2     phone_status_ptr);
: 732      1049 2 check (.status);
: 733      1050 2
: 734      1051 2 return phn$_ok;
: 735      1052 2
```

```

.PSECT $PLITS$,NOWRT,NOEXE,2

0010 00098 P.AAK: .WORD 16
031C 0009A .WORD 796
00000000' 0009C .ADDRESS PROCESS_NAME+8
00000000' 000A0 .ADDRESS PROCESS_NAME
000C 000A4 .WORD 12
0202 000A6 .WORD 514
00000000' 000A8 .ADDRESS USER_NAME+8
00000000' 000AC .ADDRESS USER_NAME
0007 000B0 .WORD 7
031D 000B2 .WORD 797
00000000' 000B4 .ADDRESS TERMINAL_NUMBER+8
00000000' 000B8 .ADDRESS TERMINAL_NUMBER
0004 000BC .WORD 4
0303 000BE .WORD 771
00000000' 000C0 .ADDRESS PARENT_PID
00000000' 000C4 .LONG 0
00000000' 000C8 .LONG 0
74 73 61 63 64 61 6F 72 62 6F 6E 2F 000CC P.AAM: .ASCII \nobroadcast\
00000000' 000D8 P.AAL: .LONG 12
00000000' 000DC .ADDRESS P.AAM
65 6C 62 61 6C 69 61 76 61 000E0 P.AAO: .ASCII \available\
000E9 .BLKB 3
00000009 000EC P.AAN: .LONG 9
00000000' 000F0 .ADDRESS P.AAO
65 6C 62 61 73 75 6E 75 000F4 P.AAQ: .ASCII \unusable\
00000008 000FC P.AAP: .LONG 8
00000000' 00100 .ADDRESS P.AAQ
2D 2D 2D 00104 P.AAS: .ASCII \---\
00107 .BLKB 1
00000003 00108 P.AAR: .LONG 3
00000000' 0010C .ADDRESS P.AAS
41 21 3C 36 31 21 3E 21 46 41 21 3C 36 31 21 00110 P.AAU: .ASCII \!16<!AF!>!16<!AS!>!16<!AS!>!AS\
53 41 21 3E 21 53 41 21 3C 36 31 21 3E 21 53 0011F .BLKB 2
0000001E 0012E P.AAT: .LONG 30
00000000' 00134 .ADDRESS P.AAU

.PSECT $OWNS$,NOEXE,2

00000010 000A8 PROCESS_NAME:
00000000' 000AC .LONG 16
00000000' 000B0 .ADDRESS PROCESS_NAME+8
0000000C 000C0 USER_NAME:
00000000' 000C4 .LONG 12
00000000' 000C8 .ADDRESS USER_NAME+8
00000007 000D4 TERMINAL_NUMBER:
00000000' 000D8 .LONG 7
00000000' 000DC .ADDRESS TERMINAL_NUMBER+8
000DC .BLKB 7

```



```
000E3 .BLKB 1
000E4 PARENT_PID: .BLKB 4

GET_INFO= P.AAK
          .EXTRN SYSS$GETJPI, SYSS$WAITFR
          .PSECT $CODE$,NOWRT,2

          .ENTRY PHN$DIRECTORY_LINE, Save R2,R3,R4,R5
55      0000' CF 9E 00002 MOVAB GET_INFO, R5
54      0000' CF 9E 00007 MOVAB TERMINAL_NUMBER, R4
          7E 7C 0000C 1$: CLRL -(SP)
          7E D4 0000E CLRL -(SP)
          55 DD 00010 PUSHL R5
          7E D4 00012 CLRL -(SP)
          04 AC DD 00014 PUSHL WILD_PID
          01 DD 00017 PUSHL #1
00000000G 00 07 FB 00019 CALLS #7, SYSS$GETJPI
          53 50 D0 00020 MOVL R0, STATUS
000009A8 8F 53 D1 00023 CMPL STATUS, #2472
          08 12 0002A BNEQ 2$
          50 00000000G 8F D0 0002C MOVL #PHN$_NOPROC, R0
          04 00033 RET
          01 53 D1 00034 2$: CMPL STATUS, #1
          D3 12 00037 BNEQ 1$
          64 B5 00039 TSTW TERMINAL_NUMBER
          CF 13 0003B BEQL 1$
          10 A4 D5 0003D TSTL PARENT_PID
          CA 12 00040 BNEQ 1$
          01 DD 00042 PUSHL #1
00000000G 00 01 FB 00044 CALLS #1, SYSS$WAITFR
          7E 1000 8F 3C 0004B MOVZWL #4096, -(SP)
          54 DD 00050 PUSHL R4
          0000G CF 02 FB 00052 CALLS #2, PHN$TERM_CHARACTERISTIC
          1F 50 E9 00057 BLBC R0, 4$
          52 00020000 64 9E 0005A MOVAB TERMINAL_NUMBER, TERMINAL_PTR
          8F DD 0005D PUSHL #131072
          54 DD 00063 PUSHL R4
          0000G CF 02 FB 00065 CALLS #2, PHN$TERM_CHARACTERISTIC
          06 50 E9 0006A BLBC R0, 3$
          50 40 A5 9E 0006D MOVAB P.AAL, PHONE_STATUS_PTR
          0E 11 00071 BRB 5$
          50 54 A5 9E 00073 3$: MOVAB P.AAN, PHONE_STATUS_PTR
          08 11 00077 BRB 5$
          52 64 A5 9E 00079 4$: MOVAB P.AAP, TERMINAL_PTR
          50 70 A5 9E 0007D MOVAB P.AAR, PHONE_STATUS_PTR
          50 DD 00081 5$: PUSHL PHONE_STATUS_PTR
          52 DD 00083 PUSHL TERMINAL_PTR
          EC A4 9F 00085 PUSHAB USER_NAME
          D8 A4 DD 00088 PUSHL PROCESS_NAME+4
          7E D4 A4 3C 0008B MOVZWL PROCESS_NAME, -(SP)
          08 AC DD 0008F PUSHL LINE_BUF
          08 AC DD 00092 PUSHL LINE_BUF
          0098 C5 9F 00095 PUSHAB P.AAT
          00000000G 00 08 FB 00099 CALLS #8, SYSS$FAO
          53 50 D0 000A0 MOVL R0, STATUS
```

1049
1051
1053

```
; Routine Size: 183 bytes,    Routine Base: $CODE$ + 0313
```



```
: 738 1054 1 %sbttl 'PHN$MAIL_CMD - Handle MAIL Command'
: 739 1055 1 ++
: 740 1056 1 Functional Description:
: 741 1057 1 This routine handles the MAIL command, which is used to send
: 742 1058 1 a short phone message to another user. We will spawn a subprocess
: 743 1059 1 to send the message with the MAIL utility.
: 744 1060 1
: 745 1061 1 Formal Parameters:
: 746 1062 1 parameters The rest of the command line as entered by the user.
: 747 1063 1 It is assumed to be in the following form:
: 748 1064 1 addressee "message[]"
: 749 1065 1
: 750 1066 1 Implicit Inputs:
: 751 1067 1 global data
: 752 1068 1
: 753 1069 1 Implicit Outputs:
: 754 1070 1 global data
: 755 1071 1
: 756 1072 1 Returned Value:
: 757 1073 1 none
: 758 1074 1
: 759 1075 1 Side Effects:
: 760 1076 1
: 761 1077 1 --
: 762 1078 1
: 763 1079 1
: 764 1080 2 global routine phn$mail_cmd(parameters): novalue = begin
: 765 1081 2
: 766 1082 2 bind
: 767 1083 2 parameters_dsc = .parameters: descriptor;
: 768 1084 2
: 769 1085 2 local
: 770 1086 2 status_1 : block[4,byte],
: 771 1087 2 status_2 : block[4,byte],
: 772 1088 2 priv_vec : vector[2,long],
: 773 1089 2 char_ptr: long,
: 774 1090 2 message_dsc: descriptor;
: 775 1091 2
: 776 1092 2
: 777 1093 2 ! We begin by rebuilding the parameters descriptor so it describes only
: 778 1094 2 ! the addressee. Then we build a new descriptor, message_dsc, to describe
: 779 1095 2 ! the phone message text.
: 780 1096 2
: 781 1097 2 char_ptr = ch$find_ch(.parameters_dsc[len],.parameters_dsc[ptr],''');
: 782 1098 3 if .char_ptr eql 0 then (
: 783 1099 3 phn$inform(phn$_badmailcmd);
: 784 1100 3 return;
: 785 1101 2 );
: 786 1102 2 message_dsc[len] = .parameters_dsc[len] - (.char_ptr - .parameters_dsc[ptr]);
: 787 1103 2 parameters_dsc[len] = .char_ptr - .parameters_dsc[ptr];
: 788 1104 3 if .parameters_dsc[len] eql 0 then (
: 789 1105 3 phn$inform(phn$_badmailcmd);
: 790 1106 3 return;
: 791 1107 2 );
: 792 1108 2
: 793 1109 2 message_dsc[ptr] = .char_ptr + 1;
: 794 1110 2 char_ptr = ch$find_ch(.message_dsc[len]-1,.message_dsc[ptr],''');
```

```

: 795 1111 3 message_dsc[len] = (if .char_ptr eqla 0 then .message_dsc[len] - 1
: 796 1112 2 else .char_ptr - .message_dsc[ptr]);
: 797 1113 2
: 798 1114 2 ! Put out a prompt so the user knows it's gonna take a while to send
: 799 1115 2 ! this bloody message.
: 800 1116 2
: 801 1117 2 phn$inform(phn$_sendingmail);
: 802 1118 2
: 803 1119 2 ! Now we can create a subprocess to mail the message to the addressee.
: 804 1120 2 ! This is done using LIB$SPAWN.
: 805 1121 2
: 806 1122 3 begin
: 807 1123 3 local
: 808 1124 3     local_described_buffer(command_buf,128);
: 809 1125 3
: 810 1126 3 ch$copy(30,uplit byte('MAIL/SUBJECT='PHONE Message: '),
: 811 1127 3     .message_dsc[len],.message_dsc[ptr],
: 812 1128 3     6,uplit byte('' NL: '),
: 813 1129 3     .parameters_dsc[len],.parameters_dsc[ptr],
: 814 1130 3     .command_buf[len],.command_buf[ptr]);
: 815 1131 3 priv_vec[0] = (1^$bitposition(prv$v_netmbx)) or
: 816 1132 3     (1^$bitposition(prv$v_oper)) or
: 817 1133 3     (1^$bitposition(prv$v_prmmbx)) or
: 818 1134 3     (1^$bitposition(prv$v_sysnam)) or
: 819 1135 3     (1^$bitposition(prv$v_world));
: 820 1136 3 priv_vec[1] = 0;
: 821 1137 3 $setprv(enbflg=0,prvadr=priv_vec);
: 822 1138 3 status_2 = lib$spawn(command_buf,$descriptor('NL:'),0,%ref(%b'010'),0,
: 823 1139 3     0,status_1);
: 824 1140 3 $setprv(enbflg=1,prvadr=priv_vec);
: 825 1141 3 end;
: 826 1142 2
: 827 1143 2 ! If we encountered a problem, tell the user about it. Otherwise just
: 828 1144 2 ! clear away the prompt.
: 829 1145 2
: 830 1146 3 phn$inform((if not .status_2 !Report lib$spawn error
: 831 1147 3     and not .status_2[sts$v_inhib_msg]
: 832 1148 3     then .status_2
: 833 1149 3     else if not .status_1 !or error from mail
: 834 1150 3         and not .status_1[sts$v_inhib_msg] ! unless inhibited
: 835 1151 3         then .status_1
: 836 1152 3         else 0));
: 837 1153 2
: 838 1154 2 return;
: 839 1155 2
: 840 1156 1 end;

```

														.PSECT \$PLIT\$,NOWRT,NOEXE,2					
50	22	3D	54	43	45	4A	42	55	53	2F	4C	49	41	4D	00138	P.AAV:	.ASCII	\MAIL/SUBJECT='PHONE Message: \	:
20	20	3A	65	67	61	73	73	65	4D	20	45	4E	4F	48	00147				:
									20	3A	4C	4E	20	22	00156	P.AAW:	.ASCII	\'' NL: \	:
												3A	4C	4E	0015C	P.AAY:	.ASCII	\NL:\	:
															0015F		.BLKB	1	:
															00000003	00160	P.AAX:	.LONG	3

				00000000' 00164	.ADDRESS P.AAY	:		
					.EXTRN SYS\$SETPRV			
					.PSECT \$CODE\$,NOWRT,2			
				OFFC 00000	.ENTRY PHN\$MAIL_CMD, Save R2,R3,R4,R5,R6,R7,R8,R9,-;	1080		
			5B	00000000G	00 9E 00002	MOVAB	R10, R11	
			5E	FF60	CE 9E 00009	MOVAB	SY\$SETPRV, R11	
			56	04	AC D0 0000E	MOVAB	-160(SP), SP	
04	B6		66		22 3A 00012	MOVL	PARAMETERS, R6	1083
					02 12 00017	LOCC	#34, (R6), @4(R6)	1097
					51 D4 00019	BNEQ	1\$	
			53		51 D0 0001B	CLRL	R1	
					CF 13 0001E	MOVL	R1, CHAR_PTR	
					53 C3 00020	BEQL	2\$	1098
					50 A1 00025	SUBL3	CHAR_PTR, 4(R6), R0	1102
F0	50	04	A6		50 AE 0002A	ADDW3	R0, (R6), MESSAGE_DSC	
	AD		66		09 12 0002D	MNEGW	R0, (R6)	1103
					00000000G	BNEQ	3\$	1104
					00DD 31 00035	PUSHL	#PHN\$_BADMAILCMD	1105
					01 A3 9E 00038	BRW	11\$	
		F4	AD	01	AD 3C 0003D	MOVAB	1(R3), MESSAGE_DSC+4	1109
			52	F0	52 D7 00041	MOVZWL	MESSAGE_DSC, R2	1110
					02 12 00048	DECL	R2	
F4	BD		52		51 D4 0004A	LOCC	#34, R2, @MESSAGE_DSC+4	
					05 12 0004F	BNEQ	4\$	
			53		51 D0 0004C	CLRL	R1	
					52 D0 00051	MOVL	R1, CHAR_PTR	
					05 11 00054	BNEQ	5\$	1111
			51		AD C3 00056	MOVL	R2, R1	
					51 B0 0005B	BRB	6\$	
			53	F4	8F DD 0005F	SUBL3	MESSAGE_DSC+4, CHAR_PTR, R1	1112
			AD		01 FB 00065	MOVW	R1, MESSAGE_DSC	1111
					8F 9A 0006A	PUSHL	#PHN\$_SENDINGMAIL	1117
					8F 9E 0006F	CALLS	#1, PHN\$INFORM	
			0000G	CF	AD 3C 00074	MOVZBL	#128, COMMAND_BUF	1124
			08	AE	66 3C 00078	MOVAB	COMMAND_BUF+8, COMMAND_BUF+4	
			OC	AE	AE D0 0007F	MOVZWL	MESSAGE_DSC, R10	1127
			5A	F0	1E 2C 00083	MOVZWL	(R6), R9	1129
			59		67 0008A	MOVZWL	COMMAND_BUF, R8	1130
			58	08	2C 18 0008B	MOVZWL	COMMAND_BUF+4, R7	
			57	OC	1E C0 0008D	MOVL	COMMAND_BUF+4, R7	
58	20	0000'	CF		5A 2C 00093	MOVCS	#30, P.AAV, #32, R8, (R7)	
					67 00099	BGEQ	7\$	
					1D 18 0009A	ADDL2	#30, R7	
					5A C0 0009C	SUBL2	#30, R8	
			57		5A C2 0009F	MOVCS	R10, @MESSAGE_DSC+4, #32, R8, (R7)	
			58		06 2C 000A2	BGEQ	7\$	
					67 000A9	ADDL2	R10, R7	
					0D 18 000AA	SUBL2	R10, R8	
			57		06 C0 000AC	MOVCS	#6, P.AAW, #32, R8, (R7)	
			58		06 C2 000AF	BGEQ	7\$	
						ADDL2	#6, R7	
						SUBL2	#6, R8	

58	20	04	B6	59	2C	000B2	MOV C5	R9, @4(R6), #32, R8, (R7)	
		F8	AD 00150804	67		000B8			
			FC	8F	D0	000B9	7\$:	MOVL	#1378308, PRIV_VEC
				AD	D4	000C1		CLRL	PRIV_VEC+4
				7E	7C	000C4		CLRQ	-(SP)
			F8	AD	9F	000C6		PUSHAB	PRIV_VEC
				7E	D4	000C9		CLRL	-(SP)
		6B		04	FB	000CB		CALLS	#4, SYS\$SETPRV
			04	AE	9F	000CE		PUSHAB	STATUS_1
				7E	7C	000D1		CLRQ	-(SP)
		0C	AE	02	D0	000D3		MOVL	#2, 12(SP)
				AE	9F	000D7		PUSHAB	12(SP)
			0C	7E	D4	000DA		CLRL	-(SP)
				CF	9F	000DC		PUSHAB	P.AAX
			0000'	AE	9F	000E0		PUSHAB	COMMAND_BUF
		00000000G	00	07	FB	000E3		CALLS	#7, LIB\$SPAWN
			52	50	D0	000EA		MOVL	R0, STATUS_2
				7E	7C	000ED		CLRQ	-(SP)
				F8	AD	9F	000EF	PUSHAB	PRIV_VEC
				01	DD	000F2		PUSHL	#1
			6B	04	FB	000F4		CALLS	#4, SYS\$SETPRV
			08	52	E8	000F7		BLBS	STATUS_2, 8\$
	04		52	1C	E0	000FA		BBS	#28, STATUS_2, 8\$
				52	DD	000FE		PUSHL	STATUS_2
				13	11	00100		BRB	11\$
			0B	AE	E8	00102	8\$:	BLBS	STATUS_1, 9\$
	06	07	AE	04	E0	00106		BBS	#4, STATUS_1+3, 9\$
			50	AE	D0	0010B		MOVL	STATUS_1, R0
				02	11	0010F		BRB	10\$
				50	D4	00111	9\$:	CLRL	R0
				50	DD	00113	10\$:	PUSHL	R0
		0000G	CF	01	FB	00115	11\$:	CALLS	#1, PHN\$INFORM
				04	0011A			RET	

; Routine Size: 283 bytes, Routine Base: \$CODE\$ + 03CA

; 841 1157 1
; 842 1158 0 end eludom

.EXTRN LIB\$SIGNAL

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	232	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	1253	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
-LIB\$KEYO\$	0	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(1)
-LIB\$STAT\$	46	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(1)
\$PLITS	360	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

MISCCMDS
V04-000

MISCCMDS - Miscellaneous Phone Commands
PHN\$MAIL_CMD - Handle MAIL Command

C 9
16-Sep-1984 02:13:32
14-Sep-1984 12:53:27

VAX-11 Bliss-32 V4.0-742
[PHONE.SRC]MISCCMDS.B32;1

Page 35
(13)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
;\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	36	0	581	00:00.8
;\$255\$DUA28:[SYSLIB]TPAMAC.L32;1	42	22	52	14	00:00.1

COMMAND QUALIFIERS

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

; Size: 1253 code + 638 data bytes
; Run Time: 00:19.5
; Elapsed Time: 01:18.0
; Lines/CPU Min: 3568
; Lexemes/CPU-Min: 51864
; Memory Used: 161 pages
; Compilation Complete

0305 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY