ATARI® PROGRAM EXCHANGE

John Palevich

MANTIS BOOT TAPE DEVELOPMENT SYSTEM

Develop assembler cassettes on disk-based systems

Diskette: 40K (APX-20143)

User-Written Software for ATARI Home Computers

John Palevich

MANTIS BOOT TAPE DEVELOPMENT SYSTEM

Develop assembler cassettes on disk-based systems

Diskette: 40K (APX-20143)

MANTIS
(BOOT TAPE DEVELOPMENT SYSTEM)

by

John H. Palevich

Distributed By

The ATARI Program Exchange
P.O. Box 3705
Santa Clara, CA 95055

To request an APX Product Catalog, write to the address above, or call toll-free:

800/538-1862 (outside California)
800/672-1850 (within California)

Or call our Sales number, 408/727-5603

Table of Contents

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

OVERVIEW

A praying mantis is a large green insect that catches bugs. MANTIS BOOT TAPE DEVELOPMENT SYSTEM (MANTIS) is a powerful debugging tool that helps you through the entire program development cycle. You can read, write, and test self-booting cassette programs with ease. A full screen, three-format memory map and a comprehensive set of memory editing commands let you modify your program interactively. MANTIS works as follows. It saves DOS in high memory, simulates a 16K cassette-based ATARI Computer for testing purposes, and then restores DOS for further development work. You assemble and debug your boot tape program on the same ATARI Computer, without having to remove memory boards, unplug your disk drives, or make temporary cassette copies. In fact, you won't have to use an ATARI 410 Program Recorder until you've completely debugged your program.

If you're developing an assembly language program (especially one using graphics) to run on a 16K cassette-based ATARI 400 or ATARI 800 Computer, then MANTIS will save you both time and effort. With MANTIS you can:

1. Load and save DOS II binary-load files containing object code between diskette and computer memory

2. Read and write boot tapes

3. Examine and modify programs in memory

4. Execute boot tapes in memory, returning to MANTIS and DOS upon demand

REQUIRED ACCESSORIES

40K RAM
ATARI 810 Disk Drive
ATARI Assembler Editor Cartridge or ATARI Macro Assembler

CONTACTING THE AUTHOR

Users wishing to contact the author about MANTIS may write to him at:

Apt. F-211
175 Calvert Drive
Cupertino, CA 95014

GETTING STARTED

LOADING MANTIS INTO COMPUTER MEMORY

1. Connect your disk drive to your ATARI Computer and turn on the disk drive.

2. When the BUSY light goes out, open the disk drive door and insert the MANTIS diskette with the label in the lower right-hand corner nearest to you. Close the drive door.

3. Turn on your computer console and TV set and wait until the DOS loads into computer memory and the DOS menu displays on your TV screen.

4. Type L and press the RETURN key. Then type MANTIS and press the RETURN key. In a moment the MANTIS display screen will appear.

THE DISPLAY SCREEN

The MANTIS display screen looks approximately as follows:

```
      -------------------------------------------
1 | Mantis 1.0 (c)1982 John Howard Palevich   |
    ==========================================
2 | addr   08 19 2a 3b 4c 5d 6e 7f  01234567  |
    ==========================================
3 | 0700   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0708   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0710   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0718   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0720   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0728   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0730   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0738   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0740   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0748   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0750   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0758   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0760   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0768   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0770   00 00 00 00 00 00 00 00::xxxxxxxx  |
  | 0778   00 00 00 00 00 00 00 00::xxxxxxxx  |
    ==========================================
4 |  ->_                                      |
  |                                           |
  |                                           |
  |                                           |
    ==========================================
5 | Load,Save,Read,Write,Go,Cont,Modify,Dos   |
      -------------------------------------------
```

Figure 1   MANTIS Display Screen

The display screen is divided into five sections:

1. <u>Copyright Notice</u>

   The first line shows the version number and copyright message of your MANTIS program. It is for identification and legal purposes only, and doesn't change.

2. <u>Memory Map Label</u>

   The second line labels the columns of the memory map directly beneath it. "Addr" is short for "address", "08" to "7f" is the last digit of the address for that particular column of hex digits. Because data is displayed eight bytes to a line, each column can have two last digits, thus the pairs of numbers in the label line. The legend "01234567" serves the same purpose for the ASCII display. The memory map label line won't change; it's there only for your convenience.

3. Memory Map

The next sixteen lines of the screen show you a 128-byte portion of memory. Each line of the memory map displays eight bytes of memory.

The leftmost column of four-digit hex numbers tells where in memory that row of data is located.

The eight two-digit hex numbers to the right of the address are the hexadecimal representation of the eight data bytes. The leftmost byte is from the address in the addr column, the next byte is from addr+1, and so on. You use this hexadecimal representation to edit memory in the Modify Mode. Since you haven't loaded a boot tape, computer memory is empty, and so all the hex bytes are 00.

The eight characters on the right-hand side of the memory map show the eight bytes of data as ATASCII characters. This mode is useful for detecting and examining text, such as instructions, within your program. Since you haven't loaded a boot tape the 0 bytes show up as hearts.

The two character positions between the last digit of the hex display and the first character of the ASCII display appear to be empty, but in fact, they contain the graphic display of the memory map. Each line of data is displayed as graphics data, with each "1" bit shown black, and each "0" bit shown blue. Since all the bits are "0", the graphics display appears to be empty. Graphics mode is particularly useful for finding and editing character and player-missile data.

Note that the hexadecimal display for byte 700 appears in inverse video. Inverse video display indicates the editing cursor is over that byte, and that Modify Mode commands start at that byte.

4. Command Box

The four lines of text below the memory map are the command box. You type commands and MANTIS prints its replies in this box. The right arrow prompt (->) indicates MANTIS is waiting for you to type something. You can use all the screen editor control keys to edit your input.

5. Menu Line

The bottom line is the Menu Line listing the commands you give MANTIS. This line changes as you enter various submodes.

USING MANTIS

INTRODUCTION

MANTIS is a passive program. Like BASIC, it waits for you to command it to do something, carries out your command, and then waits for you to give it another command. The right-arrow (->) is like the READY prompt in ATARI BASIC. It tells you that MANTIS is waiting for your command.

The eight commands you can give MANTIS are listed on the Menu Line and described in detail below. Although both the menu line and the manual use the full mnemonic names for the commands, use only the first character to enter a command (for example, type L for Load DOS II Format File).

1. <u>L - Load DOS II Format File</u>

   Use this command to load the output of an assembler into MANTIS. A typical example is:

```
->L D:PROG.OBJ
File loaded OK.  Blocks:  1
Load addr:  $0700   Init addr:  $0707
->
```

   The example transfers the object code in the disk file PROG.OBJ into memory. The L command assumes that the file you ask it to read contains the object code for a boot tape. Sample boot tape programs appear in Appendix II.

2. <u>S - Save Boot Tape as DOS II Format File</u>

   Use this command to save a boot tape in memory to a file. A typical example is:

```
->S D:BOOT.SAV
->
```

   The example saves the boot tape in memory into diskette file BOOT.SAV. You can't use DOS to execute a "saved" boot tape, but you can use the L command in MANTIS to reload and execute the tape at a later time.

3. <u>R - Read Boot Tape Format File</u>

   Use this command to read a boot tape format file into memory. A typical example is:

```
->R C:
<Console beeps.  Press PLAY on the Program Recorder, and
then press RETURN on the the keyboard.  MANTIS reads in
first block of data.>
File has 27 blocks
Load addr:  $0700  Init addr: $0707 Re-reading
<Console beeps.  Rewind the Program Recorder and press
```

```
PLAY; press RETURN on the keyboard.  MANTIS reads the whole
boot tape.>
File loaded OK.  Blocks:  27
Load addr:  $0700  Init addr:  $0707
->
```

4. W - Write Boot Tape Format File

Use this command to write a boot tape format file from memory to tape. A typical use is:

```
->W C:
<Console beeps twice.  Put blank tape in Program Recorder,
press PLAY and RECORD, and type RETURN.  MANTIS writes the
boot onto cassette.>
->
```

5. G - Go To Boot Tape

Use this command to execute the boot tape in memory. The boot tape runs in an environment identical to a 16K cassette-based ATARI Computer. You can regain control by holding down the SHIFT key while pressing the SYSTEM RESET key, which returns you to MANTIS. A typical example is:

```
->G
<MANTIS display disappears, boot tape is run, user holds
down SHIFT key and presses SYSTEM RESET.  MANTIS screen
redisplays.>
Reset
->
```

Go also returns to MANTIS if your boot tape jumps to DOSVEC or if your second stage boot signals an error. Neither condition is likely, but in the first case MANTIS prints "Dosvec" and in the second it prints "Second Stage boot signalled error." If you haven't yet loaded a boot tape into memory, then MANTIS displays the message "Haven't loaded program yet." Load a boot tape program using the L or R command.

6. C - Continue Boot Tape

Use this commad to return to a boot tape program that was interrupted by SHIFT-SYSTEM RESET. Continue Boot Tape essentially executes a warm start, just as if you had pressed SYSTEM RESET rather than SHIFT and SYSTEM-RESET. A typical use is:

```
->C
<MANTIS display disappears and boot tape is RESET.>
```

If you have not used SHIFT-SYSTEM-RESET, then there is no program to return to and MANTIS displays the message "Can't Continue."

7. M - Modify Mode

Use this command to enter Modify Mode, which allows you to edit memory. Modify Mode optionally accepts an address to modify; otherwise, it uses the current address (that is, the byte in inverse video on the hex display). A typical example is:

```
->M 700
```

This example moves you into Modify Mode with the memory map cursor at memory location $700. The Modify Mode menu looks like this:

```
-----------------------------------------------------------------
|  Mod:   0-9,A-F,up,lf,rt,dn,arrows<>@?./;     <sp><bs><esc>  |
-----------------------------------------------------------------
```

Modify mode lets you edit memory quickly. Pressing the RETURN key is unnecessary after any Modify Mode command. Each command you type will be executed, and the result will be displayed in the memory map. The Modify Mode commands are as follows.

a. 0-9, A-F - Enter Hex Digit

Type any numeral, or the letters A through F, to enter that hexadecimal digit into the byte under the cursor. Like the exponent display of many scientific calculators, the rest of the digits are shifted left. For example, suppose the cursor is over the byte $6f and you want to enter the byte $54:

```
Byte under cursor        Then you type
       6f                           5
       f5                           4
       54                      <done>
```

If you make a mistake typing in the second digit, type both digits over again. Don't use the Back S key, since that key is used for another purpose while you're in Modify Mode.

b. Arrow Keys - Move Cursor Around in Memory

Use the arrow keys to move the cursor in the appropriate directions around the memory map. The up-arrow moves up one line (addr-8). The down-arrow moves down one line (addr+8). The left-arrow moves left one byte (addr-1) and the right-arrow moves right one byte (addr+1). If you're about to move off the screen, MANTIS scrolls the Memory Map up or down to keep the cursor on the screen.

For your convenience, you can use the arrow keys without pressing the CTRL key.

c. < > - Page Backward and Forward

Use the left angle bracket key (<) to move the cursor up one page of memory (addr-128), displaying the previous 128 bytes of memory. Use the right angle bracket key (>) to move the cursor down one page of memory (addr+128), displaying the next 128 bytes of memory. These two keys let you skim through memory quickly.

d. <u>@ - Look at Address</u>

Use the at-sign key (@) to move the cursor to the address in memory pointed at by the bytes under and immediately to the right of the cursor. For example, if the cursor is at memory location $700, the byte under the cursor is $43, and the byte to the right of the cursor is $21, then typing @ positions the cursor at memory location $2143. This command makes it easy to follow jumps and calls while debugging your code. Use the "." command (Enter Address) to return to the cursor's previous position.

e. <u>?- Find Address</u>

Use the question mark key (?) to move the cursor to the next occurrence of the two bytes under and to the right of the cursor. You can use "?" to search for subroutine calls, immediate loads, pointers, and so on. If there is no occurrence of those two bytes between the present cursor position and $4000, MANTIS prints the message "Didn't find it." and leaves the cursor where it was.

f. <u>. - Enter Address</u>

Use the period key (.) to move the cursor directly to an address of your choice. The Menu line changes to:

```
 ----------------------------------------
| Enter address, any non-hex char ends. |
 ----------------------------------------
```

and the cursor moves to $0000. As you type in your new address, the cursor jumps to it one byte at a time. If you make a mistake, retype all four digits of the address. When you've finished entering the address, type any non-hex character (e.g., press the space bar) to return to Modify Mode. For example, to enter address $1234:

```
Cursor at address        Then you type
    0000                       1
    0001                       2
    0012                       3
    0123                       4
    1234                 space <to exit>
```

g. <u>/ - Complement and Move Right</u>

Use the slash key (/) to reverse the bits of a byte. This command also moves the cursor rightward one byte, so you can complement several bytes in a row by repeatedly typing "/".

h. <u>; - Zero Byte and Move Right</u>

The semicolon key (;) is equivalent to typing "00" <space> in that it zeros the byte at the cursor and moves on to the next byte. Typing a string of semicolons is an easy way to zero out large portions of memory.

i. <u>SPACE - Move Cursor Right</u>

Press the space bar to move the cursor right one byte (addr+1). You can accomplish the same thing by pressing right-arrow. The alternate command is simply a convenience.

j. <u>BACK S - Move Cursor Left</u>

Use the BACK S key to move the cursor left one byte. You can accomplish the same thing by pressing the left-arrow. The alternate command is simply a convenience.

k. <u>ESC - Exit Modify Mode</u>

Press the ESC key to return to the Main Menu.

8. <u>D - Return to DOS II</u>

```
-->D
```

Use this command to leave MANTIS BOOT and return to DOS II.

BOOT TAPES

INTRODUCTION

Boot tape format is a file used to distribute automatically loading, machine language programs. It's like a poor man's ROM cartridge; it's much cheaper to produce than a ROM cartridge is, yet is almost as easy to use as a cartridge. The ATARI Technical User's Notes contain complete details on boot tapes. Here's a simplified diagram of a boot tape:

Byte	Value
0	0
1	number of 128 byte blocks of data in whole boot tape
2	low byte of load address
3	high byte of load address
4	low byte of init address
5	high byte of init address
6	first byte of second-stage-boot
...	rest of program

In addition to MANTIS, your diskette contains three files:

1. BOOT.AED - A simple boot tape in ATARI Assembler Editor form

2. BOOT.MAC - A simple boot tape in ATARI Macro Assembler form

3. BOOT.OBJ - the output of the ATARI Macro Assembler version of the example; essentially identical to the output of the Assembler Editor version

MAKING A BOOT TAPE

Load MANTIS, and then issue the command:

```
L D:BOOT.OBJ <RETURN>
```

followed by the command:

```
G <RETURN>
```

The program endlessly prints the character "!" on the screen.

Now press the SYSTEM RESET key. Notice that the screen clears, and the program again starts to fill the screen with "!".

Hold down the SHIFT key and press SYSTEM RESET to return to MANTIS. Look at the Memory Map. In the ATASCII format at memory location $721 you should see the "!" character that's being printed. Type

```
M 721 <RETURN>
```

and then type

```
00
```

to turn the "!" into a heart. Press the ESC key to return to the Main Menu, and then type

```
C <RETURN>
```

to continue the program. The screen clears (just the way it did when you pressed SYSTEM RESET), but now the program prints hearts instead of "!".

Go back to MANTIS (by pressing SHIFT-SYSTEM RESET) and type

```
W C: <RETURN>
```

The console will beep twice. Put a blank boot tape into your Program Recorder, press PLAY and RECORD, and then press the RETURN key. MANTIS will write the boot tape onto the cassette tape.

To see that you've really made a boot tape, turn off your system and then turn on your ATARI Computer while holding down the START button. The console will beep. Rewind your tape and press PLAY. Now press RETURN on the console and wait while the boot tape loads. As soon as it does, the Program Recorder will stop and the screen will fill with hearts. Congratulations! you've made your first boot tape!

TROUBLESHOOTING

If you type a command MANTIS doesn't understand, it replies:

```
Pardon?
->
```

If the program you're testing "runs wild", then it's quite possible that MANTIS could be destroyed. If you press SHIFT and SYSTEM RESET while testing your program and nothing happens, then turn off your computer and start over to regain control.

It is possible to modify the saved copy of DOS II and the running code for MANTIS via the "M" command, but such modifications will almost certainly lead to system lockup. To avoid these problems, don't change data above $3FFF.

If file I/O errors occur while you're using the L, S, R , or W commands, MANTIS stops the I/O and prints:

```
File I/O Error ###
->
```

where "# # #" is the standard decimal number of the file error. For example, if you aborted I/O by pressing the BREAK key, then # # # would be 128.

If you try to load a nonboot tape load file, you will get the error message:

```
Bad boot tape format.
->
```

If during the load, MANTIS encounters a record that exceeds $3FFF, the program stops loading the file and prints:

```
Data block out of range.
->
```

DOS II Run and Init addresses are ignored, but the data block of a load file must be the beginning of the boot tape. The rest of the data blocks in the load file need not be contiguous or in ascending order.

WHAT MANTIS CAN'T DO

As good a program as MANTIS is, it can't do several things:

1. MANTIS can't debug boot tapes requiring more than 16K of memory to run. Since the vast majority of cassette-based ATARI Computers have only 16K of memory, you should either keep your program small or decide not to support cassette users. A machine language program requiring 24K or more of memory is cetain to require DOS, too, if for no other reason than to reduce the loading time. A 16K boot tape takes about five minutes to load. No user is going to wait much longer than that. If you find yourself running out of room, either strip down your program or give up on the boot-tape market and switch to a disk-based debugging utility.

2. MANTIS can't return to where it left off. It would be nice to continue a program from exactly where it was when you pressed SHIFT and SYSTEM RESET. It would be even nicer to provide single stepping and breakpoints. Unfortunately, the ATARI OS fails to save enough state (e.g., PMBASE,HPOSPX,HITCLR) to allow this.

3. MANTIS files produced by the S command won't run "as is " under DOS II. That is. you must load the file back into MANTIS and use the G command to run it. Though not perfect, this method is still faster than reloading the file from tape.

Appendix I. Command Summary

TABLE 1

MAIN MENU - Press RETURN after entering command

L D:BOOT.OBJ	Load binary-load file D:BOOT.OBJ into RAM from disk
S D:BOOT.OBJ	Save binary-load file D:BOOT.OBJ from RAM onto disk
R C:	Read boot tape file into RAM from cassette
W C:	Write boot tape file from RAM onto cassette
G	"GO" - Run boot tape in RAM
C	Continue execution of boot tape after Reset
M nnn	Modify RAM, beginning at $nnn
D	DOS - return to DOS

TABLE 2

MODIFY MENU -- Don't press RETURN

0 - 9, A - F	Enter hexadecimal digit
arrow keys	Move cursor in appropriate direction
< >	Move cursor up or down 128 bytes
@	Goto address
?	Search for address
.	Enter address
/	Complement byte and move right
;	Zero byte and move right
space bar	Right one byte
BACK S	Left one byte
ESC	Exit modify mode

Appendix II
Program Listings

Here is an assembly language "wrapper" that you can "wrap around" your programs to turn them into boot tapes:

```
110             *= $700 ;origin of program
120     ZZZBEG  .BYTE 0,ZZZEND-ZZZBEG/128+1     ;# of blocks
130             .WORD ZZZBEG     ;origin of program
140             .WORD ZZZINI     ;initialization of program
150             CLC     ;empty second-stage-boot
160             RTS
170     ZZZINI  LDA #$3C         ;turn off cassette motor
180             STA $D302
190             JMP START        ;Start of program.
200

...     <body of program>        ...

9990    ZZZEND  .END             ;last byte of program
```

Figure 2: Atari Assembler Editor Cartridge Boot Tape Wrapper

```
        org $700         ;origin of program
zzzbeg  db 0, [zzzend-zzzbeg]/128+1 ;Round up # of blocks
        dw zzzbeg,zzzini

        clc     ;second stage boot
        rts

zzzini  lda #$3C         ;turn off cassette motor
        sta $D302
        jmp start        ;start of program

...     <Body of program>        ...

zzzend  END     ;last byte of program
```

Figure 3: Atari Macro Assembler Boot Tape Wrapper

```
        db 'This character ->'
char    db '!'  ;character to be printed.
        db '<- is the one'

start   lda #27
        jsr printc      ;print <escape>
        lda char
        jsr printc      ;print character
        jmp start

printc  pha     ;save A
        ldx #0
        txa
        sta $348        ;icbll[0]
        sta $349        ;icblh[0]
        lda #11         ;putchr
        sta $342        ;iccom[0]
        pla
        jsr $E456       ;ciov
        rts
```

Figure 4: BOOT source code

APX ATARI PROGRAM EXCHANGE
P.O. Box 3705
Santa Clara, CA 95055

Review Form

We're interested in your experiences with APX programs and documentation, both favorable and unfavorable. Many of our authors are eager to improve their programs if they know what you want. And, of course, we want to know about any bugs that slipped by us, so that the author can fix them. We also want to know whether our instructions are meeting your needs. You are our best source for suggesting improvements! Please help us by taking a moment to fill in this review sheet. Fold the sheet in thirds and seal it so that the address on the bottom of the back becomes the envelope front. Thank you for helping us!

1. Name and APX number of program.

____

____

2. If you have problems using the program, please describe them here.

____

____

____

3. What do you especially like about this program?

____

____

____

4. What do you think the program's weaknesses are?

____

____

____

5. How can the catalog description be more accurate or comprehensive?

____

____

6. On a scale of 1 to 10, 1 being "poor" and 10 being "excellent", please rate the following aspects of this program:

_______ Easy to use
_______ User-oriented (e.g., menus, prompts, clear language)
_______ Enjoyable
_______ Self-instructive
_______ Useful (non-game programs)
_______ Imaginative graphics and sound

7. Describe any technical errors you found in the user instructions (please give page numbers).

____

____

____

8. What did you especially like about the user instructions?

____

____

____

9. What revisions or additions would improve these instructions?

____

____

____

10. On a scale of 1 to 10, 1 representing "poor" and 10 representing "excellent", how would you rate the user instructions and why?

____

____

11. Other comments about the program or user instructions:

____

____

____

From

____

____

____

STAMP

ATARI Program Exchange
P.O. Box 3705
Santa Clara, CA 95055

[seal here]