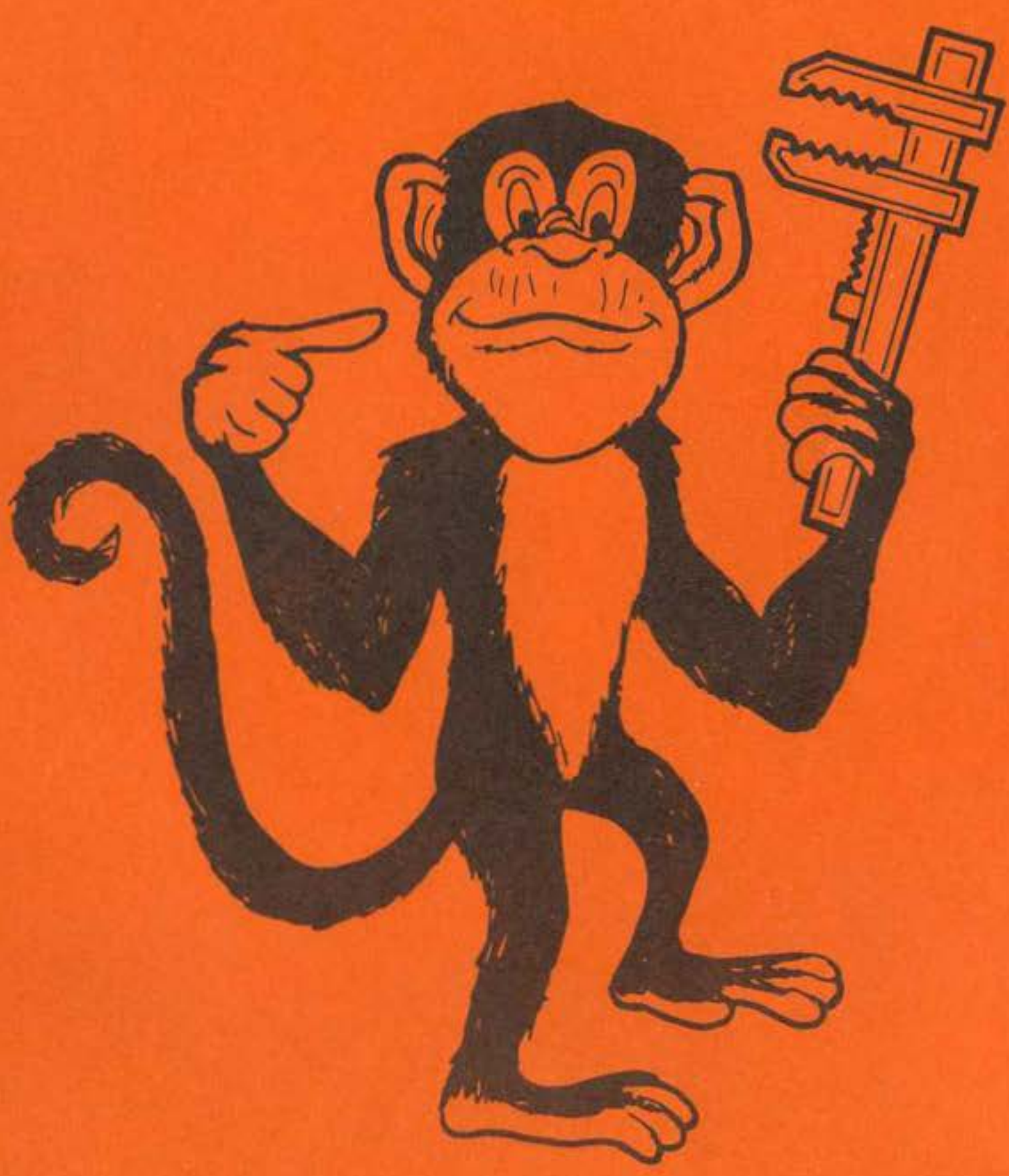
THE
MONKEY WRENCH II

USERS GUIDE



A PROGRAMMERS AID FOR ATARI 800 COMPUTERS

THE MONKEY WRENCH II (tm)

USERS GUIDE

WHAT IS THE MONKEY WRENCH ?

The Monkey Wrench is a machine language cartridge which extends the operating capability of the ATARI 800 computer. When installed into the ATARI, the Monkey Wrench provides 18 new direct mode BASIC commands. The commands are:

Auto Line Numbering	- Provides new line numbers when entering BASIC program lines.
Renumber	- Renumbers BASIC's line numbers including internal references.
Delete Line Numbers	- Removes a range BASIC line numbers.
Variables	- Display all BASIC variables and their current value.
Scrolling	- Use the START & SELECT keys to display BASIC lines automatically. Scroll up or down BASIC program.
Find String	- Find every occurrence of a string.
Xchange String	- Find every occurrence of a string and replace it with another string.
Move Lines	- Move lines from one part of program to another part of program.
Copy Lines	- Copy lines from one part of program to another part of program.
Formatted List	- Print BASIC program in special line format and automatic page numbering.

Disk Directory - Display Disk Directory.

Change Margins - Provides the capability to easily change the screen margins.

Memory Test - Provides the capability to test RAM memory.

Cursor Exchange - Allows usage of the cursor keys without holding down the CTRL key.

Upper Case Lock - Keeps the computer in the upper case character set.

Hex Conversion - Converts a hexidecimal number to a decimal number.

Decimal Conversion - Converts a decimal number to a hexdecimal number.

Monitor - Enter the machine language monitor.

In addition to the BASIC commands, the Monkey Wrench also contains a machine language monitor with 16 commands used to interact with the powerful features of the 6502 microprocessor.

GETTING STARTED

Normally, the Monkey Wrench is installed at the same time as the ATARI BASIC cartridge. The Monkey cartridge is inserted into the RIGHT CARTRIDGE slot. Once the Monkey Wrench is inserted into the RIGHT slot, simply turn on the power switch (see NOTE). The Monkey Wrench will automatically initialize itself and display:

```
READY
 THE MONKEY WRENCH II
  COPYRIGHT 1983 BY
EASTERN HOUSE SOFTWARE
```

Now you're ready to MONKEY around with the commands !!!

NOTE: Since the right slot is not normally used, the contacts in the slot may become dirty and cause the MONKEY WRENCH to malfunction when first installed. To help clean the contacts, wipe some alcohol on the fingers of the Monkey Wrench board and plug it in and then out of the right slot several times. During normal usage, the fingers on the board can be cleaned using a pencil eraser.

*** IMPORTANT NOTE ***

It is important to note that the Monkey Wrench is a program development aid. It is intended to be used with ATARI BASIC. It contains very complex software. For this reason, it should be removed from the computer when running programs or games. Running a program with it installed could cause the computer to function improperly. (This is mainly true when the BASIC program does certain POKE's into memory.) Thus, do so at your own risk.

BASIC COMMAND DETAILS

The 18 new BASIC commands are designed to give you those extra features that are not provided by the ATARI BASIC. These commands must be executed in the direct mode. That is, they cannot be entered into a BASIC program. They must typed on the screen without a line number.

All of the Monkey Wrench commands must be entered with the greater than symbol (>) preceding the command syntax. In addition, all of the commands must be entered with the greater than symbol (>) in the first character position on the line. Finally, all commands with parameters must contain at least one space between all of the parameters (as shown in examples).

*** IMPORTANT NOTE ***

In ATARI BASIC, it is possible for a line to be longer than three physical lines after it has been entered. This is a poor programming practice which will cause problems at some future time. Due to the nature of the Monkey Wrench, it will truncate all BASIC lines longer than 120 characters in length (approximately three physical lines). Thus always ensure all BASIC lines are less than 120 characters.

AUTOMATIC LINE NUMBERING

>A (line number) (increment value)

The >A command provides automatic line numbering for entering your BASIC program. The first parameter entered is the starting line number. The second parameter is the increment value used to determine the value of the next line number. If the increment value is not entered, the increment value defaults to 10. Now whenever you enter your BASIC line statements and depress RETURN, the next line number will automatically be displayed on the next line of the screen. This feature allows for full screen editing and still maintains the auto numbering mode.

To exit the automatic line numbering mode, depress the BREAK key.

EXAMPLES:

```
>A  10  10                          * Start automatic
10 REM The old dog has flees!!      * line numbering
20 REM But he doesn't care!!        * at line 10 and
30                                  * increment by 10


>A  5000  1                         * Start automatic
5000 GOTO 1000                      * line numbering
5001 IF A=123 THEN B=321            * at line 5000 and
5002 LET C=0                        * increment by 1
5003
```

EXCHANGE CURSOR KEYS

>E

When the Monkey Wrench is initialized at power-up, the cursor keys (up, down, left and right) are setup to allow their usage without holding down the CTRL key. This feature makes it much more easier to use the screen editing capabilities of the ATARI. To use the +, -, = and * characters, hold down the CTRL key. If you don't wish to use this feature, type in the >E command to exchange the characters to the normal mode. If at a later time you wish this feature restored, type >E again to exchange the keys.

DELETE RANGE OF LINE NUMBERS

>D (starting line number) (ending line number)

The >D command deletes a range of BASIC line numbers. The first parameter specifies the starting line number to be deleted. The second parameter specifies the last line number to be deleted.

EXAMPLES:

```
>D  0  50                    * Delete lines 0 to 50

>D  100 200                  * Delete lines 100 200

>D 32000 32500               * Delete lines 32000 to 32500
```

When the RETURN key is depressed, the lines will be deleted. The ATARI buzzer will sound when the lines have been deleted.

*** IMPORTANT NOTE ***

Due to a problem in ATARI BASIC, deleteing lines may cause the computer to lockup. This problem can happen whether or not the Monkey Wrench is installed. Although it is not a serious problem and doesn't happen very often, it can happen. Therefore, be aware of this problem and always keep a current backup copy of the program.

UPPER CASE LOCK

>U

Turn the upper case feature on or off. How many times have you went to depress the RETURN key only to depress the CAPS LOWR key instead. ATARI BASIC does not like lower case; therefore, this is a very annoying occurrence. The upper case feature will keep the computer in the upper case character set and disable the lower case character set.

When the computer is turned on, the upper case lock feature is enabled. To disable the upper case lock feature and return to normal key usage, type >U. To enable the upper case lock feature again, type >U.

DECIMAL TO HEXIDECIMAL CONVERSION

>\# (decimal number)

Convert a decimal number to hexidecimal number. When the RETURN key is depressed, the hexidecimal number will be displayed on the next line. The maximum decimal number is 65535.

EXAMPLES:

```
>#  255
00FF

>#  10
000A

>#  65535
FFFF
```

HEXIDECIMAL TO DECIMAL CONVERSION

>$ (hexidecimal number)

Convert a hexidecimal number to a decimal number. When the RETURN key is depressed, the decimal number will be displayed on the next line. The maximum hexidecimal number is FFFF.

EXAMPLES:

```
>$  FF
255

>$  A
10

>$  FFFF
65535
```

MEMORY TEST

>T (start hex address) (end hex address)

One of the most handiest programs you can have for your computer is a MEMORY TEST program. It allows you to be sure that your RAM memory is functioning properly. This is particularly useful when you purchase additional memory modules. You only have a limited time period on your guarantee. How do you know that it is okay ? You really don´t unless you test it. Also when one of your programs isn´t working properly, is it because the problem is in the program or in RAM memory ? With the memory test, you can find out.

The test checks the memory for storage retention, an open, shorted or non-functioning data and address lines 0 through 7. This is done by writing 00 01 ... FF 00 01 ... FF continually throughout the memory range for the first pass. When this has been written, it is checked to validate the data. On the next pass 01 02 ... FF 00 01 ... FF is written and checked. This continues for 256 passes until all possible combinations of bit patterns have been checked.

The >T command parameters (start & end addreses) are hexidecimal addresses. The following are examples of the minimum and maximum practical ranges of memory that should be tested.

EXAMPLES:

```
>T  0700  7C20          * Test memory from hex 700 to hex 7C20.
                        * For 32K, 40K, or 48K of memory.

>T  0700  5C20          * Test memory from hex 700 to hex 5C20.
                        * For 24K of memory.

>T  0700  3C20          * Test memory from hex 700 to hex 3C20
                        * For 16K of memory.

>T  0700  1C20          * Test memory from hex 700 to hex 1C20
                        * For 8K of memory.
```

When the RETURN key is depressed, the screen will display TESTING. If any errors are found, they will be outputted as shown on the next page. At the end of the memory test, the screen will display TEST FINISHED and sound the buzzer. The memory test performs a lengthy but exhaustive test of RAM memory. Therefore, if you´re testing alot of memory, be prepared to wait!!! If you wish to terminate the test early, depress the SYSTEM RESET key.

When memory errors occur during the test, they will be outputted in the following format:

ADDRESS OF ERROR	DATA PATTERN STORED BY TEST	DATA READ BY TEST
xxxx	yy	zz

Example Printout of Memory Errors

3D90	F1	F0
3D90	F3	F2
3D90	F5	F4
3D90	F7	F6

To the beginning computerist, the printout of memory errors won't mean much. But it does indicate a problem in one of the memory modules. Remove one of the memory modules and test the RAM that is left. If the errors occur again, continue to swap memory modules until the errors no longer occur. At that time, the memory module not installed will be the bad one.

Since BASIC and the Monkey Wrench take up the upper 16K of memory, it is not possible to test this RAM if you have 40K or 48K of memory. In order to test this RAM you must move or exchange the third memory module with the second module and repeat the memory test. Likewise, in order to do a complete test on the lower part of memory, exchange the first memory module with the second module and repeat the test. Finally, before starting a test, ensure the DOS or a program is not loaded. These programs will be destroyed by the test.

If you would like to see the memory test at work, try testing part of the screen memory. For example, clear the screen and type:

```
>T  7E00  8000      * For 32K, 40K, or 48K of memory
>T  5E00  6000      * For 24K of memory
>T  3E00  4000      * For 16K of memory
>T  1E00  2000      * For 8K of memory
```

RENUMBER BASIC PROGRAM

>R (start line number) (increment value)

The >R command will renumber all BASIC lines in a BASIC program. It will renumber not only the line numbers but also all references to the line number by the following BASIC commands: GOTO, GOSUB, IF THEN, ON GOTO, ON GOSUB, RESTORE, and TRAP.

If the renumber routine finds that a reference is made to a nonexistent BASIC line number, it will change the reference to line number 32767. Therefore, it will be easy to locate when scanning a listing of the program. In addition, when the BASIC program is RUN, an error message will be given.

The first parameter of the >R command specifies the starting line number. That is, the first line number you want the program to have. The second parameter specifies the increment value (or the number of line numbers) between lines. The increment value can be from 1 to 255. If the increment value is not entered, it will default to 10.

EXAMPLES:

```
>R  10              * Renumber program with the first new
                    * line number being 10. An increment
                    * value of 10 will be assumed. The new
                    * line numbers will look like 10, 20,
                    * 30, 40, 50, etc.

>R  1  1            * Renumber program with the first new
                    * line number being 1 and an increment
                    * value of 1. The new line numbers will
                    * look like 1, 2, 3, 4, 5, 6, 7, etc.

>R  32000  50       * Renumber program with the first new
                    * line number being 32000 and an increment
                    * value of 50. The new line numbers will
                    * look like 32000, 32050, 32100, etc.
```

Additional Facts About Using Renumber !!!

1. The renumber command uses the screen RAM memory as a buffer. Therfore in graphics mode 0, the longest BASIC program that can be renumbered is 478 lines. An error message PROGRAM TO LONG FOR SCREEN MEMORY will be given if the program is to long. If necessary, the graphics mode may be changed to GRAPHICS 6 before renumbering the program. This will double the number of lines that can be renumbered.

2. The screen will go blank during the time the program is being renumbered. This is normal and no cause for alarm.

3. The maximum line number used by the >R command is 32766. Thus use the increment value carefully at high starting line numbers. An error message OUT OF NUMBERS will be given if this occurs. In this case, the BASIC program will be only partially renumbered and therefore is useless. Reload the BASIC program and renumber again.

4. There are many ways of constructing BASIC programs. Therefore it is impossible for the renumber routine to know exactly what the programmer had in mind. The following is a list of examples of BASIC commands which the renumber routine may or may not act upon.

* The renumber routine will not renumber BASIC statements which use symbolic labels as line numbers. For example,

```
10 LET CAT=50
20 GOTO CAT
```

* Be carefull of the following BASIC format:

```
10 IF A=1 THEN 100 + A
```

The above format is OK. The 100 will be renumbered.

```
10 IF A=1 THEN A + 100
```

In this format, the 100 will not be renumbered.

Note: This applies not only to the IF THEN command but also to the other commands.

FIND STRING COMMAND

>F d string d (start line) (end line)

The >F command will search through BASIC lines and find all occurrences of a string and print to screen. The 'd' in the command format is a delimiter character. It is used to indicate the start and end of the string to be searched for in the program. The delimiter character can be any character entered on the keyboard. The same delimiter character used to indicate the start of the string MUST also be used to indicate the end of the string.

The string can be any characters entered from the keyboard (except the character used as the delimiter and the % character). The percent character (%) is used as a 'do not care' character. That is, any character found at that place in the string will be accepted as part of the string during a search. For example, the string ABC%%FG will cause any of the following strings to be accepted by the find command as valid -- ABCDEFG, ABC12FG, ABCXYFG, and ABC FG.

A range of line numbers can be specified by entering the start and end line numbers. If no line numbers are given, the find command will search through all BASIC lines. A space must seperate the start and end line numbers.

EXAMPLES:

>F /ABCD/ 50 275	Find all ocurrences of the string ABCD in lines 50 through 275.
>F /TRAP 2000/	Find all occurrences of the string TRAP 2000 in all BASIC lines.
>F /GO%%/	Find all occurrences of the string GO followed by two 'do not care' characters. Note that in this case, the find command will display all lines which contain GOTO commands as well as GOSUB commands and anything else that matches.

>F ?TOTAL/NUM?	Find all occurrences of the string TOTAL/NUM. Notice in previous examples a slash (/) was used as the delimiter. In this case however, It was necessary to use the slash character in the string to indicate a divide operation. Thus it was necessary to use a different delimiter. In this case, a question mark was used (however any other character could have been used as long as it doesn't appear as part of the string.

Notes:

1- To stop the find command before it has finished, depress the space bar.
2- To stop the display on the screen without stopping the find command, use the CNTL-1 key.
3- The string used by the find command mut be entered exactly as it is normally listed to the screen. For example, GOTO100 will not be found; where as GOTO 100 will be found. This is because a space character always seperates the GOTO command from the line number.
4- The maximum number of characters that may be used in the string is 40.

IF CHANGING B$ TO LANG$
BB$ will become BLANG$

XCHANGE STRING COMMAND

The last letter in all strings can be changed (watch out!)

>X d search string d replace string d (start line) (end line)

The >X command will search through BASIC lines and find all occurrences of a string and replace with another string. The xchange command works like the find command but in addition to finding the string it also replaces the string it found with a new string (replace string) as shown in the command format. All lines where a string is found and replaced will be displayed on the screen.

As with the find command, the 'd' is used as a delimiter to indicate the start and end of each string. It is also possible to use the 'do not care' character in the search string of the xchange command. Note: The total number of characters used for the search and replace strings must not exceed 40 characters in length.

A range of line numbers can be specified by entering the start and end line numbers. If no line numbers are given, the xchange command will search through all BASIC lines. A space must seperate the start and end line numbers.

EXAMPLES:

>X /START/FINISH/ 1000 2500	Search through BASIC lines 1000 to 2500 for all occurrences of the string START and replace it with the string FINISH.
>X /GOTO 100/GOTO 500/	Search through all BASIC lines for all occurrences of the string GOTO 100 and replace it with the string GOTO 500.
>X /SUM=%%/SUM=0/	Search through all BASIC lines for all occurrences of the string SUM= followed by two 'do not care' characters and replace it with SUM=0.
>X /GOSUB 5000//	Search through all BASIC lines for all occurrences of the string GOSUB 5000 and replace it with a null string. In this example the last two delimiters are next to each other indicating that there is no replace string. This feature in effect allows the user to delete any string from the BASIC line.

>X /%//

Try this example. It finds any character and deletes it. If this command is allowed to finish, it will delete every line from the computer.

Notes:

1- To stop the xchange command before it has finished, depress the space bar.
2- To stop the display on the screen without stopping the xchange command, use the CNTL-1 key.
3- The string used by the xchange command mut be entered exactly as it is normally listed to the screen. For example, GOTO100 will not be found; where as GOTO 100 will be found. This is because a space character always seperates the GOTO command from the line number.
4- The maximum number of characters that may be used in both the search and replace strings is 40.
5- The xchange command searches for string and replaces that string with a new string. It does not do any syntax checking. For example, if GOTO 50 is replaced with GOTTO 100, a normal BASIC error message will result. The xchange command will not stop its processing. Therefore, the user should use the xchange command carefully and, of course, always keep a backup copy of the program.

SET MARGINS

>M (left margin) (right margin)

The >M command makes it possible to easily change the screen margins. The first number in the parameter specifies the left margin; the second number in the parameter specifies the right margin. If the second parameter is not given, only the left margin will be changed.

EXAMPLES:

```
>M  5  35                    * Set margins to 5 and 35

>M  0 39                     * Set margins to 0 and 39

>M  2                        * Set left margin to 2
```

The lowest left margin number is 0 and the highest right margin number is 39. Don't specify a left margin number greater than the right margin. If you do, use the SYSTEM RESET key to reset the margins.

SCROLL ENABLE COMMAND

>SE

The scroll enable command allows the user to scroll (list) BASIC lines to the screen by using the START and SELECT keys. The START key will scroll down BASIC lines; that is, to higher BASIC line numbers. The next higher BASIC line number will be displayed at the bottom of the screen with all other lines being moved up the screen. The SELECT key will scroll up BASIC lines; that is, to lower line numbers. The next lower line number will br displayed at the top of the screen.

As discussed above, the START and SELECT keys will cause the next higher or lower line number to be displayed on the screen. How does the scroll routine know what the next line is to be ? This is accomplished by looking at the current line numbers on the screen. If the screen is clear, the scroll will start at the first or last BASIC line number depending on whether the SELECT or START key was depressed.

In order to experiment with the scroll feature, load a BASIC program which contains at least 30 lines. Clear the screen. Now depress and hold the START key. As seen, each BASIC line number is displayed at the bottom of the screen and moves up the screen as each new line is displayed. How do you know when the last line is displayed ? The ATARI ´buzzer´ will sound. Now depress and hold the SELECT key. As seen, the next lower BASIC line number is displayed at the top of the screen with all other lines moving down the screen. Hold down the SELECT key until the the first line is displayed. How do you know when the first line is displayed ? Again, the ATARI ´buzzer´ will sound.

Clear the screen. Type ia a number which represents a BASIC line number that is near the middle of the BASIC program. DO NOT DEPRESS RETURN! Depress the START key and observe the next higher line number being displayed at the bottom of the screen. This feature allows the user to scroll starting at any line desired.

Another feature of the scroll routine is the ability to insert blank lines using the OPTION key. Move the cursor to the place on the screen you wish to insert a blank line and depress the OPTION key. The blank line will be inserted and the ATARI ´buzzer´ will sound.

Note 1- The scroll feature is enabled when the computer is turned on. To disable the scroll feature, type >S.

Note 2- The START and SELECT keys MUST only be used during Graphics Mode 0. Using the scroll feature in any other mode will cause the computer to malfunction.

FORMATTED LIST COMMAND

```
>L (start line) (end line)
>LP (start line) (end line)
```

The formatted list command list BASIC lines to the screen (and/or printer using >LP command). This command does the same thing as the normal list command. However, the >L command formates the line when multiple BASIC commands are in the same line. That is, whenever a BASIC line contains more then one command, the commands will be left justified on the screen or printer. In addition, the command also prints page numbers at the top of each page. This is very usefull when sending listing to the printer via the >LP command. The following is an example of the printout.

```
          PAGE  1

110 GOSUB 2500
    :GOSUB 3000
120 IF A=B THEN 10
    :IF A<B THEN 20
    :IF A>B THEN 30
130 X=100
    :Y=210
```

This formatted list allows easier viewing of a BASIC program when multiple commands per line are used. The Range of lines to be listed may be specified by entering the start and end line numbers. If no line numbers are entered, all lines will be listed. The >L and >LP commands can be aborted by depressing the space bar.

Note: DO NOT try to do any screen editing of the formatted lines. BASIC will not recognize the entire line.

VARIABLES COMMAND

```
>V
>VP
```

The >V command will display all BASIC variables and their current value. The >VP command will also send the variables to the printer. This includes variables used as constants and strings. In the case of arrays, only the array name will be displayed (the content of the array will not be displayed). Anytime a string or array name is displayed, the current DIM value(s) will also be displayed. If the string or array has not been dimensioned, a 'U' will be displayed to indicate it is 'un-dimensioned'.

Example:

```
>V
NUM =1234
A$ [16 ] =ABCDEF
X$ [U ]
XYZ( [18,20]

*4
```

In the above example, the the variable NUM and its current value is shown. The second line contains a string A$, the dimension [16] and the current contents of the string ABCDEF. String variables are always followed by a dollar sign $. The third line in the example is also a string variable. However, the string has not been dimensioned which is shown by the [U]. The forth line contains an array variable. As shown, the array variable name ends with a (and the current dimensions follow. The last line contains an asterisk and the number of variables in the program.

DISK DIRECTORY COMMAND

```
>>x
```

The >>x command will display the disk directory of the disk drive where x is the disk drive number.

EXAMPLES:

```
>>2    - Display directory on disk drive number 2

>>4    - Display directory on disk drive number 4

>>     - Display directory on disk drive number 1.
         As shown in this example, no drive number was entered.
         In this case, the drive number defaults to 1.
```

COPY COMMAND

>CO (start line) (end line) (destination line)

The >CO command copies BASIC lines from one area of the program to another. The range of lines to be copied is specified by the start line and end line parameters as shown in the command format. The destination line parameter specifies the first line number where the new lines will be stored. The destination line number will be incremented by 1 for each line copied. For example, if lines 10 thru 100 are to be copied to line 32000, the command would look like >CO 10 100 32000. After the return key is depressed, the lines being copied will be displayed on the screen. To abort the copy command before it has completed, depress the space bar. When the last line has been copied, the buzzer will sound.

IMPORTANT: The copy command copies the lines exactly as they are stored. Therefore, all line references must be corrected by the user. Also, it is possible to copy lines over existing BASIC lines. Therefore, be careful when specifing the destination address (unless it is intended to overwrite the lines).

Note: It is not possible to specify a destination line number that falls between the start and end line numbers. An error will be given if this happens.

MOVE COMMAND

>MO (start line) (end line) (destination line)

The >MO command moves BASIC lines from one area of the program to another. The original lines are deleted. The range of lines to be moved is specified by the start line and end line parameters as shown in the command format. The destination line parameter specifies the first line number where the new lines will be stored. The destination line number will be incremented by 1 for each line moved. For example, if lines 10 thru 100 are to be moved to line 32000, the command would look like >MO 10 100 32000. After the return key is depressed, the lines being moved will be displayed on the screen. The original lines specified by start and end are deleted. To abort. the move command before it has completed, depress the space bar. When the last line has been moved, the buzzer will sound.

IMPORTANT: The move command moves the lines exactly as they are stored. Therefore, all line references must be corrected by the user. Also, it is possible to move lines over existing BASIC lines. Therefore, be careful when specifing the destination address (unless it is intended to overwrite the lines).

Note: It is not possible to specify a destination line number that falls between the start and end line numbers. An error will be given if this happens.

MONITOR

>*

Enter the machine language monitor. The >* command exits the ATARI BASIC program and turns control of the 6502 microprocessor over to the monitor. See machine language monitor commands.

MACHINE LANGUAGE MONITOR (MLM) COMMANDS

The MLM provides 16 commands which are most useful to the machine language programmer. It provides the user with the capability to easily interact with the 6502 microprocessor and system memory. The MLM uses the ATARI screen editing capability. This feature makes the MLM powerful and easy to use.

The following is a list of the MLM commands. Carefully read over the commands and examples. Then practice with the MLM to gain a better insight into their use.

. -- COMMAND PROMPT

A period is used to indicate the MLM is ready for a command.

M XXXX YYYY -- DISPLAY MEMORY

Display memory starting at hex address XXXX and ending at YYYY.

Example -

```
M 6531 653F
 :6531 01 02 03 04 05 06 07 08
 :6539 09 0A 0B 0C 0D 0E 0F 10
```

Note -- If only start address is entered, 24 memory locations will be displayed.

Note -- For long memory displays, the control-1 key can be use to stop and start the listing.

Note -- To abort a long listing, press the space bar.

I XXXX YYYY -- INTERROGATE MEMORY

Interrogate memory starting at hex address XXXX and ending at YYYY. The interrogate command works just like the .M command except it also displays the ATASCII equivalent of the memory contents. All cursor control codes are displayed with a question mark (?).

R -- DISPLAY REGISTERS

Display 6502 registers.

Example Printout -

```
 *  PC  AR XR YR PR SP
 ; 7013 41 11 FA 03 FA
```

PC = program counter; AR = accumulator; XR = X register
YR = Y register; PR = status register; SP = stack pointer

: -- ALTER MEMORY

Indicates that the following hex address and line of hex data will be used to alter memory. Cursur up and over to location and change bytes -- press RETURN.

; -- ALTER 6502 REGISTERS

Used to modify 6502 registers. Cursor up and over to register and change bytes (S) -- press RETURN.

G XXXX -- GOTO

GOTO address specified by XXXX and execute program. Program must contain a BRK instruction to return to MLM. If XXXX is not given, the GOTO address defaults to the program counter.

X -- EXIT

Exit the monitor and return to BASIC.

S XXXX YYYY -- SAVE MEMORY TO CASSETTE

Save memory starting at hex address XXXX to ending address YYYY.

Note -- The end address must be the actual address+1.

L XXXX -- LOAD MEMORY FROM CASSETTE

Load memory from cassette and store starting at hex address XXXX. At the end of the load, the screen will display END ADDRESS = followed by the hex address of the end of the program. Binary data saved using the S XXXX YYYY command can be loaded into memory at any location (as defined by L XXXX).

? -- ERROR

A question mark will be printed if a bad command or bad hex data is entered. It will also be given if any command tries to alter a ROM or non-existent memory location.

H XXXX YYYY 'ZZZZZ -- HUNT FOR ASCII STRING

Hunt memory from XXXX to YYYY for the ASCII string ZZZZZ

Example - H 1700 2A80 'ATARI COMPUTER -- Hunt memory from $1700 to $2A80 for the ASCII string ATARI COMPUTER .

Note - The ASCII string can be up to 20 characters long.

Note - If a match of the ASCII string is found, the hex address will be listed to the screen. If no match is found, only the command prompt (.) will be displayed.

H XXXX YYYY ZZ ZZ ZZ ZZ -- HUNT FOR HEX CHARACTERS

Hunt memory from XXXX to YYYY for the hex characters ZZ ZZ ZZ ZZ.

Example - H 1700 2A80 20 00 07 A9 FF -- Hunt memory from $1700 to $2A80 for the hex characters 20 00 07 A9 FF .

Note - The hex characters can be up to 20 hex bytes long.

Note - If a match of the hex characters is found, the hex address will be listed to the screen. If no match is found, only the command prompt (.) will be displayed.

B XXXX YYYY - CALCULATE BRANCH

Calculate the branch value from address XXXX to YYYY.

Example - B 4000 4013 - Calculate the value of a branch instruction when the program counter is at $4000 and branch to instruction is at $4013. In this case, the hex value 13 will be displayed.

A - ATARI DOS

Exit the monitor and go directly to the DOS menu. If DOS isn't loaded, it will enter the MEMO PAD mode.

D XXXX - DISASSEMBLE MEMORY

Disassemble memory starting at hex address XXXX.

Example - D A000 - Disassemble memory starting at $A000. The screen will clear and display the hex code as well as the disassembled mnemonics. The control-1 key is used to stop and start the listing. To terminate the listing, press the space bar.

```
,A000  A5  CA    -LDA $CA
,A002  D0  04    -BNE $A008
,A004  A5  08    -LDA $08
,A006  D0  45    -BNE $A04D
 etc
```

Note - When an unimplimented opcode is encountered, the mnemonic field will display ???.

, - ALTER DISASSEMBLE LISTING

A comma command is used to alter the hex code printed out by the disassemble command. After the listing has been stopped with space bar, simply cursor up and over and change hex code (up to 3 memory locations can be modified). When the RETURN key is pressed, the disassembly process will begin again.

PS - PRINTER SET
PC - PRINTER CLEAR

The printer set command allows machine language monitor information set to the screen to be also sent to the printer. The printer set command remains activated until a printer clear command is given.

ADDITIONAL NOTES

(1) Spaces have been shown in the examples of the MLM commands. Spaces are optional and are not required by the monitor.

(2) In addition to the MLM commands, five of the normal Monkey Wrench commands can be used. They are: >E, >U, >#, >$, >>x.

MAKING BACKUP TAPES OF CASSETTE PROGRAMS USING THE MLM

The Machine Language Monitor (MLM) load and save (binary data) commands can be used to make backup copies of cassette tape programs. This includes cassette tapes in boot format. The backup tape process is simple and requires no special knowledge by the user. Follow the simple steps shown below when making a backup tape:

(1) Bootup the computer with the ATARI BASIC and Monkey Wrench cartridges installed.
(2) Type >* , to enter the MLM mode.
(3) Insert the cassette tape to be copied into the cassette recorder and depress the PLAY button.
(4) To load the program, type L 0700 and depress RETURN key. The buzzer will beep. Depress RETURN again to start the load process. The sound of the data being loaded can be heard over the TV speaker.
(5) At the end of the load, the screen will display END ADDRESS = followed by end hex address of the program. (Note: Although some users will not understand hex numbering, it is only important here to continue following the step procedure.)
(6) The program is now loaded. Remove the old cassette tape and install a blank cassette into the recorder. Press PLAY and RECORD buttons on the cassette recorder.
(7) To save the program, type S 0700 followed by 'end address' displayed on the screen at the end of the load. (See step 5). Be sure to type the end address exactly as shown on the screen and depress RETURN key. The buzzer will beep twice. Depress RETURN again to start the save process. The sound of the data being saved can be heard over the TV speaker.
(8) When the sound stops, the backup tape process is completed. An example of the backup tape process, as seen on the screen, is shown below.

```
>*

*  PC  AR XR YR PR SP
; 8B29 27 FF 03 31 FF
.L 0700
END ADDRESS = 1AF6

.S 0700 1AF6
.
```

Note - Some commercial type software use special recording formats to prevent backups. Therefore, these types of programs cannot be backed-up with the above steps.

ABOUT THE CARTRIDGE

As stated previousily, the Monkey Wrench ROM is contained on a special printed circuit cartridge. This cartridge is configured such that it uses memory from hex 8000 to hex 9fff when inserted into the right slot. Therefore, on an ATARI with 48k memory, only 32k of memory is usable. This is ,of course, a constraint of the ATARI computer.

WARRANTY INFORMATION

The Monkey Wrench cartridge (hardware) is warranted against defects in material and workmanship for a period of 60 days from the date of purchase. If a defect is discovered during the 60 day period, and you have registered this warranty, Eastern House will replace or repair the cartridge - provided the cartridge and proof of purchase is mailed (postage prepaid) to Eastern House.

If the defect (in the judgement of Eastern House) resulted from accident, abuse, or misapplication of the cartridge, Eastern House shall have no responsibility to replace or repair the cartridge under the terms of this warranty. After the 60 day warranty, Eastern House will repair the cartridge for the cost of parts and shipping.

If you have any questions or comments about this software (good or bad), please write or call. Eastern House has always felt it is important not to abandon the user once the sale has been made. Please feel you can contact us if necessary.