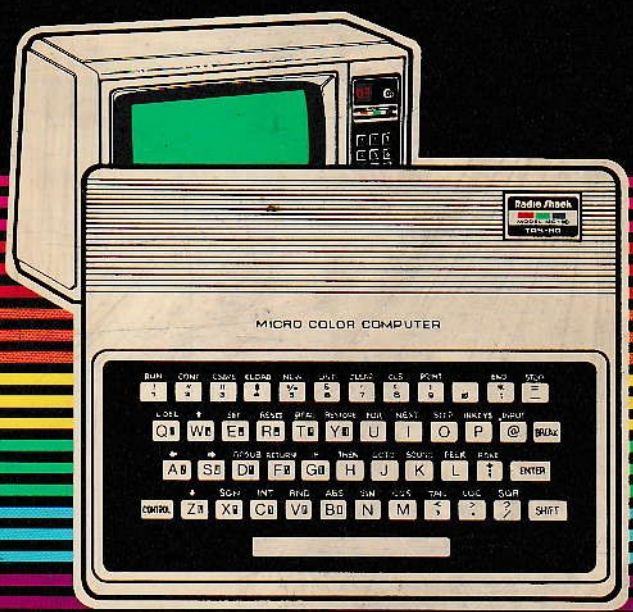
Radio Shack®

TRS-80® MC-10
Micro Color Computer
Operation and Language Reference Manual



CUSTOM MANUFACTURED BY RADIO SHACK, A DIVISION OF TANDY CORPORATION

TERMS AND CONDITIONS OF SALE AND LICENSE OF RADIO SHACK COMPUTER EQUIPMENT AND SOFTWARE PURCHASED FROM A RADIO SHACK COMPANY-OWNED COMPUTER CENTER RETAIL STORE OR FROM A RADIO SHACK FRANCHISEE OR DEALER AT ITS AUTHORIZED LOCATION

LIMITED WARRANTY

I. CUSTOMER OBLIGATIONS

A. CUSTOMER assumes full responsibility that this Radio Shack computer hardware purchased (the "Equipment"), and any copies of Radio Shack software included with the Equipment or licensed separately (the "Software") meets the specifications, capacity, capabilities, versatility, and other requirements of CUSTOMER.

B. CUSTOMER assumes full responsibility for the condition and effectiveness of the operating environment in which the Equipment and Software are to function, and for its installation.

II. RADIO SHACK LIMITED WARRANTIES AND CONDITIONS OF SALE

A. For a period of ninety (90) calendar days from the date of the Radio Shack sales document received upon purchase of the Equipment, RADIO SHACK warrants to the original CUSTOMER that the Equipment and the medium upon which the Software is stored is free from manufacturing defects. THIS WARRANTY IS ONLY APPLICABLE TO PURCHASES OF RADIO SHACK EQUIPMENT BY THE ORIGINAL CUSTOMER FROM RADIO SHACK COMPANY-OWNED COMPUTER CENTERS, RETAIL STORES AND FROM RADIO SHACK FRANCHISEES AND DEALERS AT ITS AUTHORIZED LOCATION. The warranty is void if the Equipment's case or cabinet has been opened, or if the Equipment or Software has been subjected to improper or abnormal use. If a manufacturing defect is discovered during the stated warranty period, the defective Equipment must be returned to a Radio Shack Computer Center, a Radio Shack retail store, participating Radio Shack franchisee or Radio Shack dealer for repair, along with a copy of the sales document or lease agreement. The original CUSTOMER'S sole and exclusive remedy in the event of a defect is limited to the correction of the defect by repair, replacement, or refund of the purchase price, at RADIO SHACK'S election and sole expense. RADIO SHACK has no obligation to replace or repair expendable items.

B. RADIO SHACK makes no warranty as to the design, capability, capacity, or suitability for use of the Software, except as provided in this paragraph. Software is licensed on an "AS IS" basis, without warranty. The original CUSTOMER'S exclusive remedy, in the event of a Software manufacturing defect, is its repair or replacement within thirty (30) calendar days of the date of the Radio Shack sales document received upon license of the Software. The defective Software shall be returned to a Radio Shack Computer Center, a Radio Shack retail store, participating Radio Shack franchisee or Radio Shack dealer along with the sales document.

C. Except as provided herein no employee, agent, franchisee, dealer or other person is authorized to give any warranties of any nature on behalf of RADIO SHACK.

D. Except as provided herein, **RADIO SHACK MAKES NO WARRANTIES, INCLUDING WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.**

E. Some states do not allow limitations on how long an implied warranty lasts, so the above limitation(s) may not apply to CUSTOMER.

III. LIMITATION OF LIABILITY

A. EXCEPT AS PROVIDED HEREIN, RADIO SHACK SHALL HAVE NO LIABILITY OR RESPONSIBILITY TO CUSTOMER OR ANY OTHER PERSON OR ENTITY WITH RESPECT TO ANY LIABILITY, LOSS OR DAMAGE CAUSED OR ALLEGED TO BE CAUSED DIRECTLY OR INDIRECTLY BY "EQUIPMENT" OR "SOFTWARE" SOLD, LEASED, LICENSED OR FURNISHED BY RADIO SHACK, INCLUDING, BUT NOT LIMITED TO, ANY INTERRUPTION OF SERVICE, LOSS OF BUSINESS OR ANTICIPATORY PROFITS OR CONSEQUENTIAL DAMAGES RESULTING FROM THE USE OR OPERATION OF THE "EQUIPMENT" OR "SOFTWARE." IN NO EVENT SHALL RADIO SHACK BE LIABLE FOR LOSS OF PROFITS, OR ANY INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF ANY BREACH OF THIS WARRANTY OR IN ANY MANNER ARISING OUT OF OR CONNECTED WITH THE SALE, LEASE, LICENSE, USE OR ANTICIPATED USE OF THE "EQUIPMENT" OR "SOFTWARE."

   NOTWITHSTANDING THE ABOVE LIMITATIONS AND WARRANTIES, RADIO SHACK'S LIABILITY HEREUNDER FOR DAMAGES INCURRED BY CUSTOMER OR OTHERS SHALL NOT EXCEED THE AMOUNT PAID BY CUSTOMER FOR THE PARTICULAR "EQUIPMENT" OR "SOFTWARE" INVOLVED.

B. RADIO SHACK shall not be liable for any damages caused by delay in delivering or furnishing Equipment and/or Software.

C. No action arising out of any claimed breach of this Warranty or transactions under this Warranty may be brought more than two (2) years after the cause of action has accrued or more than four (4) years after the date of the Radio Shack sales document for the Equipment or Software, whichever first occurs.

D. Some states do not allow the limitation or exclusion of incidental or consequential damages, so the above limitation(s) or exclusion(s) may not apply to CUSTOMER.

IV. RADIO SHACK SOFTWARE LICENSE

RADIO SHACK grants to CUSTOMER a non-exclusive, paid-up license to use the RADIO SHACK Software on **one** computer, subject to the following provisions:

A. Except as otherwise provided in this Software License, applicable copyright laws shall apply to the Software.

B. Title to the medium on which the Software is recorded (cassette and/or diskette) or stored (ROM) is transferred to CUSTOMER, but not title to the Software.

C. CUSTOMER may use Software on one host computer and access that Software through one or more terminals if the Software permits this function.

D. CUSTOMER shall not use, make, manufacture, or reproduce copies of Software except for use on **one** computer and as is specifically provided in this Software License. Customer is expressly prohibited from disassembling the Software.

E. CUSTOMER is permitted to make additional copies of the Software **only** for backup or archival purposes or if additional copies are required in the operation of **one** computer with the Software, but only to the extent the Software allows a backup copy to be made. However, for TRSDOS Software, CUSTOMER is permitted to make a limited number of additional copies for CUSTOMER'S own use.

F. CUSTOMER may resell or distribute unmodified copies of the Software provided CUSTOMER has purchased one copy of the Software for each one sold or distributed. The provisions of this Software License shall also be applicable to third parties receiving copies of the Software from CUSTOMER.

G. All copyright notices shall be retained on all copies of the Software.

V. APPLICABILITY OF WARRANTY

A. The terms and conditions of this Warranty are applicable as between RADIO SHACK and CUSTOMER to either a sale of the Equipment and/or Software License to CUSTOMER or to a transaction whereby RADIO SHACK sells or conveys such Equipment to a third party for lease to CUSTOMER.

B. The limitations of liability and Warranty provisions herein shall inure to the benefit of RADIO SHACK, the author, owner and/or licensor of the Software and any manufacturer of the Equipment sold by RADIO SHACK.

VI. STATE LAW RIGHTS

The warranties granted herein give the **original** CUSTOMER specific legal rights, and the **original** CUSTOMER may have other rights which vary from state to state.

The FCC Wants You To Know . . .

This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC Rules. Only peripherals (computer input/output devices, terminals, printers, etc.) certified to comply with the Class B limits may be attached to this computer. Operation with non-certified peripherals is likely to result in interference to radio and TV reception.

To Our Customers . . .

Congratulations for selecting the TRS-80® MC-10 microcomputer — the latest addition to our Color Computer family. The MC-10 is a new, compact computer that easily attaches to your Color T.V. set, producing full-color displays. Its size and ease of use make it the ideal first computer or an excellent experimental unit if you've been around computers for awhile.

The MC-10 uses a powerful version of TRS-80 Color BASIC we call Micro Color BASIC which includes commands that:

- Create low resolution graphics.
- Allow you to "save" programs on cassette tapes, then "load" them back into the Computer at a later date.
- Print out graphic displays, text, or data on a serial printer for "hard-copies."

and more!

About This Manual . . .

This manual was written with several purposes in mind:

- To show you how to set up and operate the MC-10.
- To describe Micro Color BASIC, the programming language the MC-10 uses.
- To teach you how to program using the MC-10 and Micro Color BASIC.

If this is your first encounter with computers, we suggest you read this entire manual. By the time you finish it, you'll be able to write your own Micro Color BASIC programs.

If you're familiar with computers and the BASIC programming language, be sure to read the **"Operation"** section of this manual, then turn to Appendix F — the **"MC-10 Micro Color BASIC Language Reference Summary."**

And don't forget! Micro Color BASIC contains most of the features (plus a few additional ones!) of regular TRS-80 Color BASIC. This means that most of the books (and programs that require less than 3K of memory) for Color BASIC will apply to the MC-10 as well.

So sit back and enjoy your MC-10. It won't take you long to find out that the MC-10 is one of the easiest and most enjoyable ways to learn about computers!

Contents

Part I/ Operation

1/ Meet the MC-10! 1
Keyboard (Top View) 2
Power Switch (Side View) 3
Channel Selector (Bottom View) 3
Connection Panel (Rear View) 4

2/ Setting Up the MC-10 5
Connection to a TV 5
Connection to a Cassette Recorder 9
Connection to a Printer 10
Connection to a Power Source 11

3/ Operating the MC-10 13
Turning the Power On 13
Pressing the RESET Button 14
Turning the Power Off 14

4/ Using the MC-10 15
The MEM Function 15
The MC-10 Keyboard 16
The (SHIFT) Key 16
The (CONTROL) Key 18
The MC-10 and a TV 19
The MC-10 and a Cassette Recorder 21
Volume Control 21
Saving a Program on Tape (CSAVE) 21
Loading a Program from Tape (CLOAD and SKIPF) 22
The MC-10 and a Printer 24
Printers You Can Use with the MC-10 24
Listing a Program (LLIST) 25
Printing Data (LPRINT) 25

Part II/ Learning and Using Micro Color BASIC

5/ Getting Started 27
The OK Prompt 27
The (ENTER) Key 27

Strings vs. Numbers 28
The MC-10 as a Calculator 29
Error Messages 29

6/ Re-Usable Instructions (Programs) 31
Immediate vs. Program Mode 31
Line Numbers 31
NEW 32
Writing Your First Program 33
RUN 33
LIST 34
LLIST 36

7/ Data Output to the TV and Printer 37
PRINT 37
PRINT @ 41
PRINT TAB 41
LPRINT 43

8/ Variables and Constants 45
Numeric Variables 45
String Variables 47
LET 48

9/ Stop, Look, and Listen 49
END 49
STOP 49
The BREAK Key 50
CONT 51
CLS 52
SOUND 53
REM 54

10/ Interactive Programming and the MC-10 55
INPUT 55
GOTO 58
ON...GOTO 60
GOSUB/RETURN 61
ON..GOSUB/RETURN 63

11/ Operators and Controlled Looping 65
IF/THEN 65
AND/OR/NOT 67
FOR/TO/NEXT...STEP 68

12/ Reading Data 73
DATA 73
READ 73
RESTORE 74

13/ Arrays 77
DIM 80
CSAVE* 82
CLOAD* 82
RND 83

14/ String Functions 85
INKEY$ 85
ASC 87
CHR$ 88
STR$ 89
LEN 89
LEFT$ 90
MID$ 90
RIGHT$ 90
CLEAR 91

15/ Graphics 93
SET 94
RESET 95
POINT 96

16/ Math, Trig, and Other Functions 99
↑ 99
SQR 100
ABS 100
SGN 101
INT 101
SIN 103
COS 104
TAN 105
LOG 106
VAL 107
EXP 107

Part III/ Appendix

Appendix A/ Sample Programs 109

Appendix B/ Error Messages 113

Appendix C/ Codes 115

Appendix D/ Troubleshooting and Maintenance 119

Appendix E/ Specifications 125

Appendix F/ Micro Color BASIC Language Reference Summary 129

Part I
Operation

1/ MEET THE MC-10!

Carefully unpack the Computer. Remove all packing material and save it just in case you ever need to transport the MC-10. Locate all cables and papers that may be included in the shipping carton.

When you take your new MC-10 out of the box, be sure the package includes:

- The MC-10 itself.
- An Antenna Switch that connects the Computer to the TV.
- A Connecting Cable that connects your MC-10 to the TV.
- A Power Supply that connects your MC-10 to a wall-outlet. (Use ***only*** the Power Supply included with this package!)
- This manual.
- Quick Reference Card.

Before connecting the MC-10 to a TV or a power source (or before doing anything else for that matter), be sure you're familiar with your MC-10. Remember that proper operation of the Computer depends on proper connection to the TV, to power sources, and to optional equipment (such as a printer or cassette recorder).

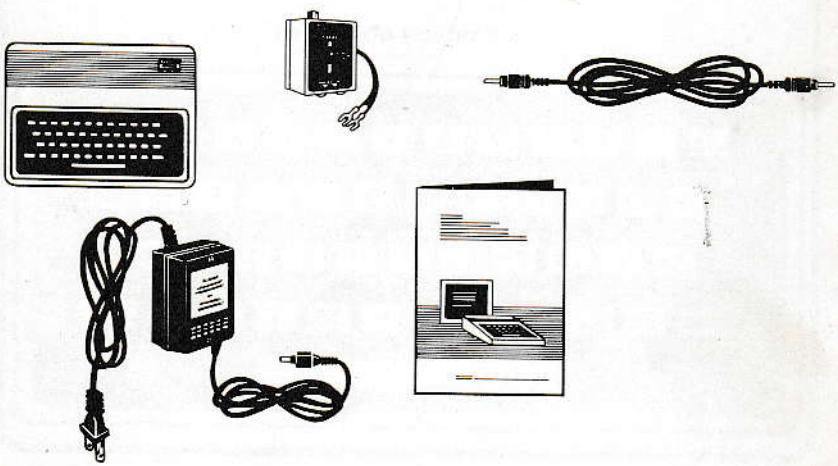


Figure 1. The MC-10 Package

Read the following section carefully so you are completely familiar with the MC-10 before going ahead.

① **Command Keys** The Micro Color BASIC keywords (and other special symbols) listed above the regular keys can be accessed by pressing (CONTROL) and the key just below the statement you wish to use.

② **Regular Keys** For normal operation, simply press these keys. **Note:** You will not have to press (SHIFT) to produce uppercase letters. The MC-10 is normally in "All-Caps" Mode.

③ **Graphics Character Keys** A few keys, such as this one, have alternate definitions that allow you to display Block Graphics Characters on the Screen. To produce these characters, press (SHIFT) and one of the keys that has a Graphics Character on it.

④ (CONTROL) This is the "control" key which lets you perform special operations when it is pressed in conjunction with other keys. For instance,

Figure 2. MC-10 Keyboard (Top View)

you can use the predefined Micro Color BASIC commands, backspace the Cursor, delete the current program line, and more.

⑤ (SPACEBAR) When you want to move the Cursor to the right, press this key.

⑥ (SHIFT) This key allows you to display the alternate definition some keys have. This includes punctuation symbols and Graphics Characters. You will not have to press (SHIFT) to produce uppercase letters. The MC-10 is normally in "All-Caps" Mode. If you enter Upper/Lowercase Option Mode (by pressing (SHIFT) (0)), lowercase letters will be displayed in reverse video and you will have to press (SHIFT) to produce uppercase letters.

⑦ (ENTER) To "enter" a program line or execute a command, press this key. It is essentially the same as the carriage return key on a standard typewriter.

⑧ (BREAK) To stop (or "break") program execution and return to the OK prompt, press this key.

Once the MC-10 is properly connected to a power source, the TV, and any optional equipment, set this switch to ON.

Be sure the setting of this switch matches the channel you've selected on the TV. Which channel (either 3 or 4) should you use? That depends on the channel that is either weaker or not used in your area. Try both, then use the channel which produces the best quality display.

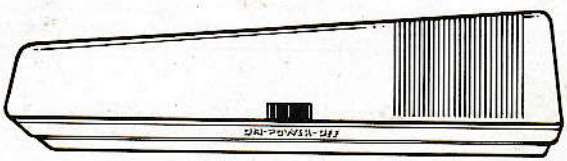


Figure 3. MC-10 Power Switch (Side View)

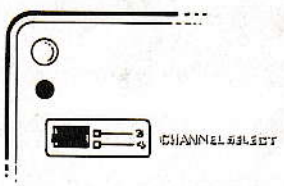


Figure 4. MC-10 Channel Selector (Bottom View)

① **AC ADAPTER** Connect the Power Supply included with this package to this connector.

② **TO TV** Connect one end of the Connecting Cable included with this package to this connector. Connect the other end of the Cable to the Antenna Switchbox which should be connected to your TV. (Both ends of the cable are the same.)

③ **Memory Expansion Connector** This slot is reserved for future memory expansion kits.

④ **RESET Button** If the Computer ever "hangs-up" or if you ever want to start over, press this Button.

⑤ **RS-232C Connector** When connecting the MC-10 to a serial printer, use this connector.

⑥ **CASSETTE Connector** To store and retrieve programs or data, connect the cassette recorder to this connector.

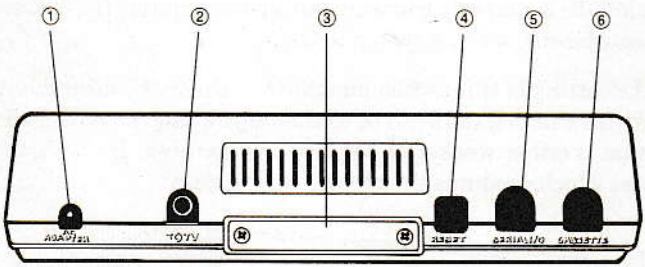


Figure 5. MC-10 Connection Panel (Rear View)

2/ SETTING UP THE MC-10

Proper connection of the MC-10 to a TV, a power source, and various optional equipment is important if your Computer is to provide you with reliable service.

Place the Computer on a solid surface near the TV set you'll be using. An appropriate power source should be nearby. We do not recommend using a regular extension cord if a power source is not close enough for direct connection. Instead, try using Radio Shack's ***Plug-In Power Strip*** (Radio Shack Catalog Number 61-2619) or the ***Power Line Filter*** (26-1451).

Do not connect the Computer to the AC power source yet.

Connection to a TV

The MC-10 contains its own, built-in television interface which converts information the computer understands to information the television understands. That information is then sent as a VHF signal to your TV.

Note that the MC-10 comes with an Antenna Switchbox. This Switchbox enables you to use your TV for normal viewing or for computer operation without having to connect/disconnect the computer everytime you use it. Set the Switchbox's slide-switch to:

- **TV**, for normal television viewing.
- **COMPUTER**, for computer input.

Mounting the Antenna Switchbox

The Switchbox consists of a:

- Box
- Short section of twin-lead cable (like the cable which connects an antenna to your TV)
- Two screw terminals
- Coaxial connector
- Slide switch

We suggest you attach the Switchbox to the back of your TV. See Figure 6.

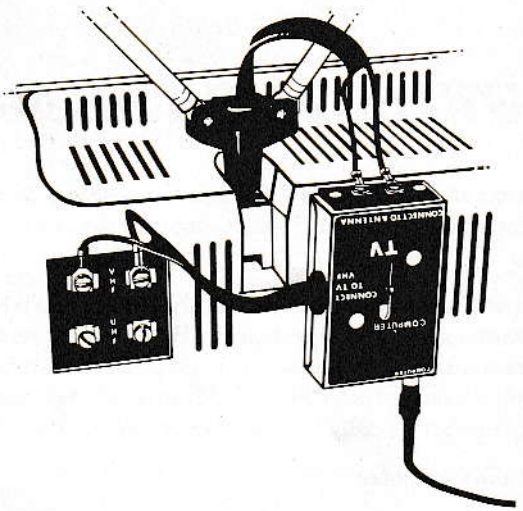


Figure 6. Mounting Antenna Switchbox onto Your TV

To mount the Switchbox, follow these steps:

1. Select a smooth, flat surface on the TV that is well within reach of the antenna cables. Wipe off any dust, dirt, or grease from the mounting surface.
2. Remove the backing from the double-sided tape (one side is already attached to the Switchbox) to expose the sticky surface.
3. Press it against the back of the TV in the desired location.

Connecting the Antenna Cable to the Switchbox

You need to connect the VHF antenna cables (that come from an external antenna or the TV's internal antenna) to the Switchbox. If these wires are presently connected to the VHF terminals on your TV set, disconnect them now. (Be sure to use a small screwdriver.)

What you do next depends on the type of antenna installation you have. Read the following section and pay careful attention to Figures 7 through 12 to decide which installation you have.

Internal Antenna and Screw Leads Connect the internal antenna cables from the TV to the terminals on the Switchbox labeled **CONNECT**

TO ANTENNA. Then connect the short twin-lead cable from the Antenna Switchbox to your TV's VHF screw terminals. See Figure 7.

External Antenna and Twin-Leads Connect the cable from the external antenna (including Rabbit Ears) to the terminals on the Antenna Switchbox labeled **CONNECT TO ANTENNA**. Connect the short twin-lead from the Antenna Switchbox to your TV's VHF screw terminals. See Figure 8.

75-Ohm Coaxial Lead-In To make this connection, you will need to obtain a special 75-ohm to 300-ohm matching transformer. We suggest Radio Shack's ***Indoor/Outdoor Matching Transformer*** (15-1140). Con-

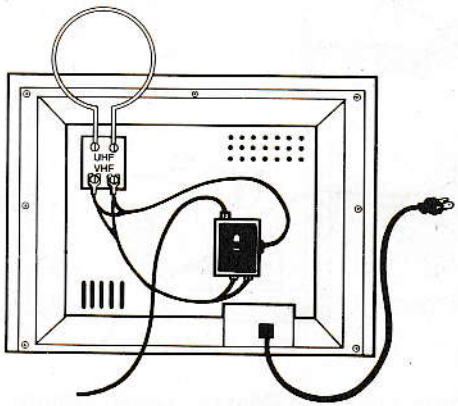


Figure 7. Internal Antenna and Screw Lead Connection

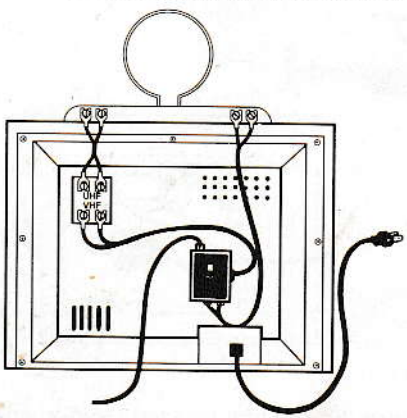


Figure 8. External Antenna and Twin-Leads Connection

nect your coaxial cable lead-in to the transformer and connect the transformer's twin-lead to the screw terminals on the Antenna Switchbox.

Move the TV's slide switch to the CLOSED position.

Connect the short twin-lead from the Antenna Switchbox to your TV's VHF screw terminals. See Figure 9.

75-Ohm Coaxial Lead-In and Matching Transformer Connect the short twin-leads from the transformers to the screw terminals on the Antenna Switchbox. Connect the short twin-lead from the Antenna Switchbox to your TV's VHF screw terminals. See Figure 10.

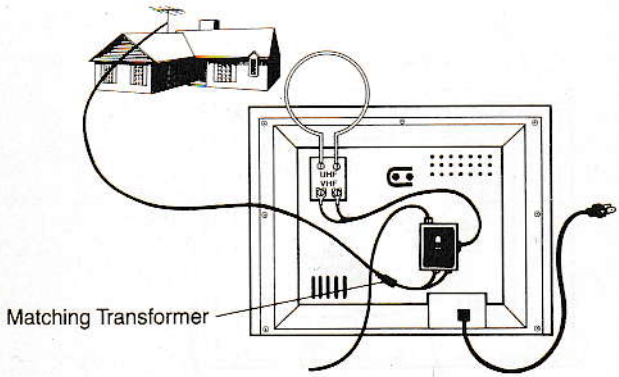


Figure 9. 75-Ohm Coaxial Lead-In Connection

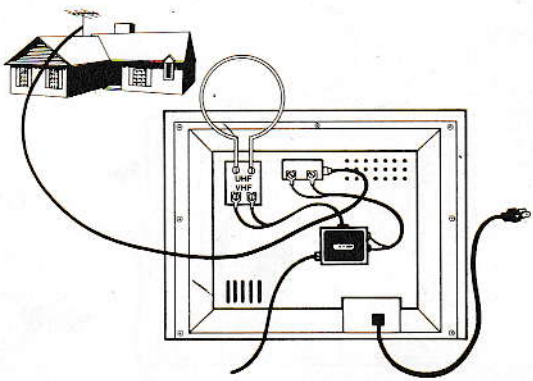


Figure 10. 75-Ohm Coaxial Lead-In and Matching Transformer Connection

In this section, we have shown five typical TV antenna installations. There are some other antenna types and features you may need to be aware of, however.

- If your TV has a 75- 300-ohm switch on the back, be sure the switch is set to the 300-ohm position.
- If your TV has a round jumper cable protruding from the back, be sure the jumper cable has been attached to the connector next to it.
- If your TV's antenna system is one we've just described, there should be no problem at all. The connections between the other TV antenna terminals and the Antenna Switchbox will be as in one of the illustrations.
- If you have cable TV, we strongly recommend that you call in a qualified service technician.

Connecting the MC-10 to the Switchbox

After connecting your TV's antenna to the Antenna Switchbox, connect the Computer's coaxial cable (supplied with the MC-10 package) to the Computer's TO TV connector on one end and the Antenna Switchbox's **COMPUTER** connector on the other. Note that both ends of the cable are the same. See Figure 11.

Your MC-10 Computer is now ready for use.

Connection to a Cassette Recorder

To save programs or data for future use, you'll want to store that information on cassette tape. We suggest you use Radio Shack's ***CCR-81 Com-***

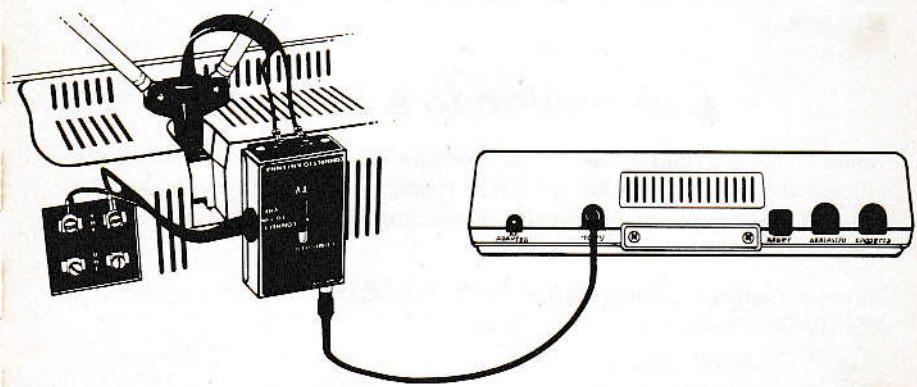


Figure 11. Computer to Switchbox Connection

puter Recorder (26-1208) and our ***C-20 Leaderless Computer Cassette Tapes*** (26-301). If you use a different recorder, connection and operation may vary.

Note: You do not need to connect the Cassette Recorder unless you plan to record programs, save data, or load previously taped programs into the MC-10.

The CCR-81 cassette recorder includes a cable specifically designed for computer's such as the MC-10. We recommend you use this cable only.

1. Connect the short cable (DIN plug on one end and three plugs on the other) to the **CASSETTE** connector on the back of the Computer. Be sure you get the plug to mate correctly.

   The three plugs on the other end of the cable are for connection to the recorder.

2. Connect the black plug into the **EAR** jack on the side of the recorder.

   This connection provides the output signal from the recorder to the MC-10 (for loading tape programs into the Computer).

3. Connect the larger gray plug into the **AUX** jack on the recorder. This connection provides the recording signal to record programs from the Computer onto the tape.

   Leave the **AUX** plug in whether you are recording or playing back cassette data.

4. Connect the smaller gray plug into the smaller MIC jack on the recorder.

Note: Do not plug a remote microphone or a dummy plug into the larger MIC jack.

Connection to a Printer

Radio Shack provides a thermal printer, the **TP-10** (26-1261), which is specifically designed for the MC-10. This printer provides a 32-column print out on 4" wide paper and is capable of printing all the graphics the MC-10 can produce.

However, you can use any Radio Shack serial printer or plotter with the MC-10. This includes the:

- Line Printer VII (26-1167)
- Line Printer VIII (26-1168)
- DMP-100 (26-1253)

- DMP-120 (26-1255)
- DMP-200 (26-1254)
- DMP-400 (26-1251)
- CGP-115 Plotter/Printer (26-1192)
- FP-215 Flatbed Plotter (26-1193)

To use the MC-10 with any of these printers or plotters, you'll need to use the proper connecting cable. To connect the MC-10 to any of these optional devices, you'll need the Color Computer Serial Interface Cable (see Figure 12).

Before connecting any accessory device (such as a serial printer) to the MC-10, be sure that both the Computer and the accessory are both turned OFF.

To connect a printer to the MC-10:

1. Insert one end of the Color Computer Serial Cable into the RS-232C Connector on the Computer.
2. Connect the other end of the Cable to the **SERIAL I/O CONNECTOR** (4-pin DIN jack) on the Connection Panel of the printer or plotter.

Note that the MC-10 has full, RS-232C communications capabilities. This means that the RS-232C Connector can be used with any RS-232C compatible device, including modems. A modem will allow you to transmit information over the telephone to other computers via the RS-232C Connector as long as you have communications software.

Connection to a Power Source

Before connecting the MC-10 to a power source, be sure all accessories (such as a printer or recorder) are turned OFF!

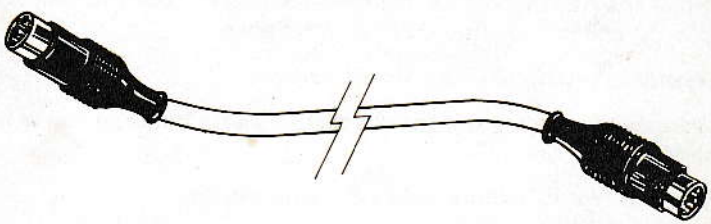


Figure 12. Color Computer Serial Interface Cable (26-3020)

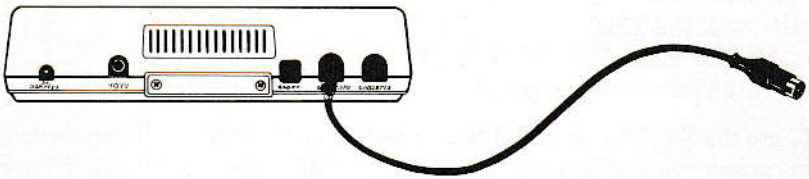


Figure 13. Connecting a Printer to the MC-10 (RS-232C)

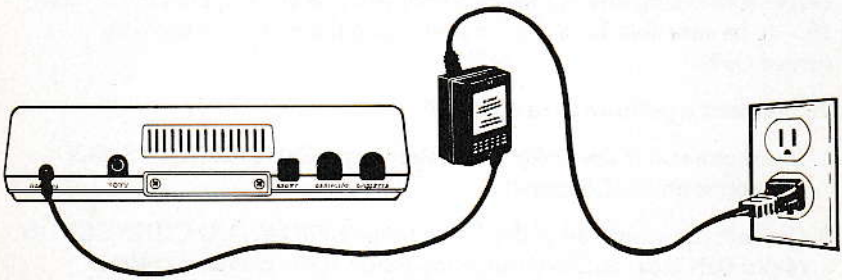


Figure 14. Connecting the MC-10 to an AC Power Source

The MC-10 comes with an AC Adapter. ***Do not use any other AC Adapter or you may damage your Computer!***

Try to have the MC-10 close to a 120V AC wall-outlet. If it is not close enough, we suggest that you use an approved power strip, such as the Radio Shack ***Plug-In Power Strip*** (61-2619) or the ***Power Line Filter*** (26-1451) instead of a lightweight extension cord.

Power requirements for Radio Shack products are specified on the unit or in the **"Specifications"** section of the appropriate owner's manual.

Note that the AC Adapter has two cords exiting it—one that has a round plug on the end and another that has a flat plug.

To connect the MC-10 to a power source:

1. Insert the round plug into the **AC ADAPTER** Connector on the back of the MC-10.
2. Insert the flat AC plug into the AC power source.

3/ OPERATING THE MC-10

Only after all connections (TV, recorder, printer, and power source) have been made should you power up the Computer.

Never turn any accessories on or off while the Computer is in use. Doing so can cause abnormal operation of the Computer or cause it to "hang up." If this occurs, you may have to press the RESET Button (located on the back of the MC-10) or turn the system off and on again. Be aware that when you turn the Computer off, all data currently in memory will be erased.

Turning the Power On

At this stage, the Computer and all accessories must be OFF!

1. Turn on the TV and adjust the volume to a normal listening level.
2. On the TV, select either channel **3** or **4** (whichever is weaker or not used in your area).
3. On the MC-10, select the same channel by positioning the **CHANNEL SELECT** Switch (on the bottom of the Computer) to the appropriate setting.
4. Set the Antenna Switchbox to the **COMPUTER** position.
5. Turn the MC-10 Computer ON.

   The Power On/Off Switch is on the right side of the Computer.
6. Turn any accessory equipment (printer, cassette recorder, etc.) ON.

Your TV Screen should then turn green and display the following message:

```
MICROCOLOR BASIC v.r.
COPYRIGHT 1982 MICROSOFT
OK
```

where *v.r.* is a pair of numbers specifying which version and release you have.

If the message does not appear:

• Be sure your TV is turned on and is operational.

- Check your TV's Brightness, Contrast, and fine tuning adjustments.
- If the message still doesn't appear, turn the entire system OFF, recheck all connections and try again. For further assistance, see **"Troubleshooting and Maintenance"** later in this manual.

Pressing the RESET Button

If the system ever "hangs up" or you ever want to start over, press the red RESET Button on the back of the MC-10. The Screen will clear and the OK prompt should return.

Turning the Power Off

Always turn the Computer off first—then turn accessories off.

If you turn the Computer off for any reason, leave it off for at least 15 seconds before turning it back on again. The Computer's power supply needs this much time to discharge its stored energy before starting up again.

Whenever you turn the Computer off, all programs and data in memory are erased. If you have data or programs you want to use again, save that information before turning the Computer off. See **"The MC-10 and a Cassette Recorder"** later in this manual for details on saving and re-loading information.

4/ USING THE MC-10

The MC-10 (or any computer for that matter) really has only four functions. It can:

- Allow information to be put into the computer ("input").
- Process information that is currently in the computer ("processing").
- Provide you with the results of the processed information ("output").
- Offer the capability to store information ("memory").

Depending on the computer, there are different ways to accomplish these feats. This section of the manual will describe, from an operational point of view, how you input and output data using the MC-10. Study this section carefully now and you'll enjoy using your MC-10 more later on.

Input The primary method of data input the MC-10 uses is the Keyboard. The other ways to input information are via the **CASSETTE** Connector from a cassette recorder or via the RS-232C Connector if you have programs that allow you to receive data.

Output The main method of data output the MC-10 uses is through the TV Screen. Data output can also be via a printer (through the RS-232C channel) or a cassette recorder (through the **CASSETTE** Connector).

As long as the power is on, the MC-10 can store over 3,000 "bytes" or characters (numbers, letters, spaces, etc.). When the power is turned off, all information is erased from memory. If you want to store information permanently, use the "auxiliary memory" capabilities of a cassette recorder.

How do you know how much storage space the MC-10 has at any one time? Use the **MEM** function which displays the amount of free memory for your programs. Turn the MC-10 off, wait 15 seconds, and turn it back on again. When the OK prompt appears, type `PRINT MEM` and press (ENTER). The Screen should display 3142. That means you can enter 3,142 numbers, letters, spaces, or carriage returns before the Computer stops accepting data. If you are typing long programs, we suggest you stop every now and then and PRINT MEM to ensure you don't run out of memory.

The rest of this chapter will describe how you can use the MC-10 input and output capabilities.

The MC-10 Keyboard

The MC-10's Keyboard has the convenience of multi-purpose keys. This includes keys that have been predefined as Micro Color BASIC keywords (commands, functions, special symbols, and Graphics Characters) so you don't have to type in the entire word when you want to use it.

In this section, we will not attempt to describe the function of each Micro Color BASIC command or function. We are only describing how you can use the MC-10's features to save some time when typing in long programs. For a complete description of the Micro Color BASIC keywords, see Part II of this manual as well as the Micro Color BASIC Language Reference section in the Appendixes.

For normal use, however, the MC-10 keyboard can be used like an ordinary typewriter or computer keyboard with a few minor exceptions.

The (SHIFT) Key

The (SHIFT) key allows you to access the alternate definition of some keys. Alternately defined keys are those keys that have more than one symbol on the key itself. This includes punctuation (such as !, $, ") and Graphics Characters. (SHIFT) also enables you to access the Upper/Lowercase Option Mode where both uppercase and lowercase letters can be used.

The Normal Operation Mode (on power-up) produces all-capital letters. You can access the alternate key definitions in this mode by pressing (SHIFT) and the key. For example, the (1) key's primary definition is "1" and its alternate definition (accessed by pressing (SHIFT) (1)) is "!." In the same sense, the primary definition of the (A) key is "A" while the Graphics Character ▚ is its alternate definition.

In the Upper/Lowercase Option Mode, you can produce the letters A-Z in reverse video to indicate the characters are lowercase. (See **"Upper and Lowercase Letters"** later in this chapter.) In this mode, you cannot produce Graphics Characters.

Whenever you turn the Computer on, you are automatically in the "All-Capitals" (CAPS) Mode in which the keyboard will always generate the uppercase version of the letters A-Z on the TV, whether or not the (SHIFT) key is pressed.

To switch to Upper/Lowercase Option Mode, press (SHIFT) (0). Then the unshifted A-Z keys will produce lowercase letters. In this mode, you must press (SHIFT) for uppercase letters.

The MC-10 can only generate uppercase (capital) letters on the TV Screen. When a lowercase letter is output to the TV Screen, it is displayed as a capital letter in "reverse" video (i.e., the background is black while the letter is green).

If the TV Screen won't display "real" lowercase letters, why even bother? For one thing, you may want to use the upper/lowercase mode when typing information which will be printed out on the printer. In other words, when you send lowercase letters to the printer, it will print lowercase letters if it has the capabilities.

Note: Micro Color BASIC requires that all commands and functions be uppercase entries.

To switch back to all-capitals, press SHIFT 0 again.

Graphics Characters are the alternate definition characters found on the keys shown in Figure 15. Note that the actual Screen display will be the reverse of the keyboard symbols.

When the MC-10 is in All-Caps Mode, and you press the key-combination of SHIFT and one of the keys which has a Graphics Character on it, the Graphics Character will be displayed on the Screen. For example, SHIFT A will display the Graphic Character ▚. (Note that you can produce similar results by sending a CHR$(137) from Micro Color BASIC. For more information on using Graphics Characters in Micro Color BASIC programs, see Chapter 15 of this manual.)

Even though there are 16 possible design combinations for Graphics Characters, you can display each one of them with a different background color (there are eight background colors) for a total of 128 individual characters.

To change background colors for the Graphics Characters, press the key-combination of CONTROL 0. You can tell what the background color will be

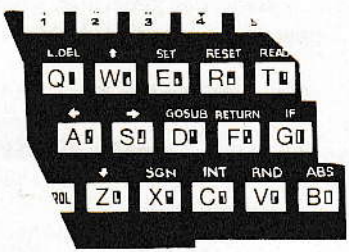


Figure 15. Graphics Character Keys

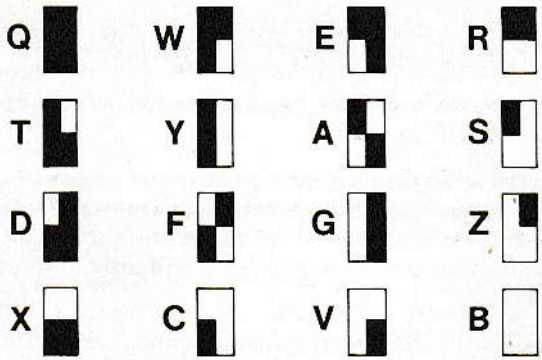


Figure 16. MC-10 Graphics Characters

by watching the color of the Cursor. On power-up, the Cursor is black. Each time you press CONTROL @, the Cursor will change colors until it recycles to the original color.

Note: If you specify a Graphics Character in a program line via CHR$( ), the color of the Cursor will not affect the Graphics Character. See Chapter 15 for more details.

Graphics Characters generated from the keyboard can be very useful when a graphics printer or graphics pad (such as the TRS-80 Model GT-16 X-Pad) is connected to the MC-10.

Other Keys with Alternate Definitions

Like a traditional typewriter, all of the keys with alternate definitions (such as the 1 key which has an exclamation mark above it) can produce the alternate meaning by pressing SHIFT and the key. In the case of the 1 key, pressing SHIFT 1 would produce an "!."

The alternate definition will always be generated when you press SHIFT — even if the MC-10 is in Upper/Lowercase Option Mode.

The CONTROL Key

The CONTROL key enables you to use Micro Color BASIC commands without having to type the entire word in. These commands can be used in the "immediate" mode or in program lines.

Note that most of the keys have a Micro Color BASIC keyword printed just above the key itself. To access that keyword, press CONTROL and the key just

below the word you wish to use. For instance, to display the command RUN, type (CONTROL) (1) and RUN will appear on the Screen.

The keys (Q), (A), (W), (S), and (Z) do not have keywords above them. Instead they have special instructions.

(CONTROL) (A) is the backspace command. When you're typing in a program line and want to erase the previous character (or move the Cursor to the left), type (CONTROL) (A).

(CONTROL) (Q) deletes the entire current program line you are typing and returns the Cursor to the left-hand side of the Screen.

(CONTROL) (W) will generate ↑, the exponentiation operator. This will raise a number to the specified power. For instance, typing `PRINT 10 ↑ 2` (ENTER) tells the MC-10 to "square the number 10 and print the answer (100) on the Screen."

(CONTROL) (S) (→) and (CONTROL) (Z) (←) are reserved for special purposes.

The MC-10 and a TV

The TRS-80® MC-10 is capable of generating nine distinct colors. These are nominally defined as:

black	blue	cyan
green	red	magenta
yellow	buff	orange

However, the actual color tones produced by your TV, and the degree of difference between tones, will depend on the quality and color adjustment of your TV — not on the Computer.

When you are using Micro Color BASIC, the display will normally be green with black characters although the lowercase mode inverts this by displaying green characters on a black background.

Micro Color BASIC also displays a blinking "cursor" which changes colors throughout the available spectrum.

Note: The following is an example of simple programming. For further examples of programs which display color as well as sound, see **"Troubleshooting and Maintenance"** later in this manual. For details on programming in general, see the second part of this manual, **"Learning and Using Micro Color BASIC."**

To control the background color of the TV, type in a "command" like this:

CLS *code* (ENTER)

where *code* is one of the numbers which represents an available color. See Table 1.

Code	For This Color
0	black
1	green
2	yellow
3	blue
4	red
5	buff
6	cyan
7	magenta
8	orange

Table 1

For example, type: CLS 3 (ENTER) for a blue background.

Whenever the Screen is cleared (by typing CLS (ENTER) or pressing RESET), the Screen will automatically revert to green no matter what color was previously displayed.

To help you adjust the color and alignment of your TV, we've included two test programs in the **"Troubleshooting and Maintenance"** section of this manual.

Sound

Besides producing a variety of colors, the MC-10 and Micro Color BASIC can control the sound coming from your TV's speaker. The sound volume can be determined manually by adjusting the column control on the TV. The volume can also be adjusted "automatically" by the Micro Color BASIC SOUND statement. Use SOUND this way:

SOUND *pitch-code, duration* (ENTER)

where *pitch-code* may be any number from 1 to 255 — 1 being the lowest available pitch, 255 the highest. *duration* may be any number from 1 to 255; it specifies the duration of the tone in units of approximately .06 seconds.

Most of the musical notes from E (below middle-C) to higher than the highest piano note possible may be approximated by a suitable choice of pitch-code. For example:

```
SOUND 39,10 (ENTER)
```

produces the musical note low-G for .6 seconds.

For a demonstration of all available tones, see the test program section in the **"Troubleshooting and Maintenance"** section of this manual.

The MC-10 and a Cassette Recorder

The MC-10's built-in cassette interface allows you to store data and programs with Radio Shack's optional/extra CCR-81 Computer Recorder (26-1208).

While other recorders can be used, we strongly recommend you use this recorder. Connection and operation will vary with other recorders.

Connect the CCR-81 to the MC-10 according to the instructions described in the section of this manual entitled **"Connection to a Cassette Recorder."** Also refer to the CCR-81 Owner's Manual for further instructions.

Important Note! If you have a printer and a cassette recorder connected to the MC-10 at the same time, the printer must be off-line when you save information to the recorder or random data will be printed. Set the printer's ON-LINE/OFF-LINE switch to OFF-LINE or disconnect the printer from the MC-10 before saving data.

Your MC-10 Computer transfers programs and data on tape at approximately 1500 baud. This is about 190 characters per second, or 11,000 characters per minute.

The most critical adjustment is the volume control on the cassette recorder. For the CCR-81 recorder, the volume control should be set between 3 and 10. The recommended setting is 5.

For any other recorder, the volume level may vary. See your recorder's owner's manual for details.

Saving a Program on Tape (CSAVE)

Whenever you want a permanent copy (one you won't have to re-type) of a Micro Color BASIC program, simply save it on tape with the CSAVE command.

We don't recommend recording over old programs. Bulk erase old tapes or use new ones.

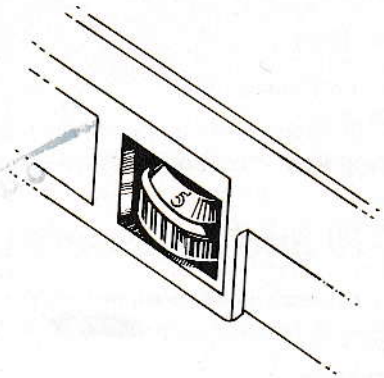


Figure 17. Setting the CCR-81 Recorder's Volume to 5

After the program has been typed into the Computer, you should:

1. Insert a blank cassette tape into the CCR-81 recorder.
2. Type: CSAVE *"filename"* where *filename* is any sequence of eight or fewer characters.
3. Press the recorder's **PLAY** and **RECORD** buttons at the same time until they lock.
4. Press (ENTER).
5. When the program has been saved, OK will appear on the TV Screen. Press the recorder's **STOP** button.

It's a good idea to make more than one copy of a program, preferably on separate cassettes, just in case one is lost or inadvertently erased.

Note: A variation of CSAVE is CSAVE* which allows you to store the contents of a numeric array for future use by CLOAD*. For more details, see Chapter 13 of this manual.

Loading a Program from Tape (CLOAD)

Be sure the recorder's volume control is properly adjusted before loading a program into your Computer.

1. To erase any existing program in the Computer's memory, type: NEW (ENTER)
2. Type: CLOAD *"filename"* where *filename* is any sequence of eight or fewer characters that was previously assigned to a file.

3. Press (ENTER)

4. Press the **PLAY** button on the recorder until it locks.

   The Computer will search for and load the program you have specified. While it is "searching" for the program, the letter **S** will appear in the upper-left corner of the TV Screen.

   When the Computer has found the program, the letter **F** and the filename will appear at the top of the Screen.

   If you do not specify a filename, the first program encountered will be loaded into the Computer.

5. When the program has been loaded, OK will appear on the Screen.

If you type `CLOAD` (ENTER) when a cassette recorder is not connected to the MC-10, the Computer will "hang-up." Press the red RESET Button on the back of the Computer and the OK prompt will return.

Note: A variation of CLOAD is CLOAD* which allows you to load the contents of a numeric array which was previously saved by CSAVE*. For more details, see Chapter 13 of this manual.

Searching for a Program (SKIPF)

If your tape contains more than one program on the same side, the Computer can search through the cassette until it finds the program you need if you use the SKIPF ("skip until you find it") command.

1. Rewind the tape.

2. Type: SKIPF *"filename"* where *filename* is any sequence of eight or fewer characters.

3. Press (ENTER).

4. Press **PLAY** on the recorder until it locks.

5. When the Computer has located the program, OK will appear on the Screen.

If you have forgotten the filename of a particular program, use an improbable filename such as GOBBLE:

```
SKIPF "GOBBLE" (ENTER)
```

All program file names will appear on the Screen as they are encountered and an error message will be displayed at the end of the tape.

Loading Errors

There are several possible error messages that may appear in the upper-left corner of the television Screen when a mistake is made during a loading operation.

You can avoid many problems with tapes by using new, high-quality Radio Shack Computer tapes (26-301) whenever you save a program or data. If it becomes necessary to re-use a tape, you should first erase the contents with a ***Magnetic Bulk Tape Eraser*** (44-210).

If you try loading a blank tape, the MC-10 will search for the program until the tape ends without giving any indication the tape is blank. You will need to press the Computer's RESET button to stop the loading process.

The MC-10 and a Printer

Radio Shack provides a thermal printer, the **TP-10** (26-1261) which is specifically designed for the MC-10. This printer provides a 32-column print out on 4" wide paper and is capable of printing all the graphics the MC-10 can produce.

However, any Radio Shack serial printer or plotters can be used with the MC-10. These include the:

- Line Printer VII (26-1167)
- Line Printer VIII (26-1168)
- DMP-100 (26-1253)
- DMP-120 (26-1255)
- DMP-200 (26-1254)
- DMP-400 (26-1251)
- CGP-115 Plotter/Printer (26-1192)
- FP-215 Flatbed Plotter (26-1193)

Connection to a Printer

To use the MC-10 with any of these printers or plotters, you'll need to use the proper connecting cable. To connect the MC-10 to any of these optional devices, follow the instructions provided earlier in this manual.

Be sure the printer or plotter is configured for serial input at 600 baud from the MC-10. For details on configuring the accessory device, see that unit's owner's manual.

Important Note! If you have a printer and a cassette recorder connected to the MC-10 at the same time, the printer must be off-line when you save information to the recorder or random data will be printed. Set the printer's ON-LINE/OFF-LINE switch to OFF-LINE or disconnect the printer from the MC-10 before saving data.

Micro Color BASIC has two commands associated with the printer. They are:

- LLIST which lists programs on the printer.
- LPRINT which prints program output on the printer.

Important Note! If you attempt to use these commands when a printer is not connected to the MC-10, the Computer will "hang up" and you'll have to press RESET to regain control.

Listing a Program

To find out how LLIST works, type in the following short program after you've connected the MC-10 to a printer.

```
10 REM THIS IS A TEST (ENTER)
20 REM TO TEST THE "LLIST" STATEMENT (ENTER)
```

(Don't worry about "running" the program, the REM statements will prevent an error.)

When the OK prompt is displayed, type: `LLIST` (ENTER). The printer should list the program.

Printing Out Data

LPRINT will send out data to the printer. Try this short program:

```
10 LPRINT "THIS IS A PRINTER TEST FOR THE MC-10"
```

Once you've entered this, type `RUN` and press (ENTER).

The printer should print out: THIS IS A PRINTER TEST FOR THE MC-10.

Part II
Learning and Using Micro Color BASIC

5/ GETTING STARTED

☑ Have you read (and do you understand?) the **"Operation"** section of this manual?
☑ Is your Computer hooked up, powered up, and ready to go?
☑ Is OK displayed on the TV?

If you answered these questions with a "yes," you're ready to begin.

OK is the MC-10's way of telling you that "everything is OK" and the Computer is waiting for you to do something. Whenever the **OK** prompt is displayed, you can type information into the Computer.

If the **OK** prompt is not displayed, either a program is being processed or there is a malfunction somewhere in the System. Press the (BREAK) key to see if OK reappears. If a program was being run, the OK prompt will reappear. If there is a problem with the System, see Appendix D, **"Troubleshooting and Maintenance,"** for a possible solution.

Press (ENTER) a few times. The only thing that should happen is that the Cursor (that blinking block) will move down the left side of the Screen.

(ENTER) serves the same purpose as the carriage return key on a normal typewriter. We call it the "enter" key, because you must press it before you "enter" any information into the Computer's memory. In other words, the MC-10 won't acknowledge anything you type in until you press (ENTER). Once you press (ENTER), the MC-10 will either:

- Store or process the information you entered if it understands what you said

  or

- Display an error message if it didn't understand what you were trying to say. (See Appendix B for a complete list of Error Messages.)

As an example, try typing in the following line exactly as it appears below (be sure OK is displayed):

```
PRINT "HI, I'M THE MC-10"
```

Now check the line. Is it exactly as above? Be sure to check the spelling of the word PRINT and the quotation marks, especially the first quote. If you made any mistakes, press the key-combination of (CONTROL) (A) to "backspace" the Cursor and erase characters.

Now press the (ENTER) key. Your Screen should display this:

```
OK
PRINT "HI, I'M THE MC-10"
HI, I'M THE MC-10
OK
```

Just like you told it, the MC-10 followed your instructions. You told it to "print" a specific message on the Screen and that's exactly what it did.

For something different, type in this:

```
PRINT 2
```

and press (ENTER). This time your Screen should look like this:

```
OK
PRINT 2
 2
OK
```

Notice the difference. When you gave the MC-10 the instruction to "print" the number 2, you didn't include quotation marks around the number.

Whether you realize it or not, you've just learned several things about Micro Color BASIC in this short example. For one thing, you've learned your first command — **PRINT**. An MC-10 command is just like any other command — you instruct the Computer to do something and you can expect the Computer to follow that instruction. PRINT is just one of many Micro Color BASIC commands that we'll be discussing in this section of the manual.

The other thing you just learned is that there are only two types of information the MC-10 understands:

- **String values** which are anything that is enclosed in quotation marks (like "HI, I'M THE MC-10").
- **Numeric expressions** which are numbers or variables which represent numbers (like the number 2).

The Computer sees everything you type as either a String or a Numeric expression (number). If what you type is in quotes, it's a String and the MC-10 sees it and prints it exactly as it is. Try this:

```
PRINT "**********"  (ENTER)
```

Just as an aside . . . The maximum number of characters that can be used at any one time is 127 — that includes spaces and the (ENTER) key. In the case above, that means you could have 118 asterisks since the command

PRINT, the quotation marks, the blank space before the first quote and pressing (ENTER) require 9 memory locations.

If what you type in isn't in quotes, the Computer will treat it as a number by adding, subtracting, multiplying, or dividing it.

The MC-10 as a Calculator

Computers have traditionally been thought of as "number crunchers" and the MC-10 is no exception. With the PRINT command, you can use the MC-10 just like a calculator. Type in the following:

```
PRINT 2 + 2 (ENTER)
```

The Screen will look like this:

```
PRINT 2 + 2
 4
OK
```

If you simply typed in 2+2, the MC-10 would go ahead and add the two numbers, but it wouldn't tell you the answer.

Addition isn't the only math operation the MC-10 can perform. It can also subtract (use the (-) key), multiply (the (*) key), and divide (the (/) key). (The MC-10 has a full range of trigonometric functions — SIN, TAN, COS, etc. — that are discussed later in this manual.)

Try these as examples:

```
PRINT 10 - 5 (ENTER)
 5
OK

? 10 * 10 (ENTER)
 100
OK

PRINT 100/10 (ENTER)
 10
OK
```

Error Messages

By this time, the MC-10 has displayed an "error message" — especially if you misspelled the word PRINT. If you haven't seen an error message yet, type in the following line exactly as it is:

```
PINT "THIS WILL CAUSE AN ERROR" (ENTER)
```

Your TV will look like this:

```
PINT "THIS WILL CAUSE AN ERROR"

?SN ERROR
OK
```

This is a "syntax" error and usually occurs when you make a typing mistake.

An error message will also occur when you ask the MC-10 to do something that is illogical or impossible. For instance, type:

```
PRINT 50/0 (ENTER)
```

The TV will display:

```
?/0 ERROR
```

which is a Division by Zero Error. (Everyone knows you can't divide by zero.) For a complete list of all error messages the MC-10 can produce, see Appendix B.

6/ RE-USABLE INSTRUCTIONS (Programs)

The previous examples used instructions that were processed "immediately" by the MC-10. This method of operation is called the "Immediate Mode"; it describes any instruction that is performed "right now" and is not stored for future repetition. (The Immediate Mode, in other words, is a lot like money—it vanishes as quickly as it appears!)

Most Micro Color BASIC keywords can be used from the Immediate Mode. Use "commands" like PRINT when you want to tell the Computer to do something. Commands can be entered directly. Use "functions," on the other hand, when you want the Computer to tell you something. Functions can also be used in the Immediate Mode but they must be used in conjunction with a command such as PRINT.

The MC-10 has much greater memory capabilities than it's exhibited so far. In fact, it can remember and re-use instructions.

Re-usable instructions are called "programs." These are instructions you can use over and over without having to re-type the message.

Line Numbers

Actually, the MC-10 doesn't remember the instruction itself. Instead, it remembers where that instruction is stored. This is somewhat like your not being able to remember every telephone number in the phone book. But you do know where to look if you want to find a specific person's phone number.

A Computer is much the same. Instead of remembering an alphabetical sequence of letters, however, it remembers numbers.

Each re-usable instruction must begin with a number called a "line number." A line number can be any decimal number between 0 and 63999. For instance:

```
1 A$ = "PROGRAM LINES"
100 PRINT A$
63999 PRINT "THE END"
```

are all valid program lines.

If you try to enter a line like:

```
64110 PRINT "THE END"
```

a Syntax Error will be displayed since 64110 is greater than the allowable line numbers.

We suggest you use line numbers in increments of 10 if possible. You can number your program lines in increments of 1, but good programming practices dictate you should try to use increments of 10 (10, 20, 30, 40, etc.). This is convenient if you want to go back and add program lines in between existing lines (10, 15, 20, 30, etc.).

The NEW Command

Before writing any programs, you should always erase any current programs or data from the MC-10's memory.

There are three ways to erase a program:

- Turn the Computer OFF. When you do so, all contents currently in memory will be erased.
- Type another program using the same line numbers. A new line 10 will replace an old line 10.
- Type in the Micro Color BASIC command NEW.

NEW

NEW simply tells the MC-10 that you want to erase the current contents of memory so you can write a "new" program.

To do this, be sure the OK prompt is displayed and type:

```
NEW (ENTER)
```

The OK prompt will reappear.

Now type `LIST` (ENTER). When you do, nothing will be listed because memory has been erased.

There Has To Be An Easier Way...

One of the biggest features of the MC-10 is its small size. But the small keyboard is also sometimes difficult to type on. Consequently, we've assigned the most commonly used Micro Color BASIC commands and func-

tions as "control" keys. That means you press (CONTROL) and then the key directly under the command or statement.

To save you the trouble of typing NEW over and over, just type:

(CONTROL) (5)

Writing Your First Program

Are you ready to write your first "computer program"? To do so, you'll need to use a line number. For example:

```
10 PRINT "THIS IS MY FIRST MC-10 PROGRAM"
```

Don't press (ENTER) yet!

Check the line for mistakes. If you found any, just press (CONTROL) (A) to backspace the Cursor and make the correction. Then finish typing the line.

Now press (ENTER).

Expand the program by adding another line to it:

```
20 PRINT "THIS IS THE SECOND LINE" (ENTER)
```

The RUN Command

Now that you've written a program, what do you do with it then?

Well, you probably want the Computer to process it. The Micro Color BASIC RUN command will process or "run" the program.

RUN *line number*
line number specifies the program line where execution is to begin. *line number* is optional; if omitted, the first line in the program is used.

Press RESET to clear the TV Screen. When the OK prompt is displayed, type:

```
RUN (ENTER)
```

and the TV will display:

```
THIS IS MY FIRST MC-10 PROGRAM
THIS IS THE SECOND LINE
OK
```

There, you've done it! You've written and run your first Micro Color BASIC program.

In most cases, you'll want to begin running the programs at the beginning. If so, then use RUN the way you just did.

However, if you ever want to run the program from a specified line, just type in the line number where program execution is to begin after the command. For instance, when you type:

```
RUN 20 (ENTER)
```

the Screen will display:

```
THIS IS THE SECOND LINE
OK
```

There Has To Be An Easier Way. . .

To save you the trouble of typing RUN over and over, just type:

(CONTROL) (1)

and the word RUN will appear on the TV. You can then press (ENTER) and program execution will begin at the first line in the program, or specify a line number and press (ENTER) to begin execution somewhere else in the program.

The LIST Command

Just because you've run the program, don't think it's gone away somewhere; it's still in the MC-10's memory — you just can't see it. (After all, we did call it a "re-usable instruction.")

To see the program again, you'll need to "list" it on the TV Screen. Use the LIST command to do this.

LIST *startline - endline*
startline is the first program line you want listed. *startline* is optional; if omitted, the first line in the program is used.
endline is the last program line you want listed. *endline* is optional; if omitted, the last line in the program is used.
If both *startline* and *endline* are omitted, the entire program will be listed.

This command will display the entire program that is currently in the MC-10 memory, or the lines in the program you specify.

If you want the entire program displayed, type `LIST` (ENTER) and the program currently in the Computer's memory will be listed.

If you want a single line displayed, type LIST followed by the line number and press (ENTER). Only that line will be displayed. For example, `LIST 50` (ENTER) will list line 50 only.

If you want to display all lines from the beginning of the program to a specified line, type LIST *-endline* and press (ENTER). For instance, `LIST -100` will list all lines from the beginning of the program to and including line 100.

If you want to display all lines from a specified line to the end of the program, type LIST *startline-* (ENTER). For example, `LIST 50-` (ENTER) will display all program lines from line 50 to the end of the program.

The last way to use LIST is to display specified lines within the program. LIST *startline-endline* (ENTER) will list all program lines within the specified range. For instance, `LIST 50-75` (ENTER) will list all lines from 50 to 75.

To list the program you previously entered, type:

```
LIST (ENTER)
```

and the TV should display:

```
OK
LIST

10 PRINT "THIS IS MY FIRST MC-10 PROGRAM"
20 PRINT "THIS IS THE SECOND LINE"
OK
```

Now list the program by typing:

```
LIST 10 (ENTER)
```

or

```
LIST -20 (ENTER)
```

or

```
LIST 10-20 (ENTER)
```

There Has To Be An Easier Way...

To list the current program without typing the command LIST, press the key-combination of:

(CONTROL) (6)

and the word LIST will appear on the TV. You can then use the command in any of the ways described above.

The LLIST Command

Do you have a printer connected to your MC-10? If you do, you can use the LLIST command to get a print-out on paper of the program.

LLIST *startline - endline*
startline is the first program line you want listed. *startline* is optional; if omitted, the first line in the program is used.
endline is the last program line you want listed. *endline* is optional; if omitted, the last line in the program is used.
If both *startline* and *endline* are omitted, the entire program will be listed.

LLIST can be used exactly like LIST except the program is listed on the printer — not on the TV Screen.

If you want to use the Control Key Command sequence to type LLIST, type:

(L) (CONTROL) (6)

and your Screen will display

```
LLIST
```

Note that if you try to LLIST a program when you do not have a printer connected to the MC-10, you will "hang-up" the Computer. The only thing you can then do is press the RESET Button (on the back of the Computer).

7/ DATA OUTPUT TO THE TV AND PRINTER

All of the example programs in the previous chapters used one Micro Color BASIC command to display information on the TV Screen. That command — PRINT — can take a variety of forms, including one form that lets you output information on a printer (if one is connected to the MC-10).

The PRINT Command

The command that displays ("prints") information on the Screen is PRINT and it's used like this:

PRINT *message*
message is either a string value or a numeric expression. If *message* is a string value, it must be enclosed in quotation marks and will be displayed exactly as it is. If *message* is a numeric expression, it will not be in quotes and will be treated as a number.

As you've already seen, PRINT can be used with either strings or numbers. In technical terms, PRINT simply outputs to the display. This means that anything with the word PRINT before it generates a display on the TV.

PRINTing Strings and Numbers

If you type in a string such as COMPUTER, but do not enclose it in quotes, the Computer will treat the word as a number, assign it the value of zero, and display that value. Try it and the Screen will look like this:

```
OK
PRINT COMPUTER
 0
OK
```

Tell the MC-10 to print 10 COMPUTER:

```
OK
PRINT 10 COMPUTER
```

```
 10 0
OK
```

In this case, the MC-10 treated the number 10 as a number and printed what you told it to print. The word COMPUTER, however, was also treated as a number and assigned the value of zero; that value was then printed.

There Has To Be An Easier Way...

PRINT is a command you'll use constantly. Notice that directly over the (9) key is the word PRINT.

Instead of typing PRINT over and over, just type:

(CONTROL) (9)

When you do, your Screen will look like this:

```
OK
PRINT
```

with the Cursor blinking after the word PRINT, waiting for you to add on what you want printed. You can then type a string or numeric expression and press (ENTER):

```
OK
PRINT "USE THE CONTROL COMMAND K
EYS WHEN YOU CAN" (ENTER)
USE THE CONTROL COMMAND KEYS WHE
N YOU CAN
OK
```

Notice the TV Screen "wraps-around" when it reaches the 32nd character (or space) at the right side of the Screen. This doesn't effect the operation of your Computer or the processing of your program at all. It's perfectly normal.

One More Time...

Unlike other Micro Color BASIC commands, PRINT has an additional method of being typed into the MC-10. Instead of typing the word itself or pressing (CONTROL) (9), just type a question mark (?). For instance, when you type:

```
? 100 (ENTER)
```

The TV will display:

```
100
```

When you get further into programming, you'll notice that you can use a ? as a PRINT statement in a program line. Then, when you "clear" (or erase) the Screen and list the program, Micro Color BASIC will automatically substitute the word PRINT for the ?.

PRINTing Alternatives

In the PRINT examples you've used so far, the MC-10 has displayed a message at the left side of the Screen. Micro Color BASIC also allows you to print information at other Screen locations. You can either use pre-determined PRINT positions, or you can specify where you want information to be PRINTed.

PRINT Punctuation

By adding a punctuation mark to PRINT, you can use a pre-determined PRINT format.

PRINTing Without Punctuation

Type in the following short program:

```
10 PRINT "1ST PRINT"
```

PRINT Suffix	PRINT Format
No punctuation	Prints the message on the next line in columns (one message per line)
Comma (,)	Prints the message in rows (16 spaces before the first character of the next row)
Semi-colon (;)	Prints the message without spaces between the next message item
@ *position*	Prints at the Screen position you specify. *position* is a numeric expression between 0 and 511.
TAB( )	Prints at the specified column position.

Table 2

```
20 PRINT "2ND PRINT"
30 PRINT "3RD PRINT"
```

When you RUN the program, the TV will look like this:

```
1ST PRINT
2ND PRINT
3RD PRINT
OK
```

Notice that nothing has changed in the printing. The second PRINT command simply goes to the next line and displays the message.

PRINTing With a Comma

Now type NEW (ENTER) to erase the program; then type in this program:

```
10 PRINT "1ST PRINT",
20 PRINT "2ND PRINT",
30 PRINT "3RD PRINT",
40 END
```

The only difference between this program and the previous one is the comma (,) at the end of line 10, 20, and 30. When you RUN the program, the display will look like this:

```
1ST PRINT      2ND PRINT
3RD PRINT
OK
```

When you use a PRINT with a comma, Micro Color BASIC skips to the 17th position ("row") to start printing the next PRINT item.

If a PRINT item has more than 17 characters, Micro Color BASIC will go to the next line when printing the next PRINT item.

PRINTing With a Semi-Colon

Type NEW (ENTER) to erase the program. Now type in this program:

```
10 PRINT "1ST PRINT";
20 PRINT "2ND PRINT";
30 PRINT "3RD PRINT";
40 END
```

When you RUN the program this time, the display will look like this:

```
1ST PRINT2ND PRINT3RD PRINT
```

```
OK
```

Notice how the PRINT items are bunched up together. If you want a space between PRINT items when you use semicolon, include it in the string (i.e., within the quotation marks). Line 10, for instance, would look like this:

```
10 PRINT "1ST PRINT ";
```

PRINT @

All of the PRINT formats we've discussed so far have used pre-determined PRINT positions. There is one PRINT variation, however, that lets you specify exactly where on the Screen you want printing to take place.

PRINT @ *position, message*
position specifies a valid Screen print position where printing is to begin and is a numeric expression between 0 and 511. *position* must be followed by a comma.
message is the message you want printed and is a numeric or data string.

When your TV is connected to the MC-10, the TV's Screen is divided into 512 (0-511) individual print positions. With PRINT @, you can specify the exact position you want printing to begin. Figure 18 defines those print positions.

The 512 Screen positions start at 0 (in the upper-left corner of the Screen) and increase as you move to the right. The upper-right corner is position 31. Counting then "wraps-around" to the beginning of the next line. Position 511 is the last position (the lower-right corner).

Type NEW (ENTER) and try this short program:

```
10 PRINT @ 230, "PRINT @ POSITION 230"
```

RUN the program and the message will be printed near the center of the Screen beginning at position 230.

PRINT TAB( )

By adding the TAB( ) suffix to the PRINT command, you can tell Micro Color BASIC exactly where you want to start printing.

PRINT TAB *(column)*
column specifies a position print column on the TV display and is a numeric expression between 0-255.

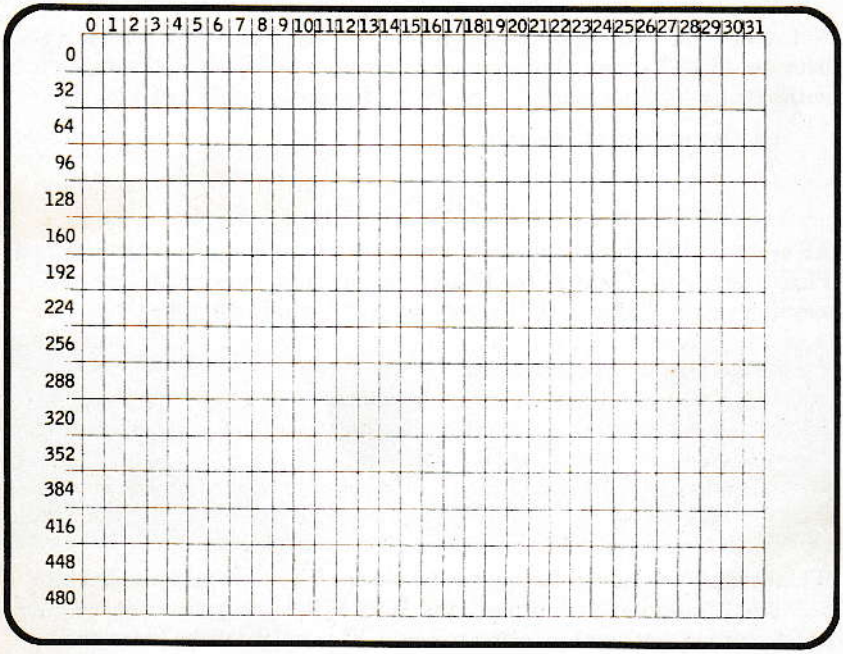


Figure 18. Print @ Screen Positions

There are 32 (0-31) print columns on the TV display and you can specify printing to start at any one of those columns. If column is greater than 31, Micro Color BASIC "wraps-around" to the next line and starts printing there. For instance, PRINT TAB(60) starts printing near the last position on the next line.

PRINT TAB( ) is useful for printing tables that have rows and columns. Try this example:

```
10 PRINT TAB(10)"A";: PRINT TAB(20)"B"
20 PRINT TAB(8)"1000";: PRINT TAB(18)"2000"
30 PRINT TAB(8)"4000";: PRINT TAB(18)"5000"
40 END
```

When you RUN this program, you have a table that looks something like this:

```
 A                B
1000             2000
4000             5000

OK
```

Notice the TAB value *(column)* doesn't have to be a numeric constant—it can be any numeric expression. This means a single TAB*(column)* function in your PRINT statement can generate any number of TAB positions.

The LPRINT Command

The MC-10 can be used with any Radio Shack printer that has a 4-pin Serial Interface Connector. For details about connecting the MC-10 to a printer, see the **"Operation"** section of this manual.

Once a printer is connected to the Computer, use the Micro Color BASIC command LPRINT to get printed copies of the data. (Remember that LLIST will provide printed copies of program listings.)

LPRINT *message*

message is either a string value or a numeric expression. If *message* is a string value, it must be enclosed in quotation marks and will be displayed exactly as it is. If *message* is a numeric expression, it will not be in quotes and will be treated as a number.

LPRINT can be used exactly like PRINT. *The only exception to this rule is that you cannot LPRINT @.*

Important Note! If you use the LPRINT command and a printer is not connected (or if one is connected but is off-line or the power is off), the Computer will "hang-up." When this occurs, you must press RESET (on the back of the Computer) before the OK prompt will reappear.

If you've connected a printer to the MC-10, try this short program:

```
10 LPRINT "THIS WILL BE PRINTED ON THE PRINTER"
20 LPRINT "AND SO WILL THIS";    :REM SEMICOLON
30 LPRINT "AND SO WILL THIS",    :REM COMMA
40 LPRINT TAB(15) "AND FINALLY THIS"    :REM TAB
50 END
```

LPRINTing Lowercase Letters

In the **"Operation"** section of this manual, we discussed the difference between upper- and lowercase letters. If you remember, we mentioned that even though lowercase letters could not be displayed on the Screen, they could be printed on the printer.

Note: Micro Color BASIC does not recognize lowercase commands.

To print out lowercase letters, press the key-combination of SHIFT 0 just before you type in the letters which are to be lowercase. For instance, type:

```
10 LPRINT "THIS IS UPPERCASE."
20 LPRINT "
```

Now press SHIFT 0 and finish the line:

```
THIS IS LOWERCASE."
```

Notice that lowercase letters are displayed in "reverse video." To get back to Normal Mode, press SHIFT 0 again. RUN the program to produce a print-out like this:

```
THIS IS UPPERCASE...
this is lowercase.
```

You must switch back to uppercase letters before you can enter the RUN command.

8/ VARIABLES AND CONSTANTS

Earlier we talked about the two types of data Micro Color BASIC recognizes — Numeric Data and String Data. You can assign both types of data to variables, creating Numeric Variables and String Variables.

Numeric Variables

Throughout this manual, we've referred to the term "numeric expression." A numeric expression is either a number or a "variable" (such as a letter) which has been assigned a numeric value. Once you assign a number to a variable, the MC-10 will remember that number until you erase memory, turn the Computer off, or assign another number to the same variable.

As an example, type this:

```
A = 5 (ENTER)
```

Try to confuse the MC-10 by typing anything you want and pressing (ENTER). To see if the Computer remembers the value of A, type:

```
PRINT A (ENTER)
```

Your TV will look like this:

```
PRINT A
 5
```

In this example, A is the variable (because its value can change — it varies) and 5 is the constant (because its value will not change — it's constant).

What Kind of Variables Does the MC-10 Recognize?

In the preceding example, A is a "numeric variable" because it represents a number. You can use any two-character combination as a numeric variable as long as the first character is a letter; the second character can be either a letter or a number.

The only exceptions are letter combinations which are Micro Color BASIC commands, or statements (called "reserved words") such as IF, OR, NOT, READ, etc.

Micro Color BASIC does not permit variable names to contain embedded reserved words. This means that you cannot use the word NOTE, for instance, since the MC-10 will see the word NOT, which is a reserved word. However, reserved words can contain spaces.

Note that you can use an entire word as a variable. Micro Color BASIC, however, recognizes only the first two letters. This means you can use the word VARIABLE as a variable although the Computer really only looks at the first two letters.

A1
AA
VARIABLE
NUMBER
Z

Acceptable Numeric Variables

Unacceptable Numeric Variables

How Can You Use Numeric Variables Effectively?

The MC-10 allows variables to be used in a variety of ways. For a quick example, try this:

```
A = 5 (ENTER)
B = 10 (ENTER)
PRINT A + B (ENTER)
```

Your TV will display:

```
 15
OK
```

You can perform any of the arithmetic function (subtraction, multiplication, division, and addition) using numeric variables.

String Variables

In the first part of this chapter, we mentioned that a string is any information enclosed in quotation marks. When you PRINT that information, it will be displayed exactly as it appears inside the quotes.

You can assign string data to variables in almost the same manner that you can assign numeric data to variables. The only difference is that string variables must end with a dollar sign ($). For instance:

```
A$ = "STRING VARIABLES"
```

When you type in the command:

```
PRINT A$ (ENTER)
```

the TV will display:

```
STRING VARIABLES
OK
```

The same rules that apply to numeric variables, apply to string variables except that string variables must be suffixed by $.

Adding String Variables

String variables can be merged by "adding" them together. (In technical terms, this is called "concatenation.")

Type NEW (ENTER) (to erase memory). Then type in this program as an example:

```
10 A$ = "MICRO "
20 B$ = "COLOR "
30 C$ = "BASIC"
40 PRINT A$ + B$ + C$
```

It's important to note the blank spaces after MICRO and COLOR. They must be included within the quotes if you want spaces between the words.

When you RUN the program, your TV will display:

```
MICRO COLOR BASIC
```

If you try to "subtract," "multiply," or "divide" string variables, Micro Color BASIC will return a Type Mismatch Error (?TM ERROR).

The LET Command

When you assign variables, you have the option of using the LET command in the following manner:

LET *variable* = *constant*
variable is either a string or numeric variable and *constant* is the value assigned to variable.

To use LET, simply place it before the variable (string or numeric) and assign the value as normal. For instance:

```
LET A = 5
```

is the same as

```
A = 5
```

For string variables:

```
LET A$ = "STRING VARIABLE"
```

is the same as:

```
A$ = "STRING VARIABLE"
```

LET can make programs easier to follow, but remember that it takes up memory space and is optional.

9/ STOP, LOOK, AND LISTEN

Do you have a program currently in memory? If you do, add one more line to finish it. For instance:

```
100 END
```

Whenever the Micro Color BASIC encounters the command END in a program, it immediately stops. Of course, if there are no other instructions at the end of a program, it will stop anyway; but good programming practices dictate that you should always end your programs with END.

END

Sometimes, you may want to even put the END command in the middle of a program, especially if you're using "subroutines" (more on those later). Here's an example of how you might want to use END:

```
10 PRINT "THIS IS THE TEST FOR 'END'"
20 PRINT "THIS LINE WILL BE PRINTED"
30 END
40 PRINT "BUT THIS LINE WON'T"
50 END
```

Now RUN it. Your TV will display this:

```
THIS IS THE TEST FOR 'END'
THIS LINE WILL BE PRINTED
OK
```

There Has To Be An Easier Way . . .

Notice that the Control Command key sequence of:

(CONTROL) (:) will also display the END command.

The STOP Command

A close cousin of END is the STOP command.

STOP

STOP also halts program execution. However, it is generally used inside the program instead of at the end. The sample program used above would be a more "correct" program if it used STOP like this:

```
10 PRINT "THIS IS THE TEST FOR 'END'"
20 PRINT "THIS LINE WILL BE PRINTED"
30 STOP
40 PRINT "BUT THIS LINE WON'T"
50 END
```

When you RUN the program with line 30 changed to STOP, the results are almost the same:

```
THIS IS THE TEST FOR 'END'
THIS LINE WILL BE PRINTED

BREAK IN 30
OK
```

This points out the main difference between STOP and END. With END, Micro Color BASIC assumes the program has ended. With STOP, the program has simply been interrupted.

There Has To Be An Easier Way . . .

Instead of typing in the command STOP everytime you want to use it, simply press the key-combination of:

(CONTROL) (−)

and STOP will appear on the Screen.

The (BREAK) Key

Pressing (BREAK) during program execution is another way of stopping program execution. While STOP halts program execution "automatically," pressing (BREAK) halts it "manually."

When you press (BREAK), a message similar to the following is displayed:

```
BREAK IN 30
OK
```

The CONT Command

Once a program has been interrupted by STOP or END, or by pressing (BREAK), you can continue program execution with the CONT ("continue") command.

CONT

CONT is handy to use in the Immediate Mode when you're trying to find out what's wrong with ("debug") a program.

For instance, you already know what will happen if you execute the following program:

```
10 PRINT "THIS IS THE TEST FOR 'END'"
20 PRINT "THIS LINE WILL BE PRINTED"
30 STOP
40 PRINT "BUT THIS LINE WON'T"
50 END
```

The TV will display:

```
THIS IS THE TEST FOR 'END'
THIS LINE WILL BE PRINTED

BREAK IN 30
OK
```

At this point, type:

```
CONT (ENTER)
```

and the rest of the program will be executed:

```
BUT THIS LINE WON'T
OK
```

There Has To Be An Easier Way . . .

To use CONT without typing it in, press the key-combination of:

(CONTROL) (2)

and then press (ENTER).

The CLS Command

Are you tired of pressing RESET to clear the Screen? Why not let the MC-10 clear its own Screen. To do this, use the CLS ("Clear Screen") command:

CLS *(color-code)*
color-code specifies the background color of the TV Screen and is a numeric expression between 0 and 8. *color-code* is optional; if omitted, 1 is used. Note: The ( ) are optional.

CLS will clear the Screen, while leaving the current contents of memory intact. If you simply type in `CLS` (ENTER), Micro Color BASIC will clear the Screen with the normal, default color (green).

If you include a number between 0-8 after the command (see Table 3), the Screen will be cleared in the specified color.

For instance, if you type `CLS 8` (ENTER) (remember that the parentheses are not required), the Screen will turn orange. Now you know for sure that it's a "color computer."

Why is there a green stripe across the top? Because the MC-10 cannot display regular characters on anything but a green background. (Graphics Characters are a different story however.)

Note that if color-code is a number greater than 8, the message `MICRO-SOFT` or an `?FC ERROR` will appear and the Screen will turn green.

CLS is a useful command to use at the beginning of a program when you want the program listing (or any other display) to be erased.

Code	For This Color
0	black
1	green
2	yellow
3	blue
4	red
5	buff
6	cyan
7	magenta
8	orange

Table 3

There Has To Be An Easier Way . . .

To enter the CLS command the easy way, press the key-combination of:

(CONTROL) (8)

You can then either type in a color code and press (ENTER) or just press (ENTER) for a green Screen.

Add CLS to a program by typing:

```
5 CLS (ENTER)
```

When you list the program, the TV will display:

```
5 CLS
10 PRINT "THIS IS MY FIRST MC-10 PROGRAM"
20 PRINT "THIS IS THE SECOND LINE"
50 END
```

When you RUN it this time, the Screen will automatically clear.

The SOUND Command

Since a TV has sound capabilities, and since the MC-10 was designed to be connected to it, your Computer can also generate audio output. To produce a sound, use the Micro Color BASIC SOUND command.

SOUND *tone, duration*
tone specifies the sound you wish to generate and is a numeric expression between 1 and 255.
duration specifies the length of time tone is to be generated and is a numeric expression between 1 and 255. 1 is approximately 7.5/100 of a second.
Both *tone* and *duration* must be specified.

Before using SOUND, be sure the volume is set to a normal listening range.

Type in this:

```
SOUND 1,100 (ENTER)
```

When you press (ENTER), the Computer's lowest tone will be produced for 7.5 seconds. Want to try the highest tone? Try this:

```
SOUND 255,100 (ENTER)
```

That, as they say, is the other end of the sound spectrum.

Try some other numbers, but don't exceed the 1-255 range for either *tone* or *duration* or you'll get an ?FC ERROR. And don't forget, both *tone* and *duration* must be included or a ?SN ERROR will occur.

The REM Statement

Are you ready for a different type of program? Type in this:

```
10 REM THIS IS MY 2ND PROGRAM
20 REM IT USES "REM"
```

LIST the program and check for typing errors—especially in the spelling of REM.

Now RUN the program. Nothing happened, right? No error messages appeared and the OK prompt returned?

REM *message*
message can be any information you want displayed.

The REM ("remark") statement allows you to include extraneous information in a program line without generating an error message. What happens is that any time Micro Color BASIC encounters a REM statement, it simply ignores everything that comes after it.

You'll find REM a handy way to include explanatory messages within programs or program lines. Note that REM can be used in the middle of a program line as long as it is preceded by a colon (:). The following are all valid uses of REM:

```
10 REM ARITHMETIC PROGRAM
20 PRINT 10 + 5  :REM ADDITION
30 PRINT 10 - 5  :REM SUBTRACTION
40 PRINT 10 / 5  :REM DIVISION
50 PRINT 10 * 5  :REM MULTIPLY
60 END
```

10/ INTERACTIVE PROGRAMMING AND THE MC-10

Up to this point, the programs you've written in Micro Color BASIC have been fairly straightforward. You've given the MC-10 instructions, and the Computer has followed those instructions.

Computers can be somewhat more inquisitive, though. They'll ask you questions — as long as you tell them what questions to ask. (In other words, they're not quite as smart as they'd like you to think!)

The INPUT Command

The Micro Color BASIC INPUT command enables "prompted" input. This means the Computer will ask you a question (the prompt) and you must respond with an answer (the input).

INPUT *"message"; variable*
message is the prompting message you want displayed on the TV and is a data string. *message* must be enclosed in quotes and followed by a semicolon.
variable can be one or more string or numeric variables.

INPUT can be used in Program Mode only. If you use the command in the Immediate Mode, an Illegal Direct Statement Error message will be displayed.

Clear memory by typing NEW (ENTER). Then, to see how INPUT works, try this example:

```
10 CLS
20 INPUT "WHAT IS YOUR NAME";NAME$
30 PRINT "HELLO ";NAME$
40 END
```

Now RUN the program. Your Screen should display:

```
WHAT IS YOUR NAME?
```

(Micro Color BASIC automatically supplies the question mark since it knows that INPUT implies question.)

Simply type in your name and press (ENTER). For instance:

```
WHAT IS YOUR NAME? JON (ENTER)
```

When you do so, the TV will change to:

```
WHAT IS YOUR NAME? JON
HELLO JON
OK
```

Do you think a little more enthusiasm is required when someone says "hello" to you? Change line 30 to this:

```
30 PRINT "HELLO ";NAME$; "!"
```

INPUT allows you to "bury" variables between string items as long as you separate items with a semicolon.

Numbers can also be used with the INPUT statement. Type NEW (ENTER) and try this:

```
10 CLS
20 INPUT "WHAT IS THE FIRST NUMBER"; A
30 INPUT "WHAT NUMBER DO YOU WANT TO ADD TO IT"; B
40 C = A + B
50 PRINT "THE ANSWER IS "; C
60 END
```

Now RUN the program. When Micro Color BASIC asks you what the first number is, type in a number and press (ENTER). For instance, type 10 (ENTER). Next, you'll be asked what number you want to add to the first number; type in another number and press (ENTER). (If you enter anything other than a number, you'll be asked to ??REDO the entry.) The answer will then be printed on the TV.

You can also use INPUT to ask more than one question. After typing NEW (ENTER), enter this program:

```
10 CLS
20 INPUT "WHAT'S YOUR NAME AND AGE
   (NAME,AGE)";NAME$,AGE
25 CLS
30 PRINT NAME$; " IS"; AGE; " YEARS OLD."
40 END
```

When you're asked:

```
WHAT'S YOUR NAME AND AGE (NAME,AGE)?
```

Type in the answer such as:

```
ASHER,14 (ENTER)
```

and the Screen will look like this:

```
ASHER IS 14 YEARS OLD
OK
```

(In this program, we used CLS to clear the prompt off the Screen before you answer the question.)

There are two things to notice about this short program:

- String and numeric variables can be "mixed and matched" in the same prompt.
- You can use a number of variable items with an INPUT statement.

Whether you have a single prompt or a dozen prompts, Micro Color BASIC will display the message ?EXTRA IGNORED if you enter more responses than you have prompts.

INPUT, used in the manner above, has a built-in PRINT statement. Although it's a little redundant, you can also use INPUT like this:

```
10 CLS
20 PRINT "WHAT IS YOUR NAME"
30 INPUT NAME$
40 CLS
50 PRINT "HELLO"; NAME$
60 END
```

This program does the same thing the first example in this chapter did (also listed below), but it takes two extra program lines to do it.

```
10 CLS
20 INPUT "WHAT IS YOUR NAME";NAME$
30 PRINT "HELLO ";NAME$
40 END
```

How about a really practical program that uses INPUT? Try this one:

```
10 CLS
20 REM *** CENTIGRADE TO FAHRENHEIT CONVERSION ***
30 INPUT "WHAT IS THE TEMPERATURE IN DEGREES
   CENTIGRADE";C
40 F = (9/5) * C + 32
```

```
50 PRINT C; "DEGREES CELSIUS = "; F; "DEGREES
   FAHRENHEIT."
60 END
```

There Has To Be An Easier Way . . .

INPUT can be entered using the Command Function Keys. Type:

(CONTROL) (@)

More Interaction . . .

Up to now, all of the programs have been strictly one-shot affairs. You type RUN (ENTER), the Computer executes the program, prints the results on the TV (if any), and comes back with the OK prompt. To repeat the program, you have to type RUN (ENTER) again.

One thing about Computers is that they're especially good at any procedure which needs to be repeated. So how do you make the MC-10 repeat things? It's easy, just use . . .

The GOTO Command

Anytime Micro Color BASIC encounters the GOTO statement in a program line, it will immediately skip to ("goto") the specified line and continue program execution at that point. This is referred as an "unconditional branch" since Micro Color BASIC must branch to the specified line under any condition.

GOTO *line number*
line number is a valid program line number between 0 and 63999.

If you tell Micro Color BASIC to goto a line number that is not in the program, an Unlisted Line Number Error message will be displayed.

Make one of the previous programs more efficient by using GOTO. Try this one:

```
10 INPUT "WHAT IS YOUR NAME";NAME$
20 PRINT "HELLO ";NAME$
30 GOTO 10
```

When you RUN this program, you'll be asked your name and the MC-10 will print it on the TV over and over. This is called an "endless loop" since there

is no way out of the program. It executes lines 10, 20, and 30 — then loops back to 10 and repeats the process.

To get out of the endless loop, press (BREAK) (or turn the MC-10 off or press RESET).

Putting GOTO in the middle of a program will cause Micro Color BASIC to skip over all lines that are between the GOTO statement and the specified line. For instance:

```
10 CLS
20 PRINT "THIS IS LINE 20"
30 GOTO 50
40 PRINT "THIS IS LINE 40"
50 PRINT "WE SKIPPED TO LINE 50"
60 END
```

When you RUN this program, the TV will display:

```
THIS IS LINE 20
WE SKIPPED TO LINE 50
OK
```

indicating that program line 40 was totally ignored.

Another way to create an endless loop is to GOTO the current line. This lets you "hold" whatever is currently being displayed (without returning to the OK prompt) because the program continues execution. For example:

```
10 PRINT "PRESS <BREAK> TO RETURN TO 'OK'"
20 GOTO 20
```

When you RUN the program, the Screen will display:

```
RUN
PRESS <BREAK> TO RETURN TO 'OK'
```

until you press (BREAK). At that time, the message:

```
BREAK IN 20
OK
```

will appear.

There Has To Be An Easier Way

GOTO can be entered using the Command Function keys. To enter this command, type:

(CONTROL) (J)

The ON . . . GOTO Command

Remember that we said GOTO was an "unconditional branching" statement. Micro Color BASIC allows you to vary the format of GOTO to allow it to test certain values before skipping to a line and to make multi-way branches. The ON . . . GOTO command does this.

ON *test value* **GOTO** *line number*
test value is a numeric expression between 0-255.
line number is a valid program line number between 0 and 63999. Multiple line numbers must be separated by a comma.

The following program doesn't use ON . . . GOTO, but it should. RUN it anyway. Then we'll re-write the program so you can see some of the power of the ON . . . GOTO statement. (We're going to use a statement that hasn't been discussed yet—IF. It'll be described later.):

```
10 INPUT "TYPE IN A NUMBER BETWEEN 1 AND 5";N
20 IF N = 1 GOTO 110
30 IF N = 2 GOTO 130
40 IF N = 3 GOTO 150
50 IF N = 4 GOTO 170
60 IF N = 5 GOTO 190
70 PRINT N; "IS NOT BETWEEN 1 AND 5"
99 END
110 PRINT "N = 1"
120 END
130 PRINT "N = 2"
140 END
150 PRINT "N = 3"
160 END
170 PRINT "N = 4"
180 END
190 PRINT "N = 5"
199 END
```

(Hint: If you want to execute the program repeatedly and not have to type RUN (ENTER) over and over, replace each of the END statements with GOTO 10.)

To see how ON . . . GOTO can shorten your programs, delete lines 20, 30, 40, 50, and 60 (simply type the line numbers and press (ENTER)) and replace them with this single line:

```
20 ON N GOTO 110, 130, 150, 170, 190
```

Now RUN the program. See, it does the same thing except it does it with four fewer program lines. What happens is this:

Line 20 says if N = 1, then GOTO line 110; if N = 2, then GOTO line 130; if N = 3, then GOTO line 150; if N = 4, then GOTO line 170; if N = 5, then GOTO line 190. If N doesn't equal any of those numbers, then move on to the next program line (line 70).

There are variations on using ON . . . GOTO. For instance, change line 20 to:

```
20 ON N - 1 GOTO 110, 130, 150, 170, 190
```

In this case, if you enter the number 3, Micro Color BASIC will subtract 1 from it and display the result as N = 2, etc.

The GOSUB/RETURN Commands

A close relative of GOTO is the GOSUB statement which is used in conjunction with RETURN. When Micro Color BASIC is performing one routine, and it needs to execute a separate "subroutine" to complete the main routine, use GOSUB. Then, to return to the next statement of the main routine, use RETURN.

GOSUB *line number*
line number is a valid program line number between 0 and 63999.

RETURN

There are two important rules concerning the use of GOSUB:

- Use END or GOTO statements directly ahead of the subroutine to insure that Micro Color BASIC can enter the subroutine only via a GOSUB.
- Every subroutine must end with a RETURN.

When a subroutine ends, it always returns control to the statement following the GOSUB.

So What is a Subroutine?

A subroutine is a short, but very specialized, program which you build into a large program to meet a specialized need. GOSUB is used to call that subroutine by directing program control to the specified line. As we said, RETURN returns control to the main program.

Type NEW (ENTER) and try this example:

```
10 PRINT "1. THIS WILL BE PRINTED FIRST"
20 GOSUB 100
30 PRINT "3. THIS WILL BE PRINTED THIRD"
40 END
100 PRINT "2. THIS WILL BE PRINTED SECOND"
110 RETURN
```

When you RUN the program, 1, 2 and 3 will be displayed in the correct order.

For a more powerful example of GOSUB, type NEW (ENTER) and this program (even though it has a couple of commands we haven't covered yet):

```
10 INPUT "TYPE ANY NUMBER"; N
20 GOSUB 1000
30 ON T + 2 GOTO 50,70,90
40 END
50 PRINT "THE NUMBER IS NEGATIVE."
60 END
70 PRINT "THE NUMBER IS ZERO."
80 END
90 PRINT "THE NUMBER IS POSITIVE."
100 END
1000 IF N < 0 THEN T = -1
1010 IF N = 0 THEN T = 0
1020 IF N > 0 THEN T = +1
1030 RETURN
```

There Has To Be An Easier Way . . .

Both GOSUB and RETURN can be entered using the Command Function keys.

To enter GOSUB, type:

(CONTROL) (D)

To enter RETURN, type:

(CONTROL) (F)

The ON . . . GOSUB/RETURN Command

Just like ON . . . GOTO, GOSUB can be used to branch subroutines. In fact, ON . . . GOSUB can be used exactly like ON . . . GOTO except that each branch to a subroutine must end with a RETURN statement like this:

```
10 INPUT "WHAT'S BEHIND DOOR #1, #2, OR #3 (TYPE
   1,2,3)"; N
20 ON N GOSUB 100,200,300
30 END
100 PRINT "YOU JUST WON AN MC-10 COMPUTER"
110 RETURN
200 PRINT "YOU JUST WON A CASSETTE RECORDER TO GO
    WITH YOUR MC-10"
210 RETURN
300 PRINT "YOU JUST WON A PRINTER TO USE WITH YOUR
    MC-10"
310 RETURN
```

ON . . . GOSUB tells the Computer to look at the number following ON. In this case, that number is N. If it is a 1, Micro Color BASIC goes to the subroutine beginning at the first line number following GOSUB. If N is 2, Micro Color BASIC goes to the subroutine beginning at the second line number; if N is 3, the Computer skips to the third line number. What if N is 4? Since this is not an option, Micro Color BASIC ignores it and goes to the next line in the program.

11/ OPERATORS AND CONTROLLED LOOPS

Micro Color BASIC recognizes both "relational operators" and "logical operators." Relational operators allow Micro Color BASIC to compare values, make a decision about those values, then take appropriate action.

There are three relational operators that enable the MC-10 to compare one value with another:

- Is greater than (>).
- Is less than (<).
- Is equal to (=).

You can also combine these to produce three more operators:

- Is not equal to (<>).
- Is less than or equal to (<=).
- Is greater than or equal to (>=).

(If you're not familiar with greater than and less than symbols, just remember that the smaller part of the < symbol points to the smaller of the two quantities being compared. For instance, 1<2 and 4>3.)

By adding these six relational operators to the four math operators described in Chapter 5 (+, −, *, and /), you can create a powerful system of comparing and calculating that becomes the central core of everything that follows.

Logical operators, on the other hand, simply "test" values to see if they are true or false (in electrical terms, on or off).

You can also combine two logical operations (tests) and then test that value. For instance, if the complete operation you're testing is composed of 1<2 AND 2<3, then both elements must be true before the test can pass. Or if it is necessary for only one element to be true for the test to pass, you might say either 1<2 OR 2<3.

Relational Operators

IF *test* **THEN** *action*

test is the value which is being compared and must include a relational operator (>, <, or =).

action is the operation performed once test has been compared.

Even though you may not realize it, you're probably familiar with how IF . . THEN statements work. For instance:

You need to rake the leaves in the backyard. IF the rake is in the garage, you'll get it and THEN rake the yard. IF it is not in the garage, you'll look in the basement, THEN rake the yard.

Try this program to see how the MC-10 uses IF . . . THEN:

```
10 INPUT "GUESS THE NUMBER I'M THINKING OF (1 TO
   5)";N
20 REM *** THE NUMBER IS 4 ***
30 IF N > 4 THEN GOSUB 100
40 IF N < 4 THEN GOSUB 100
50 IF N = 4 THEN GOTO 200
60 GOTO 10
70 END
100 PRINT "WRONG NUMBER - TRY AGAIN"
110 RETURN
200 PRINT "RIGHT! THE NUMBER IS 4"
210 END
```

Line 30 says, "IF the number you type in is greater than 4, THEN go down to line 100 and print the message that the answer is wrong, then return to try again." Line 40 says, "IF the number you type in is less than 4, THEN go down to line 100 and print the message that the answer is wrong, then return to try again." Finally, line 50 says, "IF the number you type in is exactly 4, THEN go to line 200 and print the message that the answer is correct."

By combining relational operators, we can save typing (and memory). Change the program by deleting lines 30 and 40 and typing this line:

```
30 IF N < > 4 THEN GOSUB 100
```

The program works the same, but you've shortened it by a line.

IF . . . THEN and the Implied GOTO

IF . . . THEN has the additional feature of "implying" the command GOTO. For instance, line 50 can be shortened to this:

```
50 IF N = 4 THEN 200
```

This means the same as "IF N = 4 THEN GOTO program line 200." Anytime a line number follows THEN, Micro Color BASIC will goto that line.

There Has To Be An Easier Way . . .

Both IF and THEN can be entered using the Command Function keys.

To enter IF, type:

(CONTROL) (G)

To enter THEN, type:

(CONTROL) (H)

Logical Operators (AND/OR/NOT)

Besides the relational operators IF and THEN, Micro Color BASIC also has the "logical operators" AND, OR, and NOT. These perform TRUE/FALSE tests and are used in conjunction with IF . . . THEN.

IF *test1* **THEN** *action1* **AND/OR/NOT** *test2* **THEN** *action2*
test1 is the value which is being compared and must include a relational operator (>, <, or =).
action1 is the operation performed once test1 has been compared.
test2 is the alternate value which is being compared and must include a relational operator (>, <, or =).
action2 is the alternate operation performed once *test2* has been compared.

AND takes two tests and checks whether both pass or not. If either test fails, then "test AND test" fails; if both tests pass, then "test AND test" passes.

OR takes two tests and checks whether either test passes or not. If at least one passes, "test OR test" passes; if both tests fail, then "test OR test" fails.

NOT takes a single test and checks whether it passes or not. If the test fails, "Not test" passes; if test passes, "NOT test" fails.

To understand the concept of logical operators before you begin programming with them, see if you can predict the answers to the following tests:

1. If (1 = 1) AND (2 = 3), then PASS (*Yes* or *No*).
2. If (1 = 1) OR (2 = 3), then PASS (*Yes* or *No*).
3. If NOT (1 = 1 AND 2 = 3), then PASS (*Yes* or *No*).
4. IF NOT (1 = 1 OR 2 = 3), then PASS (*Yes* or *No*).
5. IF NOT (NOT 1 = 1 OR 2 = 3), then PASS (*Yes* or *No*).

Ready for the answers?

1. No, this test would not pass.
2. Yes, this is true.
3. Yes, this is also true.
4. No, this test would not pass.
5. And Yes, this test would pass.

Here's a short program that uses logical operators:

```
5 INPUT "WHAT IS THE STUDENT'S NAME"; N$
10 INPUT "WHAT IS THE FINAL GRADE"; F
20 INPUT "WHAT IS THE MIDTERM GRADE"; M
30 INPUT "WHAT IS THE GRADE FOR HOMEWORK"; H
40 IF (F > 60) OR (M > 70) AND (H > 75) THEN 70
50 PRINT "FAILED"
60 END
70 PRINT N$ "PASSED"
```

In this example, the > sign is used instead of = as a logical test. The student passes if:

- He or she has a final grade over 60
- OR a midterm grade over 70 AND a homework grade over 75.

The next program mixes equals, greater than, and less than signs in the same program. It determines and reports whether the two numbers you input are both positive, both negative, or have different signs.

```
5 REM *** INPUT NUMBERS AND INCLUDE THE SIGNS ***
10 INPUT "FIRST NUMBER"; X
20 INPUT "SECOND NUMBER"; Y
30 IF (X >= 0) AND (Y >= 0) THEN 70
40 IF (X < 0) AND (Y < 0) THEN 90
50 PRINT "OPPOSITE SIGNS"
60 END
70 PRINT "BOTH POSITIVE"
80 END
90 PRINT "BOTH NEGATIVE"
100 END
```

FOR . . . TO . . . STEP . . . NEXT

GOTO and GOSUB allow you to create simple program loops in which you can loop back through a program. The problem with simple loops, however,

is that they are usually "endless" — they go on forever until you press (BREAK) or turn off the Computer.

FOR . . . NEXT allows you to create "controlled" loops which loop only as many times as you want — 10 times, 50 times, 100 times, etc.

FOR *counter* = *start value* **TO** *end value* **STEP** *increment*
counter is a numeric variable
start value is the numeric expression where counting is to begin.
end value is the numeric expression where counting is to stop.
increment is the number that will be added to counter each time the loop is repeated. **STEP** *increment* is optional; if omitted, 1 is used. If *increment* is a negative number, the loop counts down instead of up.

NEXT *counter*
counter is a numeric variable

A major difference between a computer and a calculator is the computer's ability to do the same thing over and over. FOR . . . NEXT is such an important part of program repetition that few of the programming areas we'll explore from now on will exclude it. Its simplicity and variations are the heart of its effectiveness.

Type in this program:

```
10 REM *** PRINT MESSAGE FIVE TIMES ***
20 PRINT "THIS IS THE MESSAGE"
30 PRINT "THIS IS THE MESSAGE"
40 PRINT "THIS IS THE MESSAGE"
50 PRINT "THIS IS THE MESSAGE"
60 PRINT "THIS IS THE MESSAGE"
70 END
```

When you RUN it, it displays the appropriate message five times. It works — but it's awkward. FOR . . . NEXT can do the same thing only more efficiently:

```
10 REM *** PRINT MESSAGE FIVE TIMES ***
20 FOR X = 1 TO 5
30 PRINT "THIS IS THE MESSAGE"
40 NEXT X
50 END
```

FOR . . . NEXT does more than just count imaginary numbers, however. You can also use it to increment what appears on the TV Screen. Use the command sequence like this:

```
10 REM *** PRINTS THE MESSAGE AND THE NUMBER ***
20 FOR N = 1 TO 5
30 PRINT "THIS IS NUMBER"; N
40 NEXT N
50 END
```

What happens in both cases is that line 20 sets the counter at 1; line 30 prints the message; line 40 says to "loop" to the next value of N (which is 2), execute the statement, and continue until the value of N is 5. When N = 5, program control drops to the next statement, which is line 50.

Notice that in this case, a program statement is executed between the FOR and the NEXT. If you want to have a short delay in program execution, tell the MC-10 to count in between. For example, insert this line:

```
35 FOR DELAY = 1 TO 300: NEXT DELAY
```

Now when you RUN it, the MC-10 prints the message, counts to 300, prints the next message, counts to 300, etc.

As you can see, Micro Color BASIC allow you to have "nested loops" or loops within loops. Another way to create nested loops is to have a FOR . . . NEXT loop inside a GOTO loop. If you do this, it's essential that the innermost loop be completely contained inside the outermost loop. In other words, when you open one loop inside another, you must close the inner loop before closing the outer loop:

```
10 FOR J = 1 TO 3
20 FOR J2 = 1 TO 2
30 NEXT J2
40 NEXT J
50 END
```

Right!

```
10 FOR J = 1 TO 3
20 FOR J2 = 1 TO 2
30 NEXT J
40 NEXT J2
50 END
```

Wrong!

To see a nested loop in action, RUN the following 24-hour, stop watch program.

```
10 INPUT "ENTER THE CURRENT HOUR"; HR
20 INPUT "ENTER THE CURRENT MINUTE"; MN
30 INPUT "ENTER THE CURRENT SECOND"; SC
40 FOR HR = HR TO 23
50 FOR MN = MN TO 59
60 FOR SC = SC TO 59
```

```
70 PRINT HR; ":"; MN ":"; SC
80 FOR T = 0 TO 735
90 NEXT T
100 NEXT SC: SC = 0
110 NEXT MN: MN = 0
120 NEXT HR: HR = 0
130 GOTO 40
```

In the above program, the Seconds counter is the innermost loop; the Minutes counter is the middle loop; and the minutes reset to zero is in the outer loop. Remember to close inner loops before outer loops.

Are you ready for a little advanced programming? In other words, do you want to put some of the things you've learned to work? If so, modify the program above so that it becomes an alarm clock. Experiment on your own, then add the lines below to see one way to do it:

```
5 INPUT "WHAT HOUR DO YOU WANT TO GET UP";HA
8 INPUT "WHAT MINUTE DO YOU WANT TO GET UP";MA
95 IF HR = HA THEN GOSUB 200
200 IF MN = MA THEN GOSUB 300
210 RETURN
300 PRINT "TIME TO GET UP!"
310 SOUND 100,100
320 END
```

The STEP Option

Normally, FOR . . NEXT "steps" through the counter 1 count at a time. However, if you want to increment at a different rate, use the STEP option and specify the counting increment. For instance:

```
10 FOR X = 1 TO 100 STEP 5
20 FOR DELAY = 1 TO 300: NEXT DELAY
30 PRINT X
40 NEXT X
50 END
```

When you RUN the program, it won't display every number between 1 and 100. Instead, it will display every 5th number (1, 6, 11, 16, 21, 26, etc.).

There Has To Be An Easier Way . . .

Notice that you can use the Command Function keys to make entering FOR, NEXT, and STEP a little easier.

To enter FOR, type:

(CONTROL) (U)

To enter NEXT, type:

(CONTROL) (I)

To enter STEP, type:

(CONTROL) (O)

12/ READING DATA

So far, you've learned that there are two ways to enter numbers into your programs:

- By simple assignment of variables to constants (LET A = 5).
- By using the INPUT statement (INPUT A).

A third way to input data is with the DATA statement. The DATA statement lets you store "data" in a program. When you want to "read" the data you've previously stored, use the READ statement.

DATA *item,item,item...*
item is either a string or numeric constant. Expressions are not allowed. If *item* includes colons, commas, or leading blanks, you must enclose these values in quotes. Every *item* but the last must be followed by a comma.

READ *variable,variable,variable...*
variable is either a string or numeric variable. Every *variable* but the last must be followed by a comma.

DATA lines can only be read by the READ statement. There must be the same number of variables following READ that there are items following DATA. If there are more READ variables than DATA items, an Out of Data Error message will appear.

DATA lines can be placed anywhere in the program. For convenience, we suggest you place your DATA statements either at the first or at the end of the program so they can be easily found.

Try this short example:

```
10 DATA 1,2,3,4,5
20 FOR N = 1 TO 5
30 READ A
40 PRINT A;
50 NEXT N
60 END
```

When you RUN the program, the Display will look like this:

```
 1  2  3  4  5
OK
```

Notice that the DATA items (line 10) are outside the loop. Delete line 10 (type 10 (ENTER)) and retype it as line 25.

```
25 DATA 1,2,3,4,5
```

It doesn't make any difference. Now delete line 25 and retype it as line 55.

```
55 DATA 1,2,3,4,5
```

Still no difference.

What happens is that Micro Color BASIC loops through N five times, reading the variable A (which contains the data) each time. Each time the program loops, the data is printed — but each time, the value of A has been incremented.

It's important to note that each piece of data in a DATA line can be read only once each time the program is run. The next time a READ statement requests a piece of data, it will read the next piece of data in the DATA line, or, if that line is all used up, go on to the next DATA line and start reading it. Is there a way around this? Yes, just use . . .

The RESTORE Statement

Whenever Micro Color BASIC comes across a RESTORE, all DATA lines are restored to their original "unread" condition.

This includes those which have been read and those that haven't.

RESTORE

To see how useful RESTORE can be, change the above program back to its original state:

```
10 DATA 1,2,3,4,5
20 FOR N = 1 TO 5
30 READ A
40 PRINT A;
50 NEXT N
60 END
```

and add this line:

```
57 GOTO 10
```

From everything you've learned so far, you might think the program would just re-cycle through an "endless" loop. RUN the program just to see. Did you get an Out of Data Error? Remember we said that when data has been read once in a program, it can't be read again unless you include RESTORE. Add this line:

```
55 RESTORE
```

and then RUN the program. Now you have an endless loop. (Press BREAK to get out of it.)

Reading String Variables

DATA, READ, and RESTORE can be used with text as well as with numeric values. To use text values, include actual words in your DATA statements, and have the program READ string values (a variable with the suffix $). For instance:

```
10 DATA APPLES, ORANGES, PEARS
20 FOR N = 1 TO 3
30 READ A$
40 PRINT A$
50 NEXT N
60 END
```

Avoiding Out of Data Errors

If you're writing complicated programs with a number of DATA items, we suggest that the last DATA item be "–1" or a similar check value. Then, elsewhere in the program, compare that value to your variables. See the following program for an example.

Using DATA and READ Effectively

DATA and READ are much more powerful than the examples we've used. The following program, which computes the average number of goldfish in the typical American household, illustrates this.

Note two things about this program:

- It includes a "check" at the end of the DATA statement which tells Micro Color BASIC that the end of the data has been reached. This is accom-

plished by making the last item −1, then checking for −1 in a program line.

- It does not compute the number of individual goldfish in each household. You must obtain and type in these values.

```
10 DATA 3,17,18,11,50,12,18,10,2,23
20 DATA 33,81,77,66,32,11,19,18,33,1
30 DATA 25,16,14,13,33,-1
100 TTL = 0  :REM TTL = TOTAL NUMBER
110 READ GF  :REM GF = GOLDFISH
120 IF GF = -1 THEN 150  :REM CHECK FOR END OF
    DATA
130 TTL = TTL + GF  :REM INCREMENT TOTAL
140 GOTO 110  :REM READ SOME MORE
150 AVG = TTL/25 :REM AVG = AVERAGE/ 25 = NUMBER
    OF FAMILIES SURVEYED
160 PRINT "THE AVERAGE FAMILY CONTAINS"; AVG;
    "GOLDFISH."
170 END
```

When you RUN the program, you find out that, of the 25 families surveyed, the average number of goldfish per household was 25.44.

There Has To Be An Easier Way . . .

Both READ and RESTORE can be entered using the Command Function key-combinations. (DATA must be typed in.)

To enter READ, press the key-combination of:

(CONTROL) (T)

To enter RESTORE, press the key-combination of:

(CONTROL) (Y)

13/ ARRAYS

Even though you can use the 26 letters of the alphabet as variables, there may be times when you wish to have more variables available — sometimes hundreds of them — to use as names for different pieces of data. For instance, a car rental agency has a list of automobiles, all of which are the same make and model, but are distinguished by different license plate numbers and colors. Micro Color BASIC uses "arrays" to differentiate between these pieces of data.

Eight cars may be lined up, as in an array. They are all the same except for their license numbers, which vary from 100 to 800, and for the color, which is different. In this case, you can give each of the cars a different letter name, using the variables A through H. But what if the agency had 8,000 cars instead of 8!

Micro Color BASIC provides for a single array which we'll call "A." This isn't the same as the alphabet letter A or the string variable A$. It is a third and totally separate "A." You will recognize it A-sub (something). In other words, you can name the cars as A(1) through A(8), pronounced A sub 1 through A sub 8.

To check this out, type this in (using the Immediate Mode):

```
A = 12 (ENTER)
A$ = "MC-10" (ENTER)
A(1) = 999 (ENTER)
```

Now type:

```
PRINT A; A$; A(1) (ENTER)
```

and your TV will display:

```
 12 MC-10 999
OK
```

One-Dimensional Arrays

A one-dimensional array is like a numbered list.

For instance, store the license numbers of the rental agency's cars in the following DATA line:

```
100 DATA 100,200,300,400,500,600,700,800
```

Now you must create a one-dimensional array to make the data items immediately addressable which is the easiest way to access a specific item in the list. Type in this part of the program:

```
50 FOR L = 1 TO 8  :REM L IS THE LICENSE NUMBER
60 READ A(L)
70 NEXT L
```

RUN the program at this point. Nothing happens except the OK prompt returns. This is because the MC-10 hasn't been told to print the results. In the Immediate Mode, type this:

```
PRINT A(1) (ENTER)
```

and the TV will display 100.

Then type:

```
PRINT A(5) (ENTER)
```

and the Screen will display 500.

The values have been read into the array; you just have to tell the Computer to display the results. You could add to your program eight lines such as:

```
200 PRINT A(1)
201 PRINT A(2)
202 PRINT A(3)
203 PRINT A(4)
204 PRINT A(5)
206 PRINT A(6)
207 PRINT A(7)
208 PRINT A(8)
```

but this can be done much more efficiently by creating another loop:

```
200 FOR N = 1 TO 8
210 PRINT A(N)
220 NEXT N
```

Add these three lines and LIST your program to be sure it looks like this:

```
50 FOR L = 1 TO 8  :REM L IS THE LICENSE NUMBER
60 READ A(L)
70 NEXT L
100 DATA 100,200,300,400,500,600,700,800
```

```
200 FOR N = 1 TO 8  :REM N IS THE NUMBER OF CARS
210 PRINT A(N)
220 NEXT N
```

Now RUN it. Your TV will display:

```
100
200
300
400
500
600
700
800
```

What happened is the DATA elements were read into the array called A(L), but printed out of an array called A(N). The array's name is A. The location of each data element within that array is identified by the number which is placed inside the parentheses.

Remember, through, there is only one array, and its name is A. Its elements are numbered, and called A-sub (number).

Try refining the program a little. Type:

```
170 PRINT
180 PRINT "LICENSE #"
```

It's also possible to include the colors for the cars. For convenience, assign each color a code. (To make it easy, use the same color codes that the CLS statement recognizes.):

Code	For This Color
1	green
2	yellow
3	blue
4	red
5	buff
6	cyan
7	magenta
8	orange

Now modify your program in this manner:

```
180 PRINT "LICENSE #","COLOR CODE"
210 PRINT A(N),N
```

Now you have every license number and color code and you're still not using any of the 26th alphabetic variables.

To make the program even more powerful, erase lines 200, 210, and 220 and type:

```
10 INPUT "WHICH CAR'S LICENSE # DO YOU WISH TO
   KNOW";C  :REM TYPE IN THE COLOR CODE
210 PRINT A(C),C
```

and RUN the program.

The DIM Statement

In most cases, you can use arrays as we've described; but if your array items (that information inside parentheses) are greater than 10, you'll probably want to use the DIM statement. This command reserves just the right amount of memory needed for your array and, with the MC-10, it's important to be conscious of the memory your program requires. To create an array using DIM:

DIM *variable (row, column)*
variable is the name of the array
row and *column* tells what the largest index or "subscript" will be. *row* and *column* can be a numeric variable, constant, or expression. *row* without *column* specifies a one-dimensional array; *row* with *column* specifies a two-dimensional array.

Once an array has been created with DIM, you can access any item in it by putting the appropriate subscript inside the ( ).

To show you how the DIM statement works, we're going to make your car rental agency program even more powerful. When a customer wants to rent a car, you might need to know three things: the color, the license number, and if it is available to be rented. The first two of these have been discussed already. In this section, we'll cover the availability for rental. In the following program, "0" will mean the car is not available for rental; "1" will mean that it can be rented.

Modify your program so it looks like this:

```
10 DIM A(8),B(8)
20 DATA 100,200,300,400,500,600,700,800
30 DATA 1,1,0,0,1,0,1,0
40 FOR L = 1 TO 8  :REM L IS THE LICENSE #
```

```
50 READ A(L)
60 NEXT L
70 FOR R = 1 TO 8  :REM R IS AVAILABILITY FOR
   RENTAL
80 READ B(R)
90 NEXT R
100 INPUT "WHAT COLOR OF CAR DO YOU WANT";C  :REM
    INPUT A COLOR CODE
110 PRINT "LICENSE # "; "COLOR "; "RENTED?"
120 PRINT A(C);"       ";C;"       ";B(C)
130 END
```

In this program, line 10 reserves space for two arrays. If the DATA statements in line 20 and 30 contain more items than allowed for, an error will occur. You'll find it convenient to put the DIM statement early in the program, certainly before the loops start in line 40. Otherwise a ?DD ERROR will occur.

The loop in lines 40-60 reads array A data. The loop in lines 70-90 reads array B data.

The most difficult part of this program for you might be printing out the results. Line 120 does this. Notice that the current value of the variable C is used to define the specified array element in both A and B.

As a car is rented or returned, the rental agent would simply have to go and change the rental status in line 30.

In the language of professional computer types, this is called a matrix.

Two-Dimensional Arrays

For programs that have more data, you might find it more efficient to use two-dimensional arrays by specifying the column option. Take a look at the following program to see how DIM is used in this manner:

```
10 DIM G(12,8)
20 FOR M = 1 TO 12
30 FOR I = 1 TO 8
40 G(M,I)=RND(10)  :REM NUMBER RENTED IS RANDOM
50 NEXT I,M
60 INPUT "WHAT MONTH DO YOU NEED INFORMATION ON";M
    :REM TYPE A NUMBER FROM 1 TO 12
```

```
70 INPUT "WHAT COLOR OF CAR";C  :REM TYPE A NUMBER
   FROM 1 TO 8
80 PRINT "YOU RENTED ";G(M,C); "IN MONTH"; M
90 END
```

This program would supply the rental agent with the number of cars of a certain color that was rented during the specified month (1-12, January-December) of the previous year. Note that in this case, the sample data is "random," supplied by the RND statement. (For a complete description of RND, see the last part of this chapter.) In a real life situation, this might be data that you actually typed into the Computer—data that was processed by the Computer.

Using DIM G(12,8) actually creates a table containing 12 columns and eight rows. If you consider how many entries there are in a 12x8 table, you can see that two-dimensional arrays offer much more storage space than a one-dimensional array.

The CSAVE* Command

The contents of a numeric array can be saved to tape as a separate file. Then, when you need to use that data, you can load it back into the MC-10's memory. CSAVE* is used like this:

CSAVE* *array name, filename*
array name is the name you previously assigned to an array.
filename is a standard Micro Color BASIC filename of eight or fewer characters. *filename* is optional; if omitted, a filename is not assigned.

The specified array must be previously defined (either explicitly by execution of a DIM statement, or implicitly by the assignment of a value to one of the elements 0 through 10).

If filename is not specified, a name is not assigned to the file on tape. For this reason, it is recommended that a filename always be specified when using CSAVE*.

For more details, see CLOAD*.

The CLOAD* Command

Once the contents of a numeric array has been saved to tape with CSAVE*, you can retrieve that data by using the CLOAD* command.

CLOAD* *array name, filename*
array name is the name you previously assigned to an array.
filename is a standard Micro Color BASIC filename of eight or fewer characters. *filename* is optional; if omitted, a filename is not assigned.

The specified array must have been previously defined (either explicitly by execution of a DIM statement, or implicitly by the assignment of a value to one of the elements 0 through 10).

If the specified array is not large enough to hold all the data contained in the cassette file, an OM (Out of Memory) error will result.

If the specified array is larger than necessary to hold the data contained in the cassette file, the data in the file will be loaded starting at the beginning of the array. The values of the extra elements of the array will be unaffected.

For example, if CLOAD* is used to load data into an array of 20 elements from a file which was created by CSAVE* using an array with 10 elements, the first ten elements of the array would contain the data read from the file. The last ten elements would contain the same values they had before CLOAD* was executed.

If a filename is not specified, Micro Color BASIC will load the first file it encounters on the tape.

If an attempt is made to CLOAD* a file of wrong type (one not created with CSAVE*), an FM (File Mode) error will result.

Example

```
CLOAD* A,"DATAFILE"
```

The RND Statement

You've already seen how RND can be used in a program. Basically, RND generates pseudo-random numbers from zero to a specified limit.

RND *(limit)*
limit is the greatest random number which can be generated and is a numeric expression.

Try this example (in the Immediate Mode):

```
N = RND (100) ENTER
PRINT N ENTER
```

RND can also be used with variables:

```
10 A = 1000
20 X = RND(A)
30 PRINT X
40 END
```

Here's a short program that selects random numbers of either 1 or 2 and uses those numbers to simulate flipping a coin.

```
10 X = RND(2)
20 INPUT "CALL IT - HEADS (PRESS 1) OR
   TAILS (PRESS 2)"; COIN
30 IF COIN <> X THEN GOSUB 100
40 IF COIN = X THEN GOSUB 200
50 GOTO 10
100 PRINT "YOU LOSE."
110 RETURN
200 PRINT "YOU WIN."
210 RETURN
```

Notice that you'll have to press BREAK to get out of this endless loop.

There Has To Be An Easier Way . . .

To enter RND without typing in the entire word, type:

CONTROL V

14/ STRING FUNCTIONS

You've already been introduced to how Micro Color BASIC uses the dollar sign (remember string variables?), but the MC-10 also recognizes several statements which have the suffix $.

The INKEY$ Function

Sometimes, when you are running a program, you want the program to pause execution until you "manually" tell it to continue. INKEY$ lets you do this.

At all times during program execution, the MC-10 keeps track of the last key pressed since you typed RUN. It stores this character in a place we'll call the "key latch." If you haven't pressed any keys, then the latch is empty.

INKEY$

The INKEY$ function returns this character as a string value and empties the latch. If a key has not been pressed, INKEY$ returns a "null" string. (BREAK) is the only key INKEY$ will not respond to.

Use INKEY$ like this:

```
10 FOR X = 1 TO 100
20 PRINT X
30 A$ = INKEY$: IF A$ = "" THEN 30
40 NEXT X
50 END
```

Important Note! Be sure there is not a space between the quotation marks in line 30. If a space is there, the string is not a "null" string since it contains a space ("" is right; " " is wrong).

When you RUN this program, it will print the first number and wait for you to press any key except (BREAK) to go on to the next number. If you delete

line 30, the numbers will be displayed so fast you won't be able to read them. INKEY$ slows this down so that you control when they're printed.

Line 30 works like this: You've assigned the INKEY$ function to the string variable A$. The second part of the line says that if A$ is a null string (""), then stay at the current line. If a key has been pressed (that is, if A$ equals a character), then drop down to the next line.

INKEY$ can also be used to test greater than or less than values.

```
10 A$ = INKEY$: IF A$ < > "" THEN 50
20 PRINT "YOU HAVEN'T PRESSED A KEY."
30 GOTO 10
50 PRINT "THE KEY YOU PRESSED IS   ";A$
60 END
```

In this instance, INKEY$ tells Micro Color BASIC to look at the keyboard to see if you have pressed anything. If you haven't pressed a key, the Computer continues to tell you that nothing has been pressed. When you do press a key, it tells you which key you pressed and then stops execution.

If you changed the < > signs to =, the program will immediately loop to line 50 because it is testing for null strings.

There Has To Be An Easier Way . . .

To enter INKEY$ without typing in the entire word, type:

(CONTROL) (P)

The ASC and CHR$ Statements

You've been told before that Computers are dumb machines. They're so dumb, in fact, that they don't even know what the letters A or B are.

Well, if that's so, how does the Computer know that you want the letter A when you press the (A) key? Because the Computer reads everything that is entered from the keyboard in terms of code numbers. Every keyboard character has been assigned a number called the ASCII (pronounced ASK-ee) code.

The letter A, for instance, is code 65. Lowercase A (a) is code 97. (That's 32 greater than the uppercase A. All lowercase letter codes are 32 greater than the uppercase equivalent.)

How can you find out what these codes are? For starters, you can look in Appendix C of this book where you'll find a complete list of all ASCII codes the MC-10 recognizes.

Another way to find out a character's ASCII code is to use the ASC function:

ASC *(string)*
string is a string variable, constant, or expression. Null strings are not allowed.

ASC returns the ASCII code for the first character of the specified string. For instance, in the Immediate Mode type:

```
PRINT ASC ("A") ENTER
```

and the TV will display:

```
 65
OK
```

which is the ASCII code for A. Now type:

```
PRINT ASC ("ALBATROSS") ENTER
```

and the TV will again display:

```
 65
OK
```

since only the first letter of a string is returned.

Here's one way ASC can be used in a program:

```
10 INPUT "TYPE IN A WORD";A$
20 INPUT "TYPE IN A DIFFERENT WORD";B$
30 IF A$ = B$ THEN 20
40 IF A$ < B$ THEN F$ = A$: S$ = B$: GOTO 60
50 F$ = B$: S$ = A$
60 PRINT F$; " PRECEDES"; S$; " ALPHABETICALLY."
70 PRINT: GOTO 10
```

This program uses ASC to read the first letter of the two words you type in, then tells you which of the two has the greater ASC value. The one with the greater value comes later in the alphabet.

The CHR$ Statement

A function you can use in conjunction with ASC is CHR$. This statement is the inverse of ASC. When you enter an ASCII code, CHR$ returns the character the code represents.

CHR$ *(code)*
code is a numeric expression representing an ASCII code between 0 and 255.

In the Immediate Mode, type this:

```
PRINT CHR$(65) (ENTER)
```

and the TV will display:

```
A
OK
```

Now type:

```
PRINT CHR$(155) (ENTER)
```

and the TV will display:

```
■
OK
```

which is a Graphics Character.

CHR$ will let you display characters from a program that aren't normally displayed on the TV. For instance, you can enter a Graphics character directly from the keyboard into a program. To generate a displayable Graphics character whose color is different from cursor color, use CHR$ and specify the character's code. Remember, (SHIFT) (A) etc. will produce graphics character of same color as cursor's.

```
10 FOR X = 150 TO 160
20 PRINT CHR$(X); " ";
30 NEXT X
40 END
```

This will produce the ten Graphics Characters from ASCII codes 150 to 160.

When you have a printer connected to the MC-10, CHR$ can be used to perform certain operations such as backspacing, repeat printing, etc. Use LPRINT CHR$*(code)* if you are performing specific operations with the printer. See your printer owner's manual for details on codes the printer uses.

The STR$ Statement

The STR$ function converts a number to a string value.

STR$ *(number)*
number is a numeric value.

Normally, Micro Color BASIC will return a Type Mismatch Error if you try to use a number as a string value. However, STR$ lets you get around this problem.

```
10 INPUT "TYPE A NUMBER";N
20 A$ = STR$(N)
30 PRINT A$; " IS NOW A STRING."
40 END
```

The LEN Function

The LEN (length) statement returns the number of characters (letters, numbers, and spaces) in a string.

LEN *(string)*
string is a string variable, constant, or expression.

To see how LEN works, type in this short program:

```
10 INPUT "TYPE IN THREE SHORT WORDS";A$
20 PRINT "WHAT YOU TYPED IN USES"; LEN(A$);
   " SPACES"
30 END
```

Notice that the spaces in between the words are counted as well.

A function like LEN can be very important with a limited memory Computer like the MC-10. You can use it to limit how many characters (hence, how much memory) are entered into a line:

```
10 INPUT "USE ONLY 3 LETTER ABBREVIATIONS FOR
   MONTH";M$
20 IF LEN(M$) > 3 THEN 10
30 PRINT "THE MONTH IS ";M$
40 END
```

LEFT$, MID$, and RIGHT$

Micro Color BASIC has three functions that examine the parts of a string: LEFT$ (which gets the left portion of the string); MID$ (which gets the middle portion); and RIGHT$ (which gets the right portion).

LEFT$ *(string, length)*
string is a string variable, constant, or expression
length tells how many characters to look for in *string*.

RIGHT$ *(string, length)*
string is a string variable, constant, or expression
length tells how many characters to look for in *string*.

MID$ *(string, position, length)*
string is a string variable, constant, or expression
position tells where the mid-portion starts and can be any number from 1 (the first character in the string) up to the total length of the *string*.
length tells how many characters to look for in *string*.

Use LEFT$ and RIGHT$ first.

```
10 INPUT "TYPE A WORD";W$
20 PRINT "THE FIRST LETTER IS "; LEFT$(W$,1)
30 PRINT "THE LAST TWO LETTERS ARE "; RIGHT$(W$,2)
```

For this example, type in WORD (ENTER) when prompted. The TV will then display:

```
THE FIRST LETTER IS W
THE LAST TWO LETTERS ARE RD
```

MID$, on the other hand, checks the middle portion of a string after you tell it where to start counting. Add this line to the above program.

```
40 PRINT "THE MID PART OF YOUR WORD IS
   ";MID$(W$,2,2)
```

Now when you RUN the program and type in WORD, the TV will display:

```
THE FIRST LETTER IS W
```

```
THE LAST TWO LETTERS ARE RD
THE MID PART OF YOUR WORD IS OR
```

Line 20 uses LEFT$ to specify the word (string) you want to examine and how many letters in that word you want to look at. Line 30 uses RIGHT$ for the same purpose. Line 40 uses MID$ to specify the word, which part of the word to start counting at, and how many letters to count.

Here's a practical example of three of the four string functions at work:

```
10 A$ = "CHANGE A SENTENCE."
20 B$ = "IT'S EASY TO "
30 C$ = B$ + " " + A$
40 PRINT C$
50 Y = LEN ("A SENTENCE")
60 FOR X = 1 TO LEN(C$)
70 IF MID$(C$,X,Y) = "A SENTENCE" THEN 90
80 NEXT X
85 END
90 D$ = LEFT$(C$,X-1)
100 E$=D$ + "ANYTHING YOU WANT"
110 PRINT E$
```

The CLEAR Statement

When using strings, DATA statements, or creating arrays with Micro Color BASIC programs, you'll sometimes need to reserve memory at the beginning of your program. CLEAR does this for you.

CLEAR *amount*

amount is the amount of memory that you need to reserve for your program.

When you do not use the CLEAR statement, the MC-10 assumes you will need only enough string space to store 100 characters. Unless you change this with CLEAR, you'll get an Out of String space error when you try to store more than 100 characters.

```
CLEAR
```

will reserve 100 character spaces for your strings.

```
CLEAR 500
```

will reserve 500 characters.

15/ GRAPHICS

It's possible to create graphic displays on the TV screen using the Micro Color BASIC statements SET and RESET and the CHR$ function you were introduced to in the last chapter. SET "sets" a graphic character ON, RESET "reset" that character to its original state (turns it OFF).

Before using these statements, you must think of the TV screen as being divided up into a grid that is 64 blocks horizontally (0 to 63) and 32 blocks vertical (0 to 31).

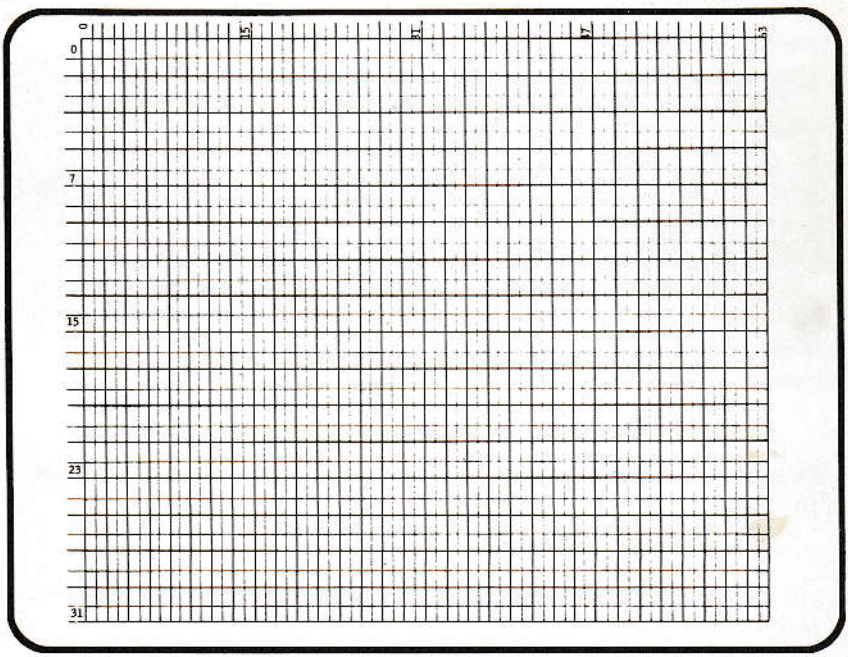


Figure 19. The MC-10 Graphics Grid

The Graphics Character

The Graphic cell is the space required by any single Graphic character you display.

The Graphics Character is made up of four "blocks." Your Computer has three statements that let you specify or address blocks.

The three keywords are:

- SET
- RESET
- POINT

Actually, you've already seen the Graphics block. We described Micro Color BASIC graphics characters earlier in the **"Operation"** section of this manual.

The SET and RESET Statements

The SET statement lets you "set" one of the blocks.

SET *(x,y color)*
x is a position on the X-axis (horizontal) and is a numeric expression between 0 and 63.
y is a position on the Y-axis (vertical) and is a numeric expression between 0 and 31.
color is the background color of the Graphics Character you want and is a numeric expression between 0 and 8.

In the Immediate Mode, change the color of the TV screen to black by typing:

```
CLS 0 (ENTER)
```

This is simply for convenience so you can see the true shape of the block. Then type:

```
SET (31,16,1) (ENTER)
```

and a green Graphics Character, on a black background, will appear in the center of the TV screen.

Now type:

```
SET (31,16,2) (ENTER)
```

and the same character, but on a black background, will appear.

RESET, on the other hand, turns the block off after it has been turned on.

RESET *(x,y)*
x is a position on the X-axis (horizontal) and is a numeric expression between 0 and 63.
y is a position on the Y-axis (vertical) and is a numeric expression between 0 and 31.

RESET the block created above by typing:

```
RESET (31,16) ENTER
```

Now the yellow block is reset to black.

The following program will change the TV screen to black and create a blue PRINT cell on the right-side of the Screen. You can use the backspace key-combination ((CONTROL) (A)) to move the blue block to the left-side.

```
10 CLS 0
20 H = 63
30 SET (H,14,3)  :REM TURNS BLOCK ON
40 A$ = INKEY$
50 IF A$ = CHR$(8) THEN 70  :REM CHR$(8) IS
   BACKSPACE
60 GOTO 40
70 H = H -1
80 IF H < 0 THEN END  :REM PROGRAM WILL END WHEN
    THE LEFT MARGIN IS REACHED
90 SET (H,14,3)
100 RESET (H+1,14)  :REM ERASES BLOCK
110 GOTO 40
```

Each time you press the combination of (CONTROL) (A), the blue block moves one space to the left.

These examples tell you two things about the four parts of a Graphics Character:

- They must be all one color (such as black or blue), or
- They must be one color and black.

There Has To Be An Easier Way . . .

To use SET without typing it in, type:

(CONTROL) (E)

To use RESET without typing it in, type:

(CONTROL) (R)

The POINT Function

POINT is a function that returns a value based upon the current status of a Graphics block.

POINT *(x,y)*
x is a position on the X-axis (horizontal) and is a numeric expression between 0 and 63.
y is a position on the Y-axis (vertical) and is a numeric expression between 0 and 31.

POINT returns the following values:

Value Returned	Meaning
−1	Block is character mode.
0	Block is turned OFF.
color code	Block is turned ON using the indicated color code. (See CLS for a list of color codes).

The following program will randomly turn on blocks in a area bounded by X = 20, Y = 10 to the color Blue — color code 3. Each time another block is turned on, POINT tests a specific location, X = 10, Y = 10, to see if it is set to blue. If the block isn't blue, another block is randomly set. When the specified block is set to blue, program control drops down to line 100 and prints the message.

```
10 X = RND (20)
20 Y = RND (10)
30 SET (X,Y,3)  :REM 3 IS THE COLOR CODE FOR BLUE
40 IF POINT (10,10) = 3 THEN 100
```

```
50 GOTO 10        :REM AND SET ANOTHER BLOCK
100 PRINT "10,10 IS SET TO BLUE"
110 END
```

RUN the program. Since the blocks are turned on in a random manner, the message will be printed almost at once in some instances. At other times, it may take a while.

16/ MATH, TRIG, AND OTHER FUNCTIONS

In addition to the simple math functions described earlier in this book, the MC-10 repertoire includes several advanced functions.

We'll give you a brief rundown of each function and show you how you can use it to perform any number of higher-level mathematical operations.

Before we do, however, there are a couple of functions and definitions you should be familiar with.

Note: Before the results of a math function can be displayed on the Screen, they must be prefaced by a PRINT statement.

Exponentiation (↑)

Anytime you want to raise a number to the nth power, follow this format:

number ↑ *power*
number specifies the number you wish to raise to power. It may be any numeric expression.
↑ is generated by pressing (CONTROL) (W).
power is the exponent to which number is raised. It may be any numeric expression.

Exponentiation has precedence over other operators. For example, if you try −2↑2, the result will be a negative number. To raise −2 to the 2nd power "correctly" (resulting in a positive number), enclose −2 in parentheses.

Start with a hard example—77 cubed. Try this:

```
PRINT 77 ↑ 3 (ENTER)
```

and your Screen should display:

```
 456533.002
OK
```

Try taking 10 to the 10th power and your Screen will display:

```
 1.0000000E+10
OK
```

Since 10,000,000,000 has more than nine significant digits, the Computer went into E (scientific) notation.

How about 100 to the 100th power? This will produce an Overflow Error message. This means the answer was too big for the MC-10 to handle. The MC-10 was designed to handle any number from -10^{38} to 10^{38}.

The SQR Function

SQR enables you to find the square root of a number.

SQR *(number)*
number is a numeric expression not less than zero.

For example, if you want the square root of 100, type:

```
PRINT SQR (100) ENTER
```

and you'll find out that the answer is 10.

There Has To Be An Easier Way . . .

If you need to enter the SQR function without typing it in, press the key-combination of:

CONTROL ?

The ABS Function

ABS returns the absolute value of a number.

ABS *(number)*
number is a numeric expression.

RUN this short program to see how it works:

```
10 INPUT "TYPE A NUMBER";N
20 PRINT "THE ABSOLUTE VALUE IS"; ABS(N)
30 END
```

There Has To Be An Easier Way . . .

If you need to enter the ABS function without typing it in, press the key-combination of:

(CONTROL) (B)

The SGN Function

SGN tells you whether a number is positive, negative, or has a value of zero.

SGN *(number)*
number is a numeric expression.

Try this short example:

```
10 INPUT "TYPE A NUMBER";X
20 IF SGN(X) = 1 THEN PRINT "POSITIVE"
30 IF SGN(X) = 0 THEN PRINT "ZERO"
40 IF SGN(X) = -1 THEN PRINT "NEGATIVE"
50 END
```

There Has To Be An Easier Way . . .

If you need to enter the SGN function without typing it in, press the key-combination of:

(CONTROL) (X)

The INT Function

The INT function tells the MC-10 to return the largest whole number not exceeding the given number.

INT *(number)*
number is a numeric expression.

Try this program to see INT at work:

```
10 INPUT "TYPE A NUMBER THAT HAS A DECIMAL
   FRACTION";N
```

```
20 PRINT "THE WHOLE NUMBER PORTION OF "; N; "IS ";
   INT(N)
30 END
```

There Has To Be An Easier Way . . .

If you need to enter the INT function without typing it in, press the key-combination of:

(CONTROL) (C)

Trig Functions

The trigonometry functions that Micro Color BASIC has have many practical applications. For instance, imagine the triangle you'll be working with in this chapter is actually the roof of a house you're building. These functions can help you determine either the length of the rafters or the slope of the roof. Notice we've labelled angles with the prefix A (angle A = AA, etc.). To distinguish between angles and sides, we've tagged the sides opposite their respective angles with the prefix S (for example, SA is opposite angle AA, etc.).

In trigonometry, SB is called "adjacent," side SA "opposite," and side SC "hypothenuse."

Using this triangle, we can define the common trig functions in the following manner:

Sine of AA = SIN (AA) = Opposite/Hypothenuse = SA/SC
Cosine of AA = COS (AA) = Adjacent/Hypothenuse = SB/SC
Tangent of AA TAN (AA) = Opposite/Adjacent = SA/SB

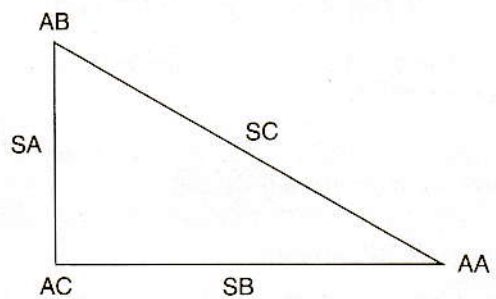


Figure 20. Sample Triangle

Degrees vs. Radians

To define an angle, you may use either of two units of measurement. While the most common unit is the degree, another "more technical" unit is the radian.

Your Computer assumes all angles are measured in radians. Since radians may be somewhat alien to you, you can convert them to degrees (and vice versa) this way:

Degrees to Radians: Degrees / 57.29577951
Radians to Degrees: Radians * 57.29577951

In the sample programs that follow, we've included a "converter." This allows you to input degrees and the Computer automatically converts them to radians (and vice versa for some purposes).

The SIN Function

The syntax for SIN is:

SIN *(angle)*
angle is a numeric expression for an angle in radians.

One way to use SIN is to determine the two unknown sides of a triangle if two angles and the third side are known. Enter and run the following program. You can input whatever values you want.

```
5 CLS
10 INPUT "WHAT IS ANGLE A (AA)";AA : IF AA=< 0 OR
    AA >= 180 THEN 100
20 INPUT "WHAT IS ANGLE B (AB)";AB : IF AB=< 0 OR
    AB >= 180 THEN 100
30 INPUT "WHAT IS SIDE C (SC)";SC : IF SC =< 0
    THEN 100
40 AC = 180 - (AA+AB)  :REM VALUE OF ANGLE·AC
50 IF (AA + AB + AC) < > 180 THEN 100
60 AA = AA/57.29577951: AB = AB/57.29577951:
    AC = AC/57.29577951  :REM CONVERT DEGREES TO
    RADIAN
70 SA = ((SIN (AA))/(SIN(AC)))*SC: IF SA < 0 THEN
    100
```

```
80 SB = ((SIN (AB))/(SIN(AC))) * SC: IF SB < 0
   THEN 100
90 PRINT "SIDE A (SA) IS" SA " LONG": PRINT "SIDE
   B (SB) IS" SB " LONG.": GOTO 10
100 PRINT "SORRY, NOT A TRIANGLE. TRY AGAIN" :
    GOTO 10
```

When the Computer prompts you for angles AB and AC, you should input the degrees-measures of the angles. If you try to use negative angles or angles greater than or equal to 180 degrees, the Computer will go to line 100, print the message, and re-prompt you. If you attempt to input a negative number for side SC, you'll have the same result.

Since you don't know the size of angle AC, the Computer automatically computes this in line 40. If the sum of the three angles is not equal to 180 degrees, the Computer takes appropriate action in line 50. Line 60 converts "your" degrees to the "Computer's" radians so the sine calculations may be done.

There Has To Be An Easier Way . . .

If you need to enter the SIN function without typing it in, press the key-combination of:

(CONTROL) (N)

The COS Statement

The cosine function is related to the sine function and uses the following syntax:

COS *(angle)*
angle is a numeric expression for an angle in radians.

One application of cosine is to determine the length of a triangle's third side if you already know two sides and an angle. For angle and side identification, refer to the sample triangle.

```
5 CLS
10 INPUT "WHAT IS ANGLE C (AC)";AC : IF AC=< 0 OR
   AC >= 180 THEN 100
20 AC = AC/57.29577951  :REM CONVERT DEGREES TO
   RADIAN
```

```
30 INPUT "WHAT IS SIDE A (SA)";SA : IF SA =< 0
   THEN 100
40 INPUT "WHAT IS SIDE B (SB)";SB : IF SB =< 0
   THEN 100
50 SC = ((SA ^ 2) + (SB ^ 2))-
   (2*(SA*SB*COS(AC))): IF SC < 0 THEN 100
60 PRINT "SIDE C (SC) IS" SQR (SC)" LONG": GOTO 10
100 PRINT "SORRY, NOT A TRIANGLE. TRY AGAIN" :
    GOTO 10
```

Notice the program works almost the same as the SIN program except for the use of exponentiation (^) in line 50 and SQR in line 60.

There Has To Be An Easier Way . . .

If you need to enter the COS function without typing it in, press the key-combination of:

(CONTROL) (M)

The TAN Function

The third trigonometric function you can use with your MC-10 is TAN. This function allows you to calculate the tangent of an angle.

TAN *(angle)*
angle is a numeric expression for an angle in radians.

Among other things, the tangent function can be used to determine the unknown length of one side of a triangle if you know the dimensions of another side and one angle.

```
5 CLS
10 INPUT "WHAT IS SIDE B (SB)"; SB: IF SB =< 0
   THEN 100
20 INPUT "WHAT IS ANGLE A (AA)";AA : IF AA=< 0 OR
   AA >= 180 THEN 100
30 AA = AA/57.29577951  :REM CONVERT DEGREES TO
   RADIAN
40 SC = SB * (TAN (AA)): IF SC =< 0 THEN 100
50 PRINT "SIDE C (SC) IS" SC" LONG": GOTO 10
100 PRINT "SORRY, NOT A TRIANGLE. TRY AGAIN" :
    GOTO 10
```

The key to this program, of course, is line 40, where the tangent of angle AA is multiplied by the length of side SB to determine the length of side SC.

There Has To Be An Easier Way . . .

If you need to enter the TAN function without typing it in, press the key-combination of:

(CONTROL) (.)

The LOG Function

LOG returns the natural logarithm of a number. This is the inverse of EXP, so X = LOG(EXP(X)).

LOG *(number)*
number is a numeric expression greater than zero.

The logarithm of a number is the power to which a given base must be raised to result in the number. Logs are useful in scientific and mathematical problems. In the LOG function, the base is e = 2.718271828.

To find the logarithm of a number to another base B, use the formula log base B (x) = log e (x)/log e (B). For example, LOG(32768)/LOG(2) returns the logarithm to base 2 for 32768 (that is, the power to which 2 is raised to get 32768).

Try these:

```
PRINT LOG (1) (ENTER)
PRINT LOG (100) (ENTER)
PRINT LOG (2.718271828) (ENTER)
```

There Has To Be An Easier Way . . .

If you need to enter the LOG function without typing it in, press the key-combination of:

(CONTROL) (.)

The VAL Function

The VAL function will convert a string value into a numeric value.

VAL *(string)*
string is a string variable, constant, or expression.

VAL accepts a string argument and evaluates it as a number. If the string characters don't make sense in a number, it returns a zero.

This function is useful for checking to see if answers are correct. How does it do this? RUN the following program to see:

```
10 X = RND(4)
20 Y = RND(4)
30 PRINT "WHAT IS "; X; "+"; Y
40 T = 0
50 A$ = INKEY$
60 T = T + 1
70 SOUND 128,1
80 IF T = 200 THEN 200
90 IF A$ = "" THEN 50
100 IF VAL(A$) = X + Y THEN 130
110 PRINT "WRONG"; X; "+"; Y; "="; X+Y
120 GOTO 10
130 PRINT "CORRECT"
140 GOTO 10
200 CLS 7
```

When you RUN this program, you'll have a few seconds to type in the answer to the arithmetic problem the Computer poses you with. Line 100 converts A$ into a numeric VALue. If A$ equals the string "5," VAL(A$) equals the number 5.

The EXP Function

The EXP function returns the natural exponential of a number, the e^{number}. This function is the inverse of LOG; therefore, X = EXP (LOG(X)).

EXP *(number)*
number is a numeric expression less than 88.

RUN this program to see EXP at work.

```
10 CLS
20 INPUT "ENTER X"; X
30 PRINT "EXP(X)="; EXP (X)
40 GOTO 20
```

Part III
Appendix

Appendix A/ Sample Programs

Music Composer

```
10 INPUT "LENGTH(1-10)";M
20 M = M*4
30 INPUT "TEMPO (1-4)"; T1
40 IF T1 = 4 THEN 60
50 T = T1 : GOTO 70
60 T = 8
70 FOR K = 1 TO M*8
80 GOSUB 1000
90 B = RND(3) * T
100 SOUND P, B
110 CLS(S)
120 NEXT K
130 IF RND(10) <=8 THEN 150
140 SOUND 125, 16*T
145 END
150 SOUND 90, 16*T
160 END
1000 X = RND(100)
1010 IF X <= 20 AND X <= 25 THEN P = 90 : S = 1
1020 IF X > 21 AND X <= 25 THEN P = 108 : S = 2
1030 IF X > 25 AND X <= 40 THEN P = 125 : S = 3
1040 IF X > 40 AND X <= 55 THEN P = 133 : S = 4
1050 IF X > 55 AND X <= 75 THEN P = 147 : S = 5
1060 IF X > 75 AND X <= 85 THEN P = 159 : S = 6
1070 IF X > 85 AND X <= 95 THEN P = 176 : S = 7
1080 IF X > 95 THEN P = 58 : S = 8
1090 RETURN
```

Crap Game

```
10 CLS
20 A = RND(6)
30 B = RND(6)
40 R = A + B
50 PRINT @ 200, A
```

```
60 PRINT @ 214, B
70 PRINT @ 394, "YOU ROLLED A"; R
80 IF R = 2 THEN 500
90 IF R = 3 THEN 500
100 IF R = 12 THEN 500
110 IF R = 7 THEN 600
120 IF R = 11 THEN 600
130 FOR X = 1 TO 800
140 NEXT X
150 CLS
160 PRINT @ 195, "ROLL ANOTHER" R "AND YOU WIN"
170 PRINT @ 262, "ROLL A 7 AND YOU LOSE"
180 PRINT @ 420, "PRESS <ENTER> WHEN READY"
185 PRINT @ 456, "FOR YOUR NEXT ROLL"
190 INPUT A$
200 X = RND(6)
210 Y = RND(6)
220 Z = X + Y
225 CLS
230 PRINT @ 200, X
240 PRINT @ 214, Y
250 PRINT @ 394, "YOU ROLLED A"; Z
260 IF Z = R THEN 500
270 IF Z = 7 THEN 600
280 GOTO 100
500 FOR X = 1 TO 1000
510 NEXT X
515 CLS
520 PRINT @ 230, "YOU'RE THE WINNER"
530 PRINT @ 294, "CONGRATULATIONS!!!"
540 GOTO 630
600 FOR X = 1 TO 1000
610 NEXT X
615 CLS
620 PRINT @ 264, "SORRY YOU LOSE"
630 PRINT @ 458, "GAME'S OVER"
```

Table of Squares

```
5 CLS
7 PRINT @ 38, "TABLE OF SQUARES"
8 PRINT
10 P = 2
```

```
20 FOR N = 2 TO 10
25 GOSUB 2000
30 PRINT N "*" N "=" E,
40 NEXT N
50 END
2000 REM FORMULA FOR RAISING A NUMBER TO A POWER
2010 E = 1
2020 FOR X = 1 TO P
2030 E = E * N
2040 NEXT X
2050 IF P = 0 THEN E = 1
2060 RETURN
```

Computing Interest Earnings for a Savings Account

```
10 INPUT "YOUR MONTHLY DEPOSIT";D
20 INPUT "BANK'S ANNUAL INTEREST RATE";I
30 I = I/12 * .01
40 INPUT "NUMBER OF DEPOSITS";P
50 GOSUB 1000
60 PRINT "YOU WILL HAVE $" FV " IN " P " MONTHS"
70 END
1000 REM: COMPOUND MONTHLY INTEREST FORMULA
1010 N = I + 1
1020 GOSUB 2000
1030 FV = D * ((E-1)/I)
1040 RETURN
2000 REM: FORMULA FOR RAISING A NUMBER TO A POWER
2010 E = 1
2020 FOR X = 1 TO P
2030 E = E * N
2040 NEXT X
2050 IF P = 0 THEN E = 1
2060 RETURN
```

Kaleidoscope

```
10 CLS0
20 X=RND(32)-1
30 Y=RND(16)-1
40 Z=RND(9)-1
50 GOSUB90
```

```
60 GOTO20
90 IFZ=0 OR RND(7)=3THEN150
100 SET(31-X,16+Y,Z)
110 SET(31-X,15+Y,Z)
120 SET(32+X,16+Y,Z)
130 SET(32+X,15-Y,Z)
140 RETURN
150 RESET(31-X,16+Y)
160 RESET(31-X,15-Y)
170 RESET(32+X,16+Y)
180 RESET(32+X,15-Y)
190 RETURN
```

Appendix B/ Error Messages

Message	Meaning
NF	NEXT without FOR
SN	Syntax Error
RG	RETURN without GOSUB
OD	Out of Data
FC	Illegal Function Call
OV	Overflow
OM	Out of Memory
UL	Undefined Line
BS	Bad Subscript
DD	Double Dimensioned Array
/0	Division by 0
ID	Illegal Direct Statement
TM	Type Mismatch
OS	Out of String Space
LS	Long String (exceeds 255 characters)
ST	String Formula Too Complex
CN	Can't Continue
IO	Input/Output Error
FM	File Mode Error

Appendix C/ Codes and Tables

ASCII CHARACTER CODES

Key	All-CAPS Mode		Upper/lowercase Mode		
	Unshifted	Shifted	Unshifted	Shifted	CONTROL
1	49	33	49	33	142
2	50	34	50	34	147
3	51	35	51	35	152
4	52	36	52	36	151
5	53	37	53	37	150
6	54	38	54	38	148
7	55	39	55	39	149
8	56	40	56	40	157
9	57	41	57	41	134
0	48	—	48	—	—
A	65	233	97	65	8
B	66	224	98	66	179
C	67	226	99	67	178
D	68	231	100	68	130
E	69	237	101	69	155
F	70	230	102	70	144
G	71	229	103	71	132
H	72	72	104	72	163
I	73	73	105	73	138
J	74	74	106	74	129
K	75	75	107	75	158
L	76	76	108	76	188
M	77	77	109	77	186
N	78	78	110	78	185
O	79	79	111	79	165
P	80	80	112	80	199
Q	81	239	113	81	21
R	82	236	114	82	156

*This table specifies decimal values only.

Key	All-CAPS Mode		Upper/lowercase Mode		
	Unshifted	Shifted	Unshifted	Shifted	CONTROL
S	83	232	115	83	9
T	84	235	116	84	140
U	85	85	117	85	128
V	86	225	118	86	181
W	87	238	119	87	94
X	88	227	120	88	177
Y	89	234	121	89	143
Z	90	228	122	90	10
:	58	42	58	42	137
-	45	61	45	61	145
@	64	—	64	19	136
;	59	43	59	43	146
,	44	60	44	60	187
.	46	62	46	62	183
/	47	63	47	63	182
!	—	33	—	33	142
"	—	34	—	34	147
#	—	35	—	35	152
$	—	36	—	36	151
%	—	37	—	37	150
&	—	38	—	38	148
'	—	39	—	39	149
(	—	40	—	40	157
)	—	41	—	41	134
*	—	42	—	42	137
=	—	61	—	61	145
+	—	43	—	43	146
<	—	60	—	60	187
>	—	62	—	62	183
?	—	63	—	63	182
BREAK	03	03	03	03	—
ENTER	13	13	13	13	—
SPACEBAR	32	32	32	32	—

KEYBOARD CHARACTERS

CHARACTER	PURPOSE
(ENTER)	Tells Computer you've reached the end of your program line or command line.
(BREAK)	Stops execution of your program.
(SHIFT) (@)	Pauses execution of your program. Press any key to continue.
(SHIFT) (0)	Switches Computer to and from upper/lower case mode.

BASIC SYMBOLS

SYMBOL	EXPLANATION
" "	Indicates that the data in quotes is a constant.
:	Separates program "statements" on the same line.
( )	Tells the Computer to perform the operation in the inside parenthesis first.
;	Causes constants and variables to be PRINTED right next to each other.

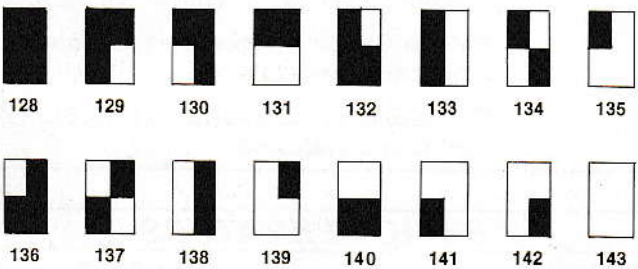
BASIC OPERATORS

OPERATOR	PURPOSE
+	Combines strings
+	Addition
−	Subtraction
*	Multiplication
/	Division
=	Equals
>	Greater Than
>= or =>	Greater than or equal to
<= or =<	Less than or equal to
<	Less than
<> or ><	Not equal to
AND	Logical AND
OR	Logical OR
NOT	Logical NOT

Code	Color
0	Black
1	Green
2	Yellow
3	Blue
4	Red
5	Buff
6	Cyan
7	Magenta
8	Orange

GRAPHICS CHARACTERS

These are the codes for the Micro Color Computer's graphics characters. To produce them, use CHR$ with the character's code. For example, PRINT CHR$ (129) produces character 129.



To create these characters in one of the colors below, add the appropriate number to the code. For example, PRINT CHR$ (129 + 16) produces character 129, except the green area is yellow.

+16 — yellow
+32 — blue
+48 — red
+64 — buff
+80 — cyan
+96 — magenta
+112 — orange

Appendix D/ Troubleshooting and Maintenance

If you have problems operating your TRS-80® MC-10, check the following table of symptoms. Hopefully, you'll find the cure as well.

Symptom	Cure
The OK prompt doesn't appear when you turn the Computer on.	1. No AC power. Check the power cord connection. 2. Incorrect power-up sequence. All accessories should be turned ON before you turn the Computer ON. 3. Accessory device is not connected properly. Check connections. 4. Your TV screen needs adjustment. Check the Contrast, Brightness, or fine tuning controls. 5. Antenna Switch should be set to COMPUTER, not TV.
Poor reception or fuzzy display.	1. Be sure TV is set on the proper channel (3 or 4—whichever gives the best display). 2. Check the antenna connections to be sure they are connected both securely and properly.
Your cassette program won't load.	1. Improper cassette connections. Check connection instructions in the cassette owner's manual. 2. Cassette volume is too low or high. Check the recorder's volume control. 3. Information on tape may have been garbled due to electricity discharge, magnetic field, or tape deterioration. Try to load the duplicate copy, if you have one.

Symptom	Cure
The Computer "hangs up" during normal operations, requiring RESET or Power-off/on.	1. Fluctuations in the AC supply. See AC Power Sources. 2. Defective or improperly installed connector. Check all connection cables to see that they are securely attached and that they are not frayed or broken. 3. Programming error. Re-check the program.
"Ghosts" or mixed Computer and TV reception.	Try using the other TV channel (3 or 4).

AC Power Sources

Computers are sensitive to fluctuations in the power supply at the wall socket. This is rarely a problem unless you are operating in the vicinity of heavy electrical machinery. The power source may also be unstable if some appliance or office machine in the vicinity has a defective switch which arcs when turned on or off.

If the fluctuations are severe, you may need to take some or all of the following steps:

- Install bypass or isolation devices in the problem-causing devices.
- Repair or replace any defective (arcing) switches on lights or appliances.
- Install a separate power-line for the Computer.
- Install a special line filter designed for computers and other sensitive electronic equipment.

Power line problems are rare and many times can be prevented by proper choice of installation location. The more complex the system and the more serious the application, the more consideration you should give to providing an ideal power source for your Computer.

Maintenance

Your MC-10 requires little maintenance. It's a good idea to keep it clean and free of dust build-up. This is especially important for the keyboard.

If you need to clean the Computer case, use a damp, lint-free cloth.

The accessory devices (cassette recorder, printer, etc.) may require more maintenance. Check the owner's manual for each accessory in your system.

Color Adjustment Test

For the purpose of adjustment, as well as example, the following program will give you a chance to view the available spectrum.

Start Micro Color BASIC and type in the program. Don't worry about spacing, but type everything else exactly as listed below:

NEW (ENTER)
5 FOR X=0 TO 63 (ENTER)
10 FOR Y=0 TO 31 (ENTER)
15 C=INT(X/8+1) (ENTER)
20 SET (X,Y,C) (ENTER)
25 NEXT Y,X (ENTER)
30 GOTO 30 (ENTER)

To check for typing errors, list the program by typing: `LIST` (ENTER).

It should look like this:

```
5 FOR X = 0 TO 63
10 FOR Y = 0 TO 31
15 C = INT(X/8+1)
20 SET (X,Y,C)
25 NEXT Y,X
30 GOTO 30
```

If you find any mistakes, simply re-type the line, then type `RUN` (ENTER).

Your TV should display eight color-bars (in the order of green, yellow, blue, red, buff, cyan, magenta, and orange) on a black background.

You can now use your TV color adjustment control to adjust the colors to their proper hue.

Press (BREAK) to stop the program.

Use the preceding test to adjust your TV for the best possible spread of colors. But remember, the range of adjustment will vary, depending on the condition and quality of your TV set.

Video Centering Test

Type in the following to insure your TV display is centered correctly on the screen.

NEW (ENTER)

10 CLS (ENTER)
15 FOR X=0 TO 63 (ENTER)
20 Y=15 (ENTER)
25 RESET (X,Y) (ENTER)
30 NEXT X (ENTER)
35 FOR Y=0 TO 31 (ENTER)
40 X=31 (ENTER)
45 RESET (X,Y) (ENTER)
50 NEXT Y (ENTER)
55 GOTO 55 (ENTER)

To check for typing errors, type: LIST (ENTER)

It should look like this:

```
10 CLS
15 FOR X = 0 TO 63
20 Y = 15
25 RESET (X,Y)
30 NEXT X
35 FOR Y = 0 TO 31
40 X = 31
45 RESET (X,Y)
50 NEXT Y
55 GOTO 55
```

If you find any mistakes, simply re-type the line, then type RUN (ENTER).

You should have a green background, surrounded by black, with a horizontal and vertical line (both black) intersecting in the center of the screen. You can now adjust your set until the display is centered.

Press (BREAK) to stop the program.

The MC-10 generates a rectangular image designed to fill most of the TV's screen. Use the horizontal and vertical centering controls on your TV to center this image as much as possible.

Don't worry if you can't get a perfectly centered image, or if you notice a slight distortion in certain areas of your TV screen. These minor variations depend on the condition of your TV set. If they are severe, consult a qualified TV service technician.

Sound Demonstration

The following program will output the entire frequency range (codes 0-255) Micro Color BASIC can produce.

NEW (ENTER)
10 FOR X=1 TO 255 (ENTER)
20 SOUND X,1 (ENTER)
30 NEXT X (ENTER)

To check for typing errors, type: LIST (ENTER)

It should look like this:

```
10 FOR X = 1 TO 255
20 SOUND X,1
30 NEXT X
```

If you find any errors, simply re-type the line. Set your TV volume to a normal level, and type: RUN (ENTER).

Appendix E/ Specifications

AC Power Adapter (required)

Catalog Number	26-3011
Power Input	120V AC 60Hz 16W
Power Output	8.0V AC 1.5A AC

Microprocessor

Type	6803

Physical Dimensions

Size	2" (H) x 8½" (L) x 7" (W) (51mm x 216mm x 178mm)
Weight	29½ ozs. (836.32 grams)

Temperature

Operating	41° to 104° (Fahrenheit) 5° to 40° (Centigrade)
Storage	−40° to 160° −40 to 71°

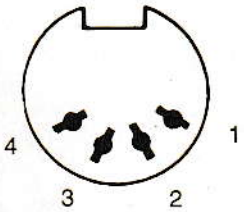
Humidity

Operating	40% to 80%
Storage	20% to 90%

Serial Interface

RS-232-C Signal	Pin #
CD Carrier Detect	1
RD Receive Data	2
GND Ground	3
TD Transmit Data	4

RS-232C Pin Description



Printer Software Requirements

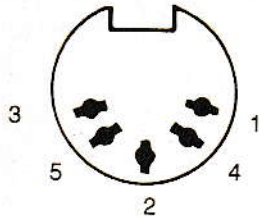
600 baud
1 Start Bit
7 Data Bits
2 Stop Bits
No parity
132-Column printer width
Automatic Carriage return at end of line

Cassette Interface

Suggested Input Level for Playback from Recorder
1 to 5 V peak-to-peak at a minimum impedance of 220 ohms

Typical Computer Output Level to Recorder
800 mV peak-to-peak at 1K ohms

Cassette Pin Description



1. Remote Ground
2. Signal Ground
3. Remote Control
4. Input from Recorder's EARphone Jack
5. Output to Recorder's AUX or MIC Jack

Appendix F/ Micro Color BASIC Language Reference Summary

Word	Purpose	Example
ABS	Computes absolute value.	`Y=ABS(5)`
ASC	Returns ASCII code of first character of specified string.	`A=ASC(T$)`
CHR$	Returns character for ASCII, control, or graphics code. P$ = CHR$(T)	`? CHR$(191)`
CLEAR	Reserves bytes of string storage space. Initializes variables.	`CLEAR` `CLEAR 500`
CLOAD	Loads specified program file from cassette. If filename is not specified, first file encountered is loaded. Filename must be eight character spaces or less.	`CLOAD` `CLOAD"PROGRAM"`
CLOAD*	Loads numeric data into an array from a cassette file which has been created using the CSAVE* statement.	`CLOAD*A,DATAFILE`
CLS	Clears display to specified color. If color is not specified, green is used.	`CLS` `CLS(3)`
CONT	Continues program execution after pressing (BREAK) or using STOP statement.	`CONT`
COS	Returns cosine of angle measured in radians.	`Y = COS (7)`
CSAVE	Saves program on cassette (program name must be eight characters/spaces or less).	`CSAVE"PROGRAM"`

Word	Purpose	Example
CSAVE*	Saves contents of a numeric array on cassette for later use by CLOAD*.	`CSAVE*B,"MOREDATA"`
DATA	Stores data in your program. Use READ to assign this data to variables.	`DATA 5,3,PEARS`
DIM	Dimensions one or more arrays.	`DIM R(65),W(40)` `DIM AR$(8,25)`
END	Ends program.	`END`
EXP	Returns natural exponential of number number (e number).	`Y = EXP(7)`
↑	Exponentiation. Raises number to specified power.	`? 3 ↑ 2`

FOR...TO STEP/NEXT	Creates a loop in program which the Computer must repeat from the first number to the last number you specify. Use STEP to specify how much to increment the number each time through the loop. If you omit STEP, one is used.	`FOR X=2 TO 5:NEXT X`
		`FOR A=1 TO 5 STEP 5:NEXT A`
		`FOR M=10 TO 5 STEP -5`
		`NEXT M`

GOSUB	Sends the Computer to subroutine beginning at specified line number.	`GOSUB 500`
GOTO	Sends the Computer to specified line number.	`GOTO 300`
IF/THEN	Test the relationship. If it is true, the Computer executes the instruction following THEN.	`IF A=5 THEN 30`
INKEY$	Strobes the keyboard and returns the key being pressed.	`A$=INKEY$`
INPUT	Causes the Computer to stop and await input from the keyboard.	`INPUT X$` `INPUT "NAME";N$`

Word	Purpose	Example
INT	Converts a number to an integer.	`X=INT(5,2)`
LEFT$	Returns left portion of string.	`P$=LEFT$(M$,7)`
LEN	Returns the number of characters in a string.	`X=LEN(SEN$)`
LET	Assigns value to variable.	`LET A$ = "JOB"`

LIST	Lists specified lines or entire program on Screen.	`LIST`
		`LIST 50-85`
		`LIST 30`
		`LIST -30`
		`LIST 30-`

LLIST	Lists specified program line(s) or entire program to printer.	`LLIST`
		`LLIST 50-85`
		`LLIST 30`
		`LLIST -30`
		`LLIST 30-`

LOG	Returns natural logarithm.	`Y=LOG(353)`
LPRINT	Prints an item or list of items on the printer.	`LPRINT,CAP$`
MEM	Returns amount of free memory.	`PRINT MEM`
MID$	Returns a substring of another string. If length option is omitted, the entire string right of position is returned.	`? MID$(A$,3)`
NEW	Erases memory contents.	`NEW`
ON...GOSUB	Multi-way branch to specified subroutines.	`ON Y GOSUB 50,100`
ON...GOTO	Multi-way branch to specified lines.	`ON X GOTO 190,200`
PEEK	Returns the contents in the memory location you specify.	`A=PEEK(32076)`
POKE	Puts value into specified RAM (memory). Value must be between 0—255 and location between 16384—20479.	`POKE 16390,0`

Word	Purpose	Example
POINT	Tests whether specified graphics cell is on or off. The value returned is −1 if the cell is in the character mode, 0 if off, or the color code if it is on.	`IF POINT (13,35)THEN`

PRINT	Prints specified message on TV Screen.	`PRINT"HI"`
		`PRINT A$`
		`? A$`

PRINT TAB	Moves cursor to specified column.	`PRINT TAB(5)"NAME"`
PRINT@	Prints specified message at specified Screen location.	`PRINT @256, "HI"` `PRINT @ 256, A$`
READ	Reads the next item in DATA line and assigns it to specified variable.	`READ A$` `READ C,B`
REM	Allows insertion of comment in program line. Everything after REM is ignored by the Computer.	`REM THIS IS IGNORED` `10 PRINT X:REM THIS IS IGNORED`
RESET	Erases dot SET at specified location.	`RESET (14,15)`
RESTORE	Sets the Computer's pointer back to first item on the DATA lines.	`RESTORE`
RETURN	Returns the Computer from subroutine to the BASIC word following GOSUB.	`RETURN`
RIGHT$	Returns a right portion of string.	`ZIP$=RIGHT$(AD$,5)`
RND	Returns a pseudo-random integer between one and specified number which must be greater than one.	`A = RND(10)`
RUN	Executes a program.	`RUN`

Word	Purpose	Example
SET	Sets a dot at specified Screen location to specified color.	`SET(14,13,3)`
SGN	Returns sign of specified numeric expression: −1 if argument is negative 0 if argument is 0 +1 if argument is positive	`X=SGN(A*B)`
SKIPF	Skips to end of next program on cassette tape, or to end of specified program.	`SKIPF"PROGRAM"`
SIN	Returns sine of angle measured in radians.	`Y = SIN(5)`
SOUND	Sounds specified tone for specified duration.	`SOUND 128,3`
STOP	Stops execution of a program.	`STOP`
STR$	Converts a number expression to a string.	`S$=STR(X)`
SQR	Returns the square root of a number.	`Y=SQR(5+3)`
TAN	Returns tangent of angle measured in radians.	`Y = TAN(45.7)`
VAL	Converts a string to a number.	`A=VAL(B$)`

SERVICE POLICY

Radio Shack's nationwide network of service facilities provides quick, convenient, and reliable repair services for all of its computer products, in most instances. Warranty service will be performed in accordance with Radio Shack's Limited Warranty. Non-warranty service will be provided at reasonable parts and labor costs.

Because of the sensitivity of computer equipment, and the problems which can result from improper servicing, the following limitations also apply to the services offered by Radio Shack:

1. If any of the warranty seals on any Radio Shack computer products are broken, Radio Shack reserves the right to refuse to service the equipment or to void any remaining warranty on the equipment.
2. If any Radio Shack computer equipment has been modified so that it is not within manufacturer's specifications, including, but not limited to, the installation of any non-Radio Shack parts, components, or replacement boards, then Radio Shack reserves the right to refuse to service the equipment, void any remaining warranty, remove and replace any non-Radio Shack part found in the equipment, and perform whatever modifications are necessary to return the equipment to original factory manufacturer's specifications.
3. The cost for the labor and parts required to return the Radio Shack computer equipment to original manufacturer's specifications will be charged to the customer in addition to the normal repair charge.

RADIO SHACK A DIVISION OF TANDY CORPORATION

U.S.A. — FORT WORTH, TEXAS 76102
CANADA-BARRIE, ONTARIO L4M 4W5

TANDY CORPORATION

AUSTRALIA
91 Kurrajong Road
Mount Druitt, N.S.W. 2770

BELGIUM
Parc Industriel De Naninne
5140 Naninne

UNITED KINGDOM
Bilston Road Wednesbury
West Midlands WS10 7JN

4A3

Printed in Japan
811012490A