MORE TRS-80™ BASIC

A SELF-TEACHING GUIDE

TANDY CORPORATION - FT. WORTH, TEXAS
NYSE STOCK ACTIVITY FOR PERIOD : 01/01/78 TO 04/30/78
AVG DAILY PRICE PER SHARE
39
38
37
36
35
34
33
32
31
30
JAN
FEB
MARCH
APRIL
Radio Shack
TRS-80
VIDEO DISPLAY

N INMAN
MON ZAMORA
ALBRECHT

MORE TRS-80 BASIC™

DON INMAN
RAMON ZAMORA
and
BOB ALBRECHT

Dymax Corporation
Menlo Park, California

John Wiley & Sons, Inc.
New York • Chichester • Brisbane • Toronto

Copyright © 1981, by John Wiley & Sons, Inc.

All rights reserved. Published simultaneously in Canada.

Reproduction or translation of any part of this work beyond that permitted by Sections 107 or 108 of the 1976 United States Copyright Act without the permission of the copyright owner is unlawful. Requests for permission or further information should be addressed to the Permissions Department, John Wiley & Sons, Inc.

Library of Congress Cataloging in Publication Data:

Inman, Don

More TRS-80 BASIC.

(Wiley self-teaching guides)
Includes index.
I. TRS-80 (Computer)—Programming. 2. Basic.
(Computer program language) I. Zamora, Ramon. II. Albrecht,
Bob, 1930- . III. Title. IV. Series.
QA76.8T18I56 001.64′2 81-150

ISBN 0-471-08010-1 AACR2

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

TO THE READER

If you bought this book we assume that you already have a TRS-80 microcomputer, or the use of one in a school or office. And we hope that you have read, enjoyed, and learned from our earlier book, TRS-80 BASIC.* (Or another book that introduced you to BASIC and the TRS-80 computer.) This book starts where TRS-80 BASIC ended and will give you more hours of pleasure from your computer than ever before.

We begin with a review of commands and statements and move quickly to a detailed guide to your TRS-80's memory so you can get the most out of PEEK and POKE and use your computer most efficiently. We then progress through Graphics, Cassette and Disk Files, and Animation. Lots of programs demonstrate what is taught and there is plenty of opportunity to change programs, add to them, and write your own. You get some tricks for saving space and many special programs that range from car races to phone indices. And Self-Tests let you check your progress and understanding throughout the book.

If you have read any of our other books you know that we feel computer terminology and concepts can be introduced within a framework of fun and exploration, and that when we do this, *true* learning takes place. Microcomputers are here to stay. Everything in this book is directed toward helping you fully explore the workings and uses of your TRS-80 microcomputer.

The next section tells you How to Use This Book. Why don't you first browse through that, then begin your further adventures with your TRS-80 and BASIC.

*TRS-80 BASIC, Albrecht, Inman, and Zamora, (C) John Wiley and Sons, Inc., 1980.

HOW TO USE THIS BOOK

This book is a Self-Teaching Guide. This means that *you* can use the book to teach *yourself*. Each chapter of the book is composed of short bites of information presenting a single idea or topic on the BASIC language, the TRS-80, a special feature, or a program that is being developed. Throughout the sections there are lots of questions and exercises to be done.

We encourage you to use this book while you are in front of a TRS-80 computer. Try the programs and exercises that are discussed on the machine. Let the TRS-80 be your "teacher."

The first page of each chapter briefly lists what the chapter is to cover. Scan that list. If you feel you already know the material to be covered skip to the back of the chapter and take the Self-Test. You can review the chapter if you have trouble answering the questions.

The material in the book gets more challenging as you move through the chapters in order. If you have only a basic knowledge of BASIC and your TRS-80 start with the first chapters and don't try to skip around. If you know quite a lot of BASIC and are pretty familiar with your computer look at the Table of Contents and feel free to explore some of the later chapters.

As a final note before you start, be aware that as you use the book, enter lines into your computer, answer Self-Test questions, and do the exercises:

YOU CANNOT DO ANYTHING WRONG!

There is no way you can "hurt" or "harm" the computer by what you type into it. You may make *mistakes,* but that is a natural part of learning and exploration. In fact, we introduce deliberate errors in several places as part of the learning process.

So, explore, enjoy, and tell us about your discoveries as you use your Radio Shack TRS-80 microcomputer.

试读结束：需要全本请在线购买： www.ertongbook.com

Contents

CHAPTER ONE

Introduction

Welcome to this second book in a series designed to help you discover and use more of the features of your Radio Shack TRS-80 Model I computer. In the first book, *TRS-80 BASIC,** we assumed you were a newcomer to computers and computing. We used a frame-by-frame presentation of material that kept everything down to small, bite-sized lumps. Here, however, we assume that you have finished the first book (or its equivalent) and that you are familiar with some aspects of the Level II BASIC language. The material is presented as a series of chapters that expand on what was covered in the beginner's text. It includes detailed discussions of several new language features of your TRS-80 and self-test exercises and answers at the end of nearly every chapter.

* Albrecht, Inman, and Zamora, *TRS-80 BASIC: A Self-Teaching Guide.* John Wiley & Sons, Inc., N.Y., N.Y., 1980.

In this chapter, we will preview some of the things you will learn and briefly review the commands, statements, terms, and other features of BASIC that we covered in *TRS-80 BASIC*. In chapter 2, we will assume that you are familiar with the elements of BASIC and begin introducing new features of TRS-80 BASIC.

Throughout this book, each new BASIC statement is described in detail when it is first used. Longer programs are broken into logical blocks, or sections, and each section is explained thoroughly. When the user must interact with a program, there are sample program results with sketches of video displays. The material covered is summarized at the end of each chapter.

We have worked to make this book easy to read, to understand, and to use. Some programs appear more than once. These repetitions provide continuity and emphasize relationships between new and previously used instructions. Elementary forms of some programs appeared in our beginners' book.

The *Level II BASIC Reference Manual,* supplied with your TRS-80 Model I computer, is used often as a source for discussions of BASIC statements, commands, and functions. Some Radio Shack hardware and software products are also used as examples.

The central theme of the book is practical, application-based uses of the machine. In some cases, the information is developed in terms of educational or recreational programs. Where possible we indicate several areas where a technique or approach may apply. You will also be introduced to file handling techniques and applications of your TRS-80 cassette recorder and disk. The TRS-80 has so many features and capabilities that we were hard pressed to keep this second book down to its current size.

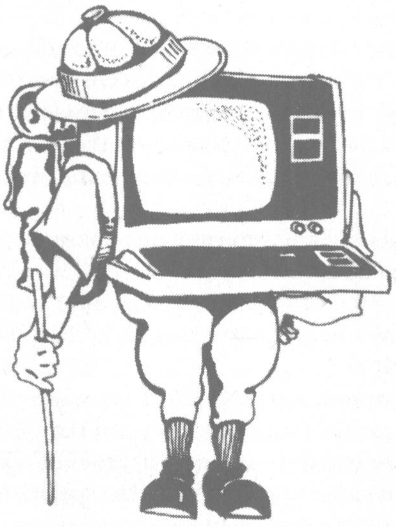
We do, however, cover new statements, such as PEEK and POKE, the many mathematical and ERROR functions and routines, and most other features of TRS-80 Level II BASIC that were not described in our first book. We include two full chapters of useful, powerful graphic techniques and a chapter on the generation of sound and music, using inexpensive devices that plug into your TRS-80.

The addition of easy-to-use graphics and sound has lifted computer use out of the number and word crunching business and industrial world and greatly enhanced the computer's recreational and educational aspects. Today's microcomputers, such as the TRS-80, have created an opportunity for nearly everyone to own, use, and master the "mysteries" of a small computer. The small computer is finding its way into the home, the school, and the small business. Thousands of programs are being developed for these machines to perform a wide variety of educational, recreational, and business-related tasks.

With the steady increase of programs and computer users, the number of new applications for these powerful tools will continue to grow. New users bring new interests and avocations that lead to different problems to be solved. Everyone benefits as new solutions are discovered and shared, opening up new areas to apply the tool. The cycle feeds on itself, and everyone has an opportunity to be an inventor and creator in this rapidly growing field.

This book will help you expand your knowledge of Level II BASIC and expand your ability to create programs. As you make greater use of your TRS-80, you will find that you and those around you have both the ability and skill to design programs that perform any number of tasks you may wish to do. By the time you complete your tour of this book, you will have used nearly *every* BASIC language command and statement available on a Level II 16K machine. For those of you with larger machines (24K, 32K, and so forth), the material in this book will work as well.

As always, we encourage you to venture forward into the book with a spirit of exploration and discovery. We believe that learning to use a small computer can be both fun and educational. In that vein, we occasionally try to lighten things here and there with a bit of humor, and even go so far as to POKE (ooops, there we go — poke) fun at ourselves. Enjoy learning some new ways to use your TRS-80, and share what you learn with those around you. Are you ready? Let's go! If you need to review, move on to the next section. If not, skip the review, and look at chapter 2 for a "guided tour of memory."



Things We Expect You to Know

Although we assume that you have a background knowledge of Level II BASIC, a brief review of the BASIC statements, commands, and functions from the beginner's book is presented here. If you feel confident in your knowledge of Level II BASIC, you can either pass quickly through the review section or skip on to chapter 2.

BASIC Statements and Commands

ASC(*string*) Returns the ASCII code of the first character in the string argument specified in parentheses.

CHR$ (*exp*) Returns a one-character string of *exp*, which may be an ASCII, control, or graphics code.

[illegible]

Clear n Sets numeric values to zero and strings to null; sets aside n bytes of string space in memory.

CLS Clears the video display.

DATA item list Holds the data for access by a READ statement; items may be numeric or string; items are separated by commas.

DELETE *mm-nn* Deletes the program lines from line *mm* through *nn.*

DIM array (*dim#1, dim#2, . . .dim #k*) Reserves storage for a k-dimensional array with the specified size per dimension; DIM may be followed by a list of arrays separated by commas.

EDIT *mm* Puts you in the edit mode at line *mm.*

EDIT Puts you in the edit mode at the last line entered, altered, or in which an error has occurred.

END Ends the execution of a program.

FOR *var = exp* TO *exp* STEP *exp* Opens a FOR-NEXT loop. STEP is optional—if not used, STEP will be 1.

GOSUB *mm* Branches to the subroutine beginning at line *mm.*

GOTO *mm* Branches to the line number specified by *mm.*

IF *exp* THEN *statement #1* ELSE *statement #2* Tests the expression, *exp*; if True, statement #1 is executed and control proceeds to the next program line (unless statement #1 is a GOTO); if *exp* is False, statement #2 is executed; the ELSE statement #2 is optional.

INKEY$ Reads the keyboard and returns a one-character string, which is the string value of the key that was last pressed.

INPUT "*message*"; *var* PRINTs the message (if any) and waits for an input from the keyboard; *var* may be a numeric or string variable; a list of variables separated by commas may be used.

INT (*exp*) Returns the largest integer that is not greater than the expression, *exp.*

LEFT$ (*string, n*) Returns the first n characters of the specified string.

LEN (*string*) Returns the length (number of characters) of the string.

LET *var = exp* Assigns a value equal to *exp* to the specified variable, *var.*

LIST Lists the entire current program in memory.

LIST *mm-nn* Lists only lines *mm* through *nn* of a program; *-nn* is optional.

MID$ (*string, p, n*) Returns a substring of the specified string; the length of the substring will be *n* starting with the character in position *p.*

NEW Deletes the current program in memory and resets all variables, pointers, etc.

NEXT *var* Closes a FOR-NEXT loop; the variable, *var*, is optional.

POS (0) Returns a number indicating the current cursor position; the argument (0) is a dummy variable.

PRINT *exp* Output the value of *exp* to the display; *exp* may be numeric or string variable or constant, or a list of such items.

PRINT @*n* Modifies the PRINT statement to begin printing at the display position *n.*

RANDOM Reseeds the random number generator (starts with new number).

READ *variable list* Assigns values to the specified variables starting with the current data item from the DATA statement.

REMARK Remark indicator; tells the computer to ignore the rest of the current line; abbreviation REM may be used also.

RESET (*X, Y*) Turns off the graphic blocks at horizontal coordinate *X* and vertical coordinate *Y*.
RESTORE Resets the data pointer to the first item in the first DATA statement.
RETURN Branches to the statement following the last executed GOSUB.
RIGHT$ (*string, n*) Returns the last *n* characters of the string specified.
RND (0) Returns a pseudorandom number between 0.000001 and 0.999999, inclusive.
RND (*exp*) Returns a pseudorandom integer between 1 and INT (*exp*), inclusive.
RUN *mm* Executes a program beginning at line *mm*; *mm* is optional; if not specified, the execution begins at the lowest numbered line.
SET (*X, Y*) Turns on the graphic block at horizontal coordinate *X* and vertical coordinate *Y*.
STRING$ (*n, char*) Returns a sequence of *n* character symbols using the first character of *char*.
TAB *n* Modifies a PRINT statement; moves cursor to the specified display position (0 through 63) on a given line.
TROFF Turns off the trace function.
TRON Turns on the trace function that displays the line number of each line executed.
VAL (*string*) Returns the number represented by the characters in the specified string.

Frequently Used Terms

Backspace key Used to move the cursor left one space and erase the character in that space. If the SHIFT key is held down at the same time, an entire line is erased.
Break key Stops the computer at the current line being executed.
Cassette recorder Used to store data outside the computer.
Clear key Clears the TV display.
Concatenation Joins two or more strings together.
Cursor Shows where the next printed character will appear.
Data pointer A pointer (or counter) the computer uses to tell it which item to select from a data list.
Debug To remove errors, or "bugs," from programs.
Edit mode Allows changes in an existing program without retyping the whole line.
Empty string("") A string with nothing in it, not even a space.
Enter key Tells the computer a command or statement is completed.
Flag A signal to the computer that some process has been completed and should be discontinued.
Floating point A special method of writing very large or very small numbers.
Inequality symbols Symbols used to express a relationship between two quantities that are not necessarily equal.
Keyboard The control center of the TRS-80.
Line number Used to tell the computer which line to execute next.
Multiple-statement line A program line that contains more than one statement to be executed.

试读结束：需要全本请在线购买： www.ertongbook.com

Numeric variable A variable that identifies a number.

One-dimensional array A list of numbers or strings arranged in a specific order.

Order of operations The order in which arithmetic operations are performed within a given arithmetic expression.

Power of numbers Tells how many times a number is to be used as a factor.

Power supply The TRS-80 power supply converts 110- to 120-volt AC to smaller DC voltages.

Print positions Positions on the video display that are numbered from the upper left to the lower right (0 through 1023).

Prompt The > symbol tells you that it's your turn to tell the computer what to do.

RAM (Random Access Memory) Used to store your programs and data.

Random number A number chosen from a given set of numbers so that each number has an equal chance of being selected.

Ready A message printed on the video screen that indicates the computer is ready to do something.

ROM (Read Only Memory) Used to store the TRS-80 operating system and the BASIC interpreter.

Rounding numbers Changing values to the nearest specified decimal place.

Sequence of values Values that come one after another in order.

String A string of numerals, letters, and/or special characters.

String variable A variable that identifies a particular string.

Superscript Specifies the power of a number.

Two-dimensional array A table of numbers or strings consisting of rows and columns.

Variable subscript Denotes the order of an element in an array named by the variable.

Video display The TV screen on which graphics and/or text is shown.

CHAPTER TWO

A Guided Tour of Memory

The TRS-80 performs its wondrous feats with a carefully planned, ingenious use of memory. Almost everything that goes on inside the computer uses memory in some way. The computer's operating procedures, the BASIC programs you write, and the data used by your programs are all stored in memory.

Did you ever wonder why you lose your programs when the computer is turned off, yet the computer is still able to operate when it is turned on again? If the computer "forgets" your programs when turned off, how can it "remember" its own operating procedures? Stay with us. This chapter presents a map to guide you through TRS-80 Memory Land. You will see the different kinds of memory and learn how each is used. When you finish the chapter, you'll be able to PEEK and to POKE around inside your own memory. You'll also have a better understanding of how the computer works and how you can use it more efficiently.

You will learn:

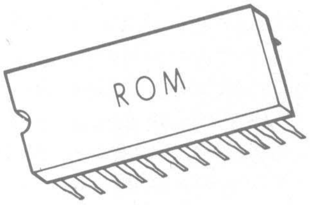
- how ROMs and RAMs are packaged,
- how the CPU works with ROM,
- how to PEEK into ROM,
- the difference between ROM and RAM,
- how ROM and RAM are organized,
- how your BASIC program uses RAM,
- why the MEMORY SIZE? prompt is used,
- how memory space is reserved for strings,
- how to conserve memory space,
- how to PEEK and/or POKE into RAM,
- how to use POKE with care, and
- where the video screen is located and how to use it.

What is Memory?

The first chapter of *TRS-80 BASIC** briefly discusses two kinds of memory—ROM and RAM. The computer can read information from ROM, but cannot erase or change it in any way. Therefore, this type of memory is called *Read Only Memory,* or ROM for short. Information in ROM is *permanently* stored, much like the information on the pages of this book or the information on a phonograph record. It cannot be changed and it is not "forgotten" when the computer is turned off.

A ROM in a computer can be replaced, just as a book is replaced by a new, revised edition or a record is replaced in your phonograph. But the information in that particular ROM cannot be changed from the keyboard or by a BASIC program.

The ROMs inside the TRS-80 are enclosed in rectangular packages with twenty-four small, metal legs, twelve on each side. These legs make electrical connections to the rest of the computer.



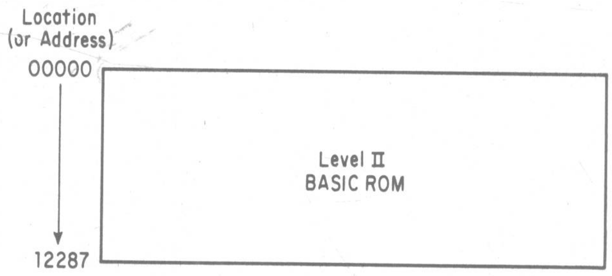
The heart of the computer is called the *Central Processing Unit* (CPU). It is contained in a package similar to the ROM's. Although the CPU does most of the work (or processing) for the computer, the ROMs are the boss; they tell the CPU what to do, how and when to do it, and where to put the results.

When the TRS-80 is turned on, the CPU immediately looks at a certain memory location in ROM to see what it should do. The ROM then takes over, giving directions to "fire up" the system and get it READY for your input.

Since the ROM is so important to the computer, it is placed at the beginning of the memory block. Regardless of whether you have a 4K, 16K, 32K, or other size TRS-80, *the ROM occupies the memory locations numbered from 0 through 12287*. Therefore, every TRS-80 Level II computer has a 12K ROM (K for thousand). Each K actually refers to 1024 locations.

*Albrecht, Inman, and Zamora, *TRS-80 BASIC: A Self-Teaching Guide*. John Wiley & Sons, Inc., N.Y., N.Y. 1980.

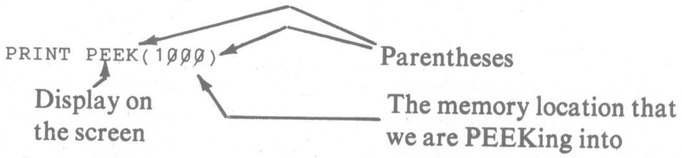
12K = 12 × 1024 = 12288 (Numbered 0 through 12287)



Remember, you can only read information *from* ROM. Nothing can be written into the memory locations from 0 through 12287.

PEEK into ROM

You can read information from ROM using Level II BASIC's PEEK function. This function lets you PEEK at (or read) the information stored in one memory location. PEEK can be used with a PRINT statement in the Immediate Mode, as well as within a program.



Now, turn on your TRS-80 and PEEK around a bit.

- When it says MEMORY SIZE? you press ENTER

This is what you see:

```
MEMORY SIZE?
RADIO SHACK LEVEL II BASIC
READY
>-
```

- You type: PRINT PEEK(1000) and press ENTER

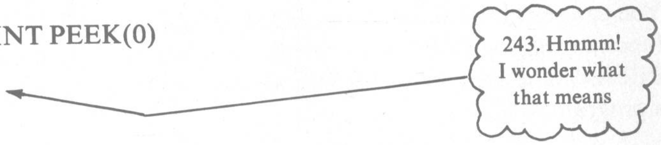
This is what you see:

```
MEMORY SIZE?
RADIO SHACK LEVEL II BASIC
READY
  PRINT PEEK(1ØØØ)
  56
READY
>-
```

By PEEKing, we found that the number 56 is stored in ROM location 1000. Hmmm, wonder what is in location 0.

- You type: PRINT PEEK(0)

  It prints: `243`



So far, you've found a number (243) in memory location 0 and another number (56) in memory location 1000. Is that all you're going to find — a bunch of numbers? Try another. This time find out if the PEEK statement can use a variable instead of the numeric location.

- You type: A = 1
- You type: PRINT PEEK(A)

  It prints: `175` (This number is in memory location 1, since A = 1.)

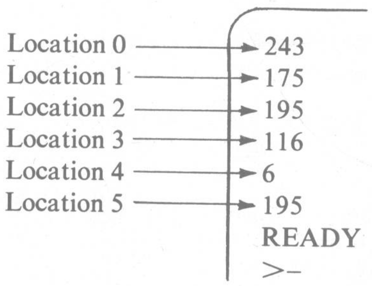
You now know this about the Level II ROM:

Location	Value
0	243
1	175
•	•
•	•
•	•
1000	56

To find what is in the first few locations, type in the following single line and press ENTER:

```
CLS: FOR A=Ø TO 5: PRINT PEEK(A): NEXT A
```

Lo and behold! You get the numbers stored in memory locations 0 through 5.



```
243
175
195
116
6
195
READY
>_
```

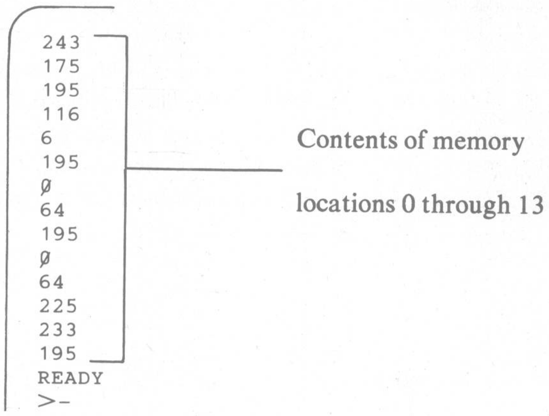
That one line in the Immediate Mode performs the same function as the following BASIC program:

```
1Ø CLS
2Ø FOR A = Ø TO 5
3Ø PRINT PEEK(A)
4Ø NEXT A
```

Try using the program and see if you get the same results. To look at more results, change line 20. If you try to get too much data on the screen, the results will scroll by so fast that you can't see them all. Try changing line 20 to:

```
2Ø FOR A = Ø TO 13
```

Now run the program again and compare your results with ours.



```
243
175
195
116
6
195
Ø
64
195
Ø
64
225
233
195
READY
>_
```

By this time, you may be convinced that all ROM contains is a bunch of numbers. What do the numbers mean? They don't mean anything to you, but they *do*

试读结束：需要全本请在线购买： www.ertongbook.com