14.95

The Super Computer Snooper

By Dr. Isaac Malitz

FIND OUT WHAT GOES ON INSIDE THE IBM-PC

The Super Computer Snooper IBM-PC

[illegible]

TP32
M1

The Super Computer Snooper IBM-PC

by
Dr. Isaac Malitz

Illustrated by
Robert Peters

DATAMOST™ INC
20660 Nordhoff Street
Chatsworth, CA 91311-6152
(818) 709-1202

RESTON PUBLISHING COMPANY, INC.
A Prentice-Hall Company
Reston, Virginia

ISBN 0-8359-7148-1

Printed in U.S.A.

Table of Contents

ACKNOWLEDGEMENTS

I want to express my deep appreciation to the many wonderful people who helped to make this book possible:

For telling me to write this book: Hal Glicksman.

For technical advice: Alan Bartholemew, Howard Preston.

For providing the illustrations and layout, which have so greatly increased the clarity of this book: Bob Peters, Scott Ross, Darlene Nunez.

For turning my manuscript into a real book: Marcia Carrozzo, Editor-in-Chief of DATAMOST, Inc.

For "making it all happen": Dave Gordon, President of DATAMOST, Inc.

Finally, I want to thank my wife and daughter for love, forbearance, and high inspiration.

INTRODUCTION

What goes on inside the IBM-PC? How does it work?

This book will help you to find out.

You will need an IBM-PC with 64K and a DOS disk. You should know how to write simple programs in BASIC on the PC, however, no other special knowledge is needed.

We are going to look at each of the main parts of the PC, and find out how they work together. You will learn about memory, screen, programs and variables, keyboard, printer, and expansion boards.

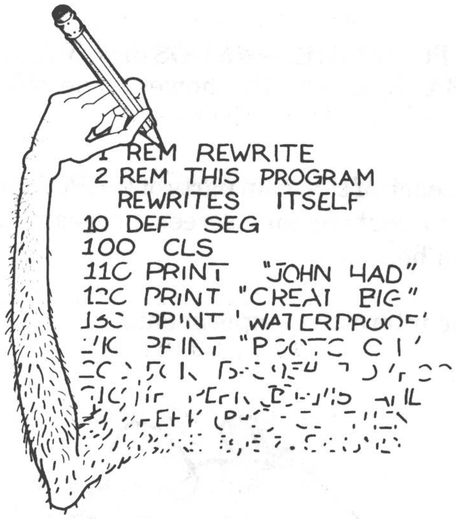
You will perform some interesting experiments:

You will find out how to restore a program that has been accidentally erased.

You will find out how to "listen" to the inner workings of the PC, using an ordinary radio.

You will find out how to identify deleted or "hidden" files on a disk.

You will write a program which re-writes itself.



You will create some amazing graphics effects.

When you have finished this book,

You will understand better how a computer works — especially the IBM-PC.

You will have learned some powerful techniques that you can use when you write your own programs.

You will be prepared to study advanced topics, such as machine language programming or arcade graphics, should you ever want to do this.

The IBM-PC is a mind-boggling machine. Each second, millions of electronic signals pass through it. These signals travel at speeds of nearly one billion feet per second. They criss-cross and interact with dazzling complexity. They are organized to perform tasks that sometimes are almost completely beyond the capability of human beings.

How does the PC work? What goes on in there?

Let's find out . . .

GETTING READY

1. Make a copy of your IBM DOS diskette and label it "Super Computer Snooper". We will call this your SCS Disk. Use it for all of the exercises in this book.

   If you need instructions for copying your DOS diskette, see the *IBM Guide to Operations*, or consult with an experienced operator.

2. Insert the SCS diskette in drive A.

3. Turn the power on. (If power is already on, press Ctrl, Alt, and Del keys simultaneously. This will "warm start" the system.)

4. The `A>` (A prompt) will appear on the screen.

5. Type: `BASICA` and press the Enter key. This will start Advanced Basic.

You are now ready to begin . . .

HELPFUL HINTS

The following tips will help you enjoy your exploration of the IBM-PC.

1. We will suggest lots of experiments in this book. We encourage you to try them, and to even make up your own experiments. You will have fun, and you will learn a lot about your IBM-PC. Don't worry, you won't break anything!

2. Some of our experiments will involve the computer's memory. To clear memory afterwards, turn the computer off, wait thirty seconds, and then turn it back on. Do not use a "warm boot" to clear memory, since a warm boot does not necessarily clear all areas of memory.

   Whenever you have finished a session with this book, and you want to do something else at the computer, always clear memory, as described above. Otherwise your experiments may leave stray information in memory that could interfere with the processing of other programs.

3. Some of the experiments may alter the data on your diskette. That is why we recommend that you make a special copy of your DOS diskette for use only with this book.

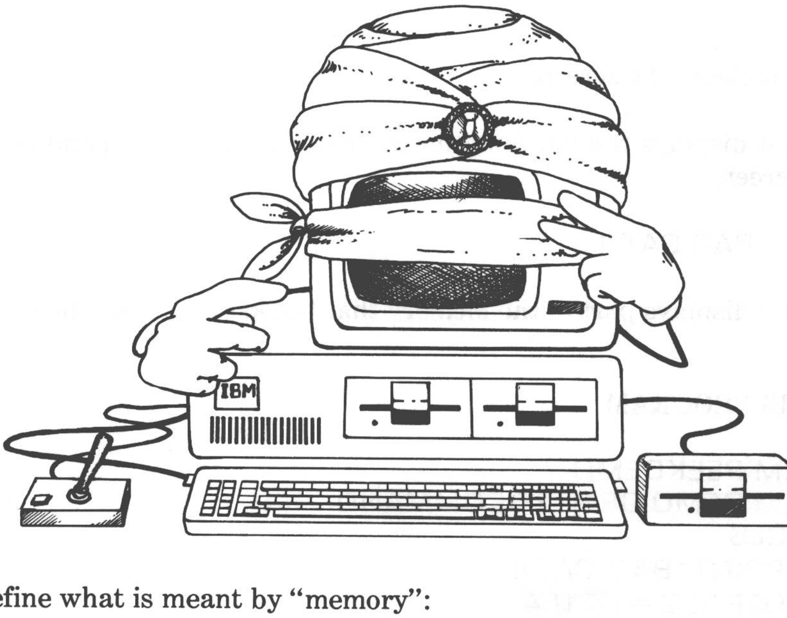
4. Nothing in this book will require you to open your computer. We recommend that you not open your computer, except with guidance from an expert.

Chapter 1
MEMORY

The fastest way to start learning about the PC is to look inside its *memory.*

Whatever is happening inside the PC, the memory usually knows about it.

> When you are running a program, the program and its variables are in the PC's memory.
>
> The memory knows at all times what line of the program is currently being executed.
>
> The memory knows what is being displayed on the screen.
>
> When the computer is sending or receiving signals from a disk drive, screen, printer, keyboard, joystick, or anything else attached to the computer, memory usually knows about it.



Let's define what is meant by "memory":

Memory is the part of your computer which holds information the computer is currently using. When you are running a program, the program and all of its variables are held in memory. Memory is also used for many other purposes, as we shall see.

Memory is divided into a large number of "storage locations." Each location is like a little box that can store a small amount of information. The locations are numbered 0, 1, 2, 3, and so on.

Each location can hold one "byte" of information. A byte is about the same as one character of information. A byte is stored as a value ranging from 0-255. If you look in any location in memory, you will find a value from 0-255.

An IBM PC comes with a minimum of "64K" of memory. "64K" means 64 thousand bytes, or 64 thousand storage locations. You may buy upgrades for memory that increase it to 256K. Additionally, the IBM PC has some special-purpose memory for its internal use, which normally you cannot access. We will find out more about this later. Altogether, the IBM PC has room for about one million bytes of memory of all types.

If you could look into the PC's memory, it would show you almost everything that is happening in the computer. You can look into the memory by using the PEEK command. Let's find out how to do this.

We are going to write a program in BASIC called PEEKDEMO. This program will show you how to use the PEEK command to look into the computer's memory.

Here is what PEEKDEMO does:

> First it clears the screen.
>
> Then it displays the following message in the upper left-hand corner of the screen:
>
> BAD CAT !!
>
> Then it displays part of the memory that tracks what is on the screen.

ENTER THIS PROGRAM

```
1 REM PEEKDEMO
2 REM DEMO OF PEEK COMMAND
100 CLS
110 PRINT "BAD-CAT !!"
200 DEF SEG = 47104!
210 PRINT:PRINT
220 FOR I = 0 TO 19
230 P = PEEK(I)
240 PRINT USING "####";P;
250 NEXT I
```

(If you are using a monochrome screen, change line 200 to

```
    200 DEF SEG = 45056!
```

In this chapter and the next, there will be some more references to the number 47104!. Always substitute the number 45056!)

Save this program on your SCS disk under the name PEEKDEMO. We will be writing many programs in this book. We suggest that you save them on your SCS disk. In that way you will build up a useful collection of programs for computer-snooping. Save all programs under the name listed in the first line of the program.

Now run the program. Here is what you will see on your screen:

```
BAD CAT!!


66 7 65 7 68 7 45 7 67 7 65 7 84 7 32 7 33 7 33 7
```

Those codes on the second line are memory's way of describing the upper left-hand corner of the screen. Each of the codes has a meaning. Notice that every other code is 7. We will explain that code later. Here is what the other codes mean:

```
66 7 65 7 68 7 45 7 67 7 65 7 84 7 32 7 33 7 33 7


B    A    D    -    C    A    T         !    !
```

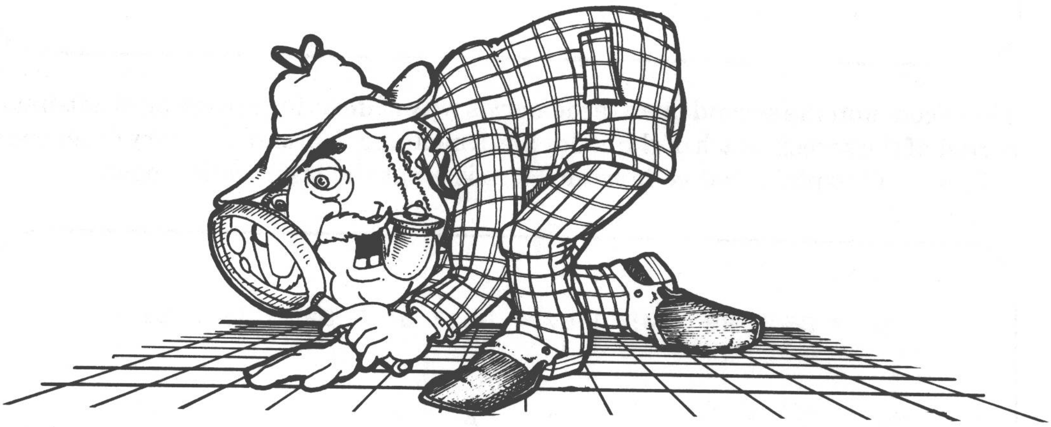
Each of the codes 66, 65, 68, and so on stand for a character that is displayed on the screen.

The 7s describe how each character is displayed on the screen. 7 means: White character on a dark background, with no blinking. Other codes are possible. For instance, the code 135 means: Blinking white character on a dark background. The code 240 means: Blinking black character on a white background. Code 1 means: a blue character on a black background, no blinking. (If you have a monochrome (black-and-white) screen, this will appear as an underlined character.)

Each of the codes — 66, 7, 65, 68, and so on — occupies one byte of memory. The value of each code can range from 0 to 255.

Now let's summarize what we have found out about memory, and how it relates to the screen.

> There is a section of memory which describes exactly what is on the screen.
>
> For each position on the screen, two bytes of memory are needed to describe what appears in that position on the screen.
>
> The first byte holds a code which describes what character is in that position. 65 means "A", 66 means "B", 67 means "C", and so on. (A complete chart of character codes is given in Appendix B.)
>
> The second byte holds a code which describes how the character is displayed. 7 means a white character on a black background. Other codes stand for reverse video, blinking, or color effects. (A complete description of these codes is given in the next chapter.)



In conclusion, a certain section of memory contains codes which describe each position on the screen. As we shall see shortly, if you alter any of those codes, the appearance of the screen will change.

Let's go back and look at your program PEEKDEMO, and find out how it works.

How PEEKDEMO Works:

Lines 100 - 110

```
100 CLS
110 PRINT "BAD-CAT !!"
```