

How To Use The **COMMODORE**[®] **64**[™] COMPUTER



Jerry Willis and Deborrah Willis

How to Use The Commodore® 64™ Computer

Jerry Willis and Deborrah Willis



dilithium Press
Beaverton, Oregon

©1984 by dilithium Press. All rights reserved.

No part of this book may be reproduced in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage and retrieval system without permission in writing from the publisher, with the following exceptions: any material may be copied or transcribed for the nonprofit use of the purchaser, and material (not to exceed 300 words and one figure) may be quoted in published reviews of this book.

Where necessary, permission is granted by the copyright owner for libraries and others registered with the Copyright Clearance Center (CCC) to photocopy any material herein for a base fee of \$1.00 and an additional fee of \$0.20 per page. Payments should be sent directly to the Copyright Clearance Center, 21 Congress Street, Salem, Massachusetts 01970.

10 9 8 7 6 5 4 3 2 1

Library of Congress Cataloging in Publication Data

Willis, Jerry.

How to use the Commodore 64 computer.

Includes index.

1. Commodore 64 (Computer)—Programming. 2. Basic (Computer program language) I. Willis, Deborah. II. Title.

QA76.8.C64W54 1984 001.64 83-15419

ISBN 0-88056-133-5

Cover: Greg Ellingson

Printed in the United States of America

dilithium Press
8285 S.W. Nimbus
Suite 151
Beaverton, Oregon 97005

Table of Contents

Chapter 1	Well What Do We Have Here?	1
Chapter 2	Setup and Installation	9
Chapter 3	How to Get Your Computer Up and Running	21
Chapter 4	Loading Information from Cassette or Disk	31
Chapter 5	Output to Cassette, Diskette or Printer	43
Chapter 6	Programming in BASIC	55
Chapter 7	Getting Information into the Computer	65
Chapter 8	More BASIC	73
Chapter 9	Graphics, Sound and more BASIC	79
Chapter 10	Software for your Computer	93
Chapter 11	Selecting Hardware and Accessories	109
Chapter 12	Sources of Additional Information	125
Appendix A	Abbreviations for BASIC Keywords	129
Appendix B	Error Messages	131

ACKNOWLEDGEMENTS

Apple II	Apple Computer, Inc.
ATARI 400,800	Atari, Inc.
Commodore PET	
Commodore 64	Commodore Computer Corporation
VIC 20	
Midwest Micro Association	
SuperCalc	Sorcim Corporation
VisiCalc	VisiCorp

CHAPTER 1

Well What Do We Have Here?

This book was written for the novice computer user who owns, or is thinking of buying the Commodore Model 64 computer.

Early advertising for this computer carried banner heads that read, "For \$595, you get what nobody else can give you for twice the price." Commodore Model 64 is a lot of computer for the money. The machine has a full size, typewriter style keyboard; it can be programmed in a version of Microsoft BASIC—the most popular computer language; and it has very good color graphics and sound synthesis features. It runs thousands of programs designed for the PET series of computers and, with some optional accessories, can run business software written for computers that use CP/M. It even has a slot for inserting cartridges of your favorite video game.

When the Model 64 was announced many magazine reviewers praised it—“Commodore launches a winner . . . all it is cracked up to be . . . exceptional graphics, color and sound . . . compatibility with VIC peripherals . . . at a price that’s difficult to beat . . . ‘best buy’ in the \$600 personal computer price range . . .”

This book was written for the computer user with little background in computer science or electronics. It provides information for maximum use of the Commodore 64. The book was written with the following objectives:

1. To introduce you to the computer and its basic components, what they do, and how they work together (Chapters 1, 9, and 10).
2. To overview things you can do with your computer (Chapter 1).
3. To give step-by-step instructions on how to set up or “install” your computer and tell you how to get it “up and running” (Chapters 2 and 3).

4. To describe how to load and save programs on standard audio cassettes and diskettes (Chapters 4 and 5).

5. To introduce the novice to BASIC, the standard computer language on the Commodore 64, and to color graphics and sound synthesis with the Commodore 64 (Chapters 6, 7 and 8).

6. To inform you about accessories for your computer and how to select, buy, install, and use them (Chapter 10).

7. To inform you of other sources of information for your computer such as publications, user's groups, and computer based information networks (Chapter 11).

8. Inform you of software available for the Commodore 64, how to select the best programs for your needs, and where to buy it (Chapter 9).

After a brief history of Commodore this chapter will discuss uses of the computer and how it works. If you are eager to get started, move on to Chapter 2.

A LITTLE HISTORY

In 1974 Commodore International was known as a successful manufacturer of calculators. It was a relatively small company that had matched development and marketing skills of giants like Texas Instruments and held its own.

Development of hand-held calculators in the early seventies required manufacturers to design systems that used electronic devices called "integrated circuits," or IC's. With IC's, designers could replace hundreds of discrete transistor circuits with three or four integrated circuits. A standard integrated circuit could perform all the functions of 50 to 200 transistors, drastically reducing the steps and time required to manufacture a calculator. Today's technology has created many hand-held calculators that have only one large integrated circuit in them. IC's can replace tens of thousands of individual transistor circuits.

The little IC, foundation of the calculator industry, was also the foundation of the personal computer revolution. MOS Technology, one of Commodore's chip suppliers, pioneered development of one "computer on a chip," the 6502 microprocessor integrated circuit. Many of the popular computers, Apple II, Atari 800 and 400, Commodore PET, and the VIC 20 all use the 6502 chip.

When Commodore moved into the developing field of personal computers, it bought MOS Technology and combined the market-

ing experience of its calculator background with the technical expertise of its chip development to create a company with a remarkably successful product, the PET computer. In 1977 Commodore announced the first appliance computer, the PET. The \$495 computer was the first one that could be taken home, plugged in, and used immediately.

Orders for the original PET, even with its hard-to-use calculator style keyboard, exceeded all expectations. Overnight, Commodore became one of the major players in the personal computer game. Some say the PET was too successful. The company was not prepared for the product demand and delivery time slowed down. At one time Commodore had a five month backlog of orders. Commodore, in fact, was never able to meet production schedules, establish a strong dealer network, and provide necessary support. Their U.S. market share dropped steadily after the first year. During the 1977-1981 period, unreturned phone calls, accessories announced but not manufactured, design errors, and a generally slipshod mode of operation created a poor image of Commodore and left many dealers and PET purchasers bitter.

While Commodore was lagging in the U.S., it was beginning to dominate major European markets. They built an excellent computer and used excellent marketing strategies in Europe. Between 1977 and 1981, Commodore expanded its original PET computer into a line of five models with four styles of disk drives. The 8032 computer, when combined with an 8050 disk drive, was one of the best computers in the small business market in 1981.

By 1981, however, a good product was not enough. Competitors also had good machines and they answered their phones, supported their dealer network, and were easy to deal with. Commodore still competes in the business market, although it has only a small percentage of the sales despite its excellent products. Commodore still has problems with support and timely production of announced products.

By 1981 the home computer phenomenon provided Commodore with another market—the “under \$300 home computer.” Commodore used the design and manufacturing expertise of MOS Technology to create the VIC 20, an inexpensive but powerful computer. Commodore introduced the VIC 20 in Japan in 1980 and began U.S. marketing in 1981. A buyer could get a real computer for only slightly more than the cost of a video game. Commodore used Star Trek’s Captain Kirk, William Shatner, in its multi-million dollar advertising campaign that sold hundreds of thou-

sands of VIC computers. The VIC is a unit good for everything from video games to stock market speculation.

Commodore's small VIC computer that sells for under \$100 is a front runner in the race to put computers in the home. The Model 64, announced in 1982, and the Executive 64—a portable version announced in 1983, cost several times more. But they offer improved sound and graphics features, a video display with double the screen capacity, and compatibility with the PET computers that have been around long enough to inspire thousands of programs in virtually every area of application. The Commodore 64 is suitable for home or office, classroom, or living room. This book will help you use the computer effectively.

WHAT CAN YOU DO WITH IT?

Many people are skeptical about inexpensive computers. To show you how powerful the Commodore 64 is, this section presents an overview of application. Later chapters discuss software and accessories.

COMPUTER LITERACY

Three hundred years ago, reading and writing were specialized occupations in many societies. Scribes mentioned in the Bible, for example, were professionals who could read and write. Over 97 percent of the adults in the world could not read and didn't have to.

Today the ability to read and write is essential if a person is to succeed in our complex society. The public school system tries to help every citizen to become literate. Some futurists believe our society is moving into an information oriented phase in which computer skills are essential skills. Computer usage, like reading and writing, will change from a task performed by a small, highly trained segment of the population to an essential skill for everyone. We may not be completely literate if we are not computer literate. The Commodore is an excellent computer with which to become computer literate.

However, it is not a computer you will outgrow. You can spend around \$550 on the basic unit, a cassette recorder, and a few programs to learn how computers operate. Add books and training material for \$50 to \$100 and you have less than a \$700 investment. After a few months of work, you may decide to use the computer for

more serious applications. You need not banish the Commodore 64. You can add accessories for less than \$1000, and the Commodore 64 becomes a powerful computer that performs many important tasks in your home, business, or school.

BASIC PROGRAMMING

Some new computer owners want to learn how to program the computer. Programming involves typing lines of instructions into the computer. In order to program the Commodore 64 you need to know how to speak its programming language, BASIC (Beginners All-Purpose Symbolic Instruction Code). BASIC is the most popular computer language in use today. It is a family of languages with each computer designed to speak a particular dialect. The Commodore 64 uses a fairly standard version also used on other computers manufactured by Commodore.

Three chapters of this book deal with BASIC programming. Commodore sells a package of materials on programming the Commodore 64. Their *Introduction to BASIC*, Part 1 and Part 2 (less than \$30) is an effective method of learning BASIC via computer.

This book focuses on BASIC. There are, however, other languages available for the Commodore 64. At least two versions of PILOT are available as well as an excellent version of Logo. PILOT is used to develop computer assisted instruction programs. Logo is a popular language for children.

MACHINE LANGUAGE/ASSEMBLY LANGUAGE PROGRAMMING

It is also possible to write programs for the Commodore 64 in 6502 machine language. This a versatile but hard to learn language used primarily by commercial programmers who create video game cartridges and word processing programs. Start with BASIC and put off tackling machine language until you have some experience. Commodore's Assembler 64 program lets you create machine language programs via another language (6500 Assembly Language) that is easier to use than pure machine language. Then the program converts your assembly language program into a machine language program the computer can understand.

FUN AND GAMES

If you play all the games available for the Commodore 64 for eight hours a day, it would probably take you a year to play each game once. By then there would be hundreds of new ones. Our guess is that 30 percent of Commodore 64 owners use the computer primarily for fun and games. Several magazines publish at least one or two game programs for the Commodore 64 in each issue (see Chapter 10), and hundreds of companies sell game programs for the Commodore 64. Many of the best programs take full advantage of the color graphics and sound synthesis features on the Commodore 64. Most also let you use the joy stick rather than the keyboard to move players around on the screen.

HOME AND SCHOOL APPLICATIONS

The Commodore 64 can perform many important jobs around the home. It can keep track of tax information, tutor the kids in academic basics, balance your checkbook, and serve as a powerful word processing system to name only a few of its capabilities.

BUSINESS AND PROFESSIONAL APPLICATIONS

Color graphics, a music synthesizer, and joystick connections make the Commodore 64 an ideal home or school computer. It is also an acceptable business computer. It has a good, full-size keyboard, and a large memory. Disk drives and printers needed for many business applications are available as options; and the screen displays 24 lines of 40 characters. An excellent business computer would have a 24 line by 80 character video display, but hundreds of thousands of people use Apple II, PET, and Atari 800 computers every day for business applications. All four computers have the same 24 line by 40 character display format as the 64.

Several companies sell add-on kits to convert the Commodore video display to 24 lines by 80. Even without the larger display, this computer is a very cost effective computer for some very small businesses and families who want a computer that can do professional as well as family oriented work. There are programs for standard financial and record keeping functions (*e.g.*, accounts payable/receivable, inventory, general ledger) and several companies sell powerful word processors for the Commodore 64.

Word processing is a very popular use for the Commodore 64. Several word processing programs are also compatible with the accessories that expand the video display of the Commodore 64. Many work only if you have purchased the 1541 disk drive. You will need a printer connected to the computer so you can print copies of the material you type. A good word processing program will let you compose, edit, correct, and update written material with less effort and more speed than is possible on a typewriter. Essentially, you create your document on the screen of the video display where changes are easily made. When you are ready, the computer prints a copy of the document. You can save the document on a cassette or diskette and load it back into the computer at a later date for revisions or printout.

A word processing system requires much more equipment than the Commodore 64 keyboard. You need a disk drive, extra memory, a printer, and a word processing program. See Chapter 9 for extra equipment; Chapter 10 for word processing programs currently available.

ELECTRONIC SPREADSHEET SOFTWARE

Many business applications need someone to calculate large tables or charts of related figures. Loan amortization tables, salary schedules, insurance fee tables, inventory order records, travel expense forms, and job cost estimates all require many tedious calculations. Electronic spreadsheet programs such as VisiCalc and SuperCalc have earned millions of dollars for the creators because so many business people need a convenient, quick way of doing such jobs. Several very good spreadsheet programs for the Commodore 64 are reviewed in Chapter 9.

TELECOMMUNICATIONS

Telecommunications is one of the fastest growing areas of interest for home computer owners. Via phone lines, you connect your home computer to hundreds, perhaps thousands, of other computers that have been programmed to provide services to personal computer owners. You can sit back in your easy chair, get a comfortable grip on the keyboard and do everything from pay bills to read weather reports for Colorado ski resorts, and dial up local area networks and community bulletin boards which are run by

colleges, computer clubs, amateur radio clubs, and other special interest groups.

You can also tap the services provided by several national "information utilities" which let you view on your screen everything from stock market information to the current AP or UPI newswire. You can purchase products, check on airline schedules, get reviews of current movies, and look for research or articles on topics of interest to you. Access to the *World Book Encyclopedia* is available on one popular information utility called Source.

Here are the addresses of the most active information utilities:

The Source (1616 Anderson Road, McLean, Virginia 22102, 703/821-6660).

CompuServe, Inc. (5000 Arlington Centre Boulevard, Columbus, Ohio 43220, 614/457-8600).

Bibliographic Retrieval Services (1200 Route 7, Latham, New York 12110, 518/783-1161).

DIALOG Information Services (3460 Hillview Avenue, Palo Alto, California 94304, 415/858-3810).

For a particular type of information, *Directory of Online Databases* (Cuadra Associates) or *Directory of Online Information Resources* (CSG Press) lists hundreds of information utilities you can contact via computer.

One of the most popular information utilities, CompuServe, is also the host for the Commodore Information Network. That service gives you access to up to date information on Commodore products and programs. You can even type in a question and get an answer from a Commodore technician!

If you want to use your Commodore 64 for telecomputing you will need a modem, software, and an account number. Get an account number from the information utility you plan to use. There is usually a one-time registration fee and a monthly bill for the time used that month. The other item, a modem, converts signals from the computer to signals that can be transmitted over telephone lines. Modems cost between \$100 and \$600 depending on features and the profit margin of the seller. Commodore sells the VICmodem for \$100. It plugs into your Commodore 64 and comes with the software needed to let the 64 communicate with most popular information utilities. You even get some free time on a CompuServe.

CHAPTER 2

Setup and Installation

Even the basic Commodore 64 computer comes with accessories. There is the power supply, the cable that connects the television to the computer, and a switch box that lets you switch between regular TV reception and the computer display. This chapter will show you how to connect everything together and how to adjust the television set. If you have purchased a cassette recorder or a disk drive for your Commodore 64, you will also find instructions for installing these accessories.

CONNECTING THE COMPUTER TO THE TELEVISION

The Commodore 64 connects to two different types of video displays. First, it has all the equipment needed to display information on an ordinary color or black and white television set. Most people attach the Commodore 64 to a color set to use the computer's color graphics features.

The Commodore 64 also connects to a color video monitor that produces much sharper color images than an ordinary color television. It costs more than a television. This book concentrates on connecting your Commodore 64 to a color television, since most homes have one already on hand. However, Commodore 64 computers produce less than perfect video displays when connected to a regular color television.

At the back of your computer, you will see five different connectors. The wide recessed slot is where you plug in cartridges. Next is the plug for output to your television. Then there is a round connector called the Audio/Video Connector. This connector has five holes in a semicircle at the bottom and an alignment slot at the

top of the circle. These round connectors, frequently used on European stereo systems, accept a 5 pin DIN connector. If you use a color monitor with the Commodore 64 instead of a television, you will need a cable with a DIN plug on one end and connectors on the other end for audio and video signals that correctly mate with input to your color monitor. There is another DIN plug with six holes rather than five. It is the serial I/O port and connects the disk drive and some printers to the computer. Beside the serial port is the cassette connection. On the end is a larger connector slot called the "User Port," which connects many different types of accessories to the Commodore 64.

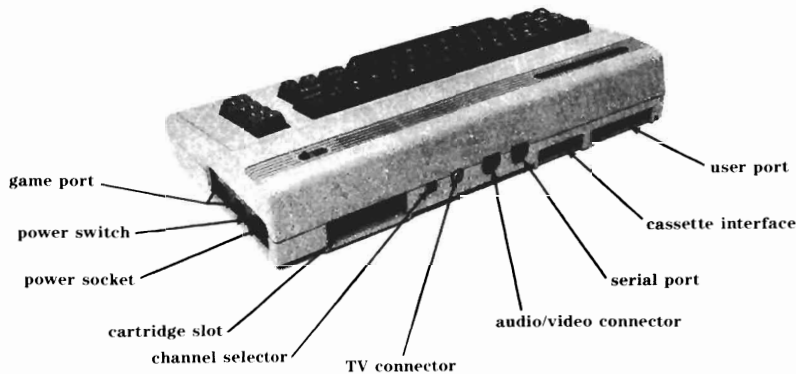


Figure 2.1 The back of the Commodore 64.

You also need a cable about three feet long with an "RCA phone plug" style jack on each end. The Commodore 64 has a built-in RF modulator that converts the sound and video signals produced by the computer to a signal the television can receive on either channel 3 or 4. A switch on the back of the computer between the cartridge slot and the modulator output lets you select the channel. If your city does not have a television station of channel 3 or 4, the switch will work fine on either one. If your area has a station on channel 3 or 4 select an unused channel to minimize interference. If you have stations on both 3 and 4, select the weaker station. If the switch on our computer is not labelled, you may have to experiment.

Plug the cord with the RCA phone plug jacks into the modulator output on the back of the computer. Now attach the TV switch box to the VHF connector on the back of the television. If you have an antenna lead or cable line connected to the VHF terminals, remove it and attach it to the switch box. Use the terminals marked "antenna." Now, connect the cord to the TV switch box.



Figure 2.2

The Commodore 64 television and power supply connectors.

With the television connected to the computer, you can now use the slide switch lever on the TV switch box to select the signal from the computer or from the antenna/cable lead. Set the switch box at the back of the television to "computer" for now.

The Commodore 64, like many computers, generates a lot of electrical noise that can be picked up as interference in the picture. Commodore's television interface circuit design seems to have a problem because the computer provides beautiful color displays when connected to a color monitor. This problem may be corrected in later production runs. When you get your Commodore 64 up and running, move the cords around to find a position that reduces interference. If you plan to buy a small television to use with the Commodore 64, take the computer to the store and test it before

making the purchase, or get permission to return it if it doesn't work well. Or buy the Commodore color monitor (\$300 list price) that eliminates the interference problems and provides a superior video display.

THE TAPE RECORDER CONNECTION

It is not absolutely necessary to buy a tape recorder to use the Commodore 64, but if you plan to load taped programs into the computer's memory or save programs on tape, you might as well set it up now.

Commodore manufactures a special tape recorder for all of its computers. There are several different versions, but the current model (C2N) is white with a sculpted case and it retails for around \$75. Standard tape recorders tend to be far less reliable than the

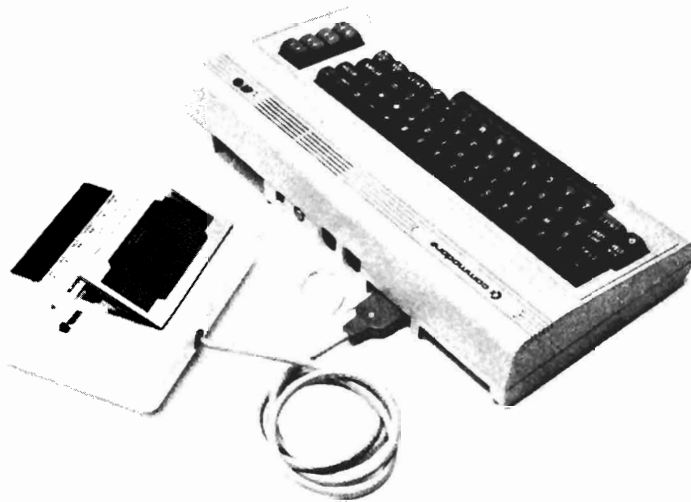


Figure 2.3 The computer recorder connections.

Commodore recorder. See Chapter 10 for where to buy a converter if you want to use a standard converter.

To attach the C2N recorder to the 64, find the grey cable on to the recorder. The flat plug on the end has six metal connector traces in it. Between the second and third connector is a tiny sliver of white plastic. On the back of the Commodore 64 the connector next to the unused DIN plug has copper traces with a slot between the second

and third trace. Orient the cable to insert the plug into the 64 with the white piece of plastic fitted firmly into the slot between trace 2 and 3. The side of the plug with the embossed Commodore logo is up. Since the white plastic slivers wear with use, mark "top" on the cable end now.

Your recorder is now properly connected. Since it gets its power from the computer, it does not need to be plugged into a wall outlet. A bare grounding wire on the cassette plug is not used with the Commodore 64. The wire can accidentally brush against the connectors on the Commodore 64 and damage the computer. Tape the wire to the cable so it will be out of the way.

INSTALLING A DISK DRIVE

See Chapter 4 for disk drive installation.

THE POWER CONNECTION

There is one more cable to be attached to the Commodore 64. Take the power supply cable and plug one end into the connector on the back, right side of the Commodore 64. Then plug the other end into a suitable wall outlet. Now plug in the television set and arrange the computer so that it is convenient for you to use while viewing the television screen. That's it!

TAKE IT FOR A SPIN

All the connections have now been made (unless you splurged on a printer, too). Turn on the television and select channel 3 or 4, whichever is unused or least used in your area. Now turn on the Commodore 64 with the ON/OFF switch on the right side. The screen should tell you that you are working with Commodore 64 BASIC V2, which translates to "Commodore Business Machines Model 64 BASIC, version 2.0." A second line will indicate "64K RAM System 38911 BASIC Bytes Free" which means you have room in the computer's memory to store 38911 characters. A third line tells you the computer is "READY."

If you don't have a display on the screen move the switch on the back of the Commodore 64 to change the channel. You should get the proper picture. If the picture is wavy or fuzzy, adjust fine tuning and brightness/contrast controls.

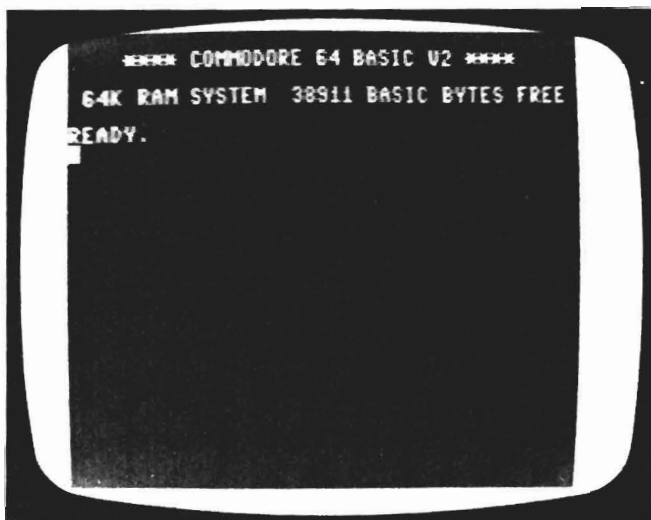


Figure 2.4 The initial screen display.

If you still have problems, turn the computer off and check all your connections. Are they correct? If all else fails try another television. When you get a display that looks like Figure 2.1, you are ready to begin.

THE ESSENTIAL COMMODORE 64

Before you begin using the computer, take a quick tour around the Commodore 64. It is a small but very powerful machine with a system that is made up of several basic components. Figure 2.5 is a block diagram of the VIC computer. Everything outside the dotted line, except the keyboard, is an option you must supply yourself.

THE POWER SUPPLY

The Commodore 64 runs on 5 volt, direct current power. The black power supply unit takes the 110 volts, alternating current, from your wall outlet and converts it to 9 volts. When you plug the power supply unit into the computer a voltage regulator changes the 9 volts to 5 volts DC which is then routed to all circuits on the board that need it.

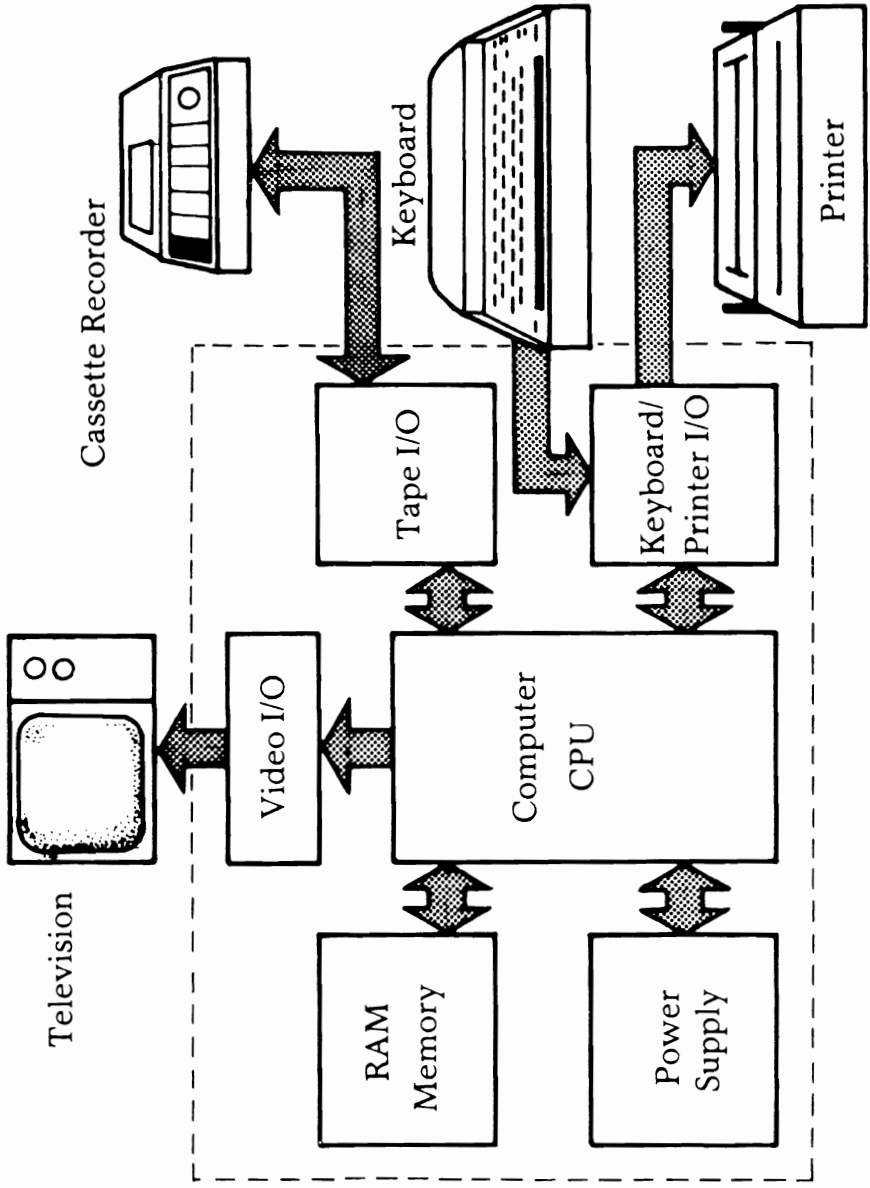


Figure 2.5 A block diagram of the Commodore 64.

I/O PORTS

I/O is an abbreviation for input/output. If a computer is to be useful, it must be able to receive information and communicate its response back to you. This basic function is called Input/Output. The places on the computer circuit board where I/O occurs are often called "ports." The Commodore 64 has several ports that allow you and the computer to communicate. The keyboard is your primary means of inputting information and instructions to the computer.

The Commodore 64 usually talks back to you via the television. It converts its messages to a video (and audio) signal that exits the computer via the connectors for the television or video monitor.

In addition to I/O ports for the keyboard and a television or monitor, the Commodore 64 can be connected to a tape recorder (back panel 6-trace edge connector), a disk drive and/or printer (6 pin DIN plug on back), and a joystick or light pen (9 pin "D" connectors on right side).

The cassette (or disk drive) I/O circuits let you store programs on a standard audio cassette tape or "floppy diskette," and then "load" the program back into the computer when you want to use it again. A program is a set of instructions the computer follows to accomplish a particular task (*e.g.*, balance your checkbook or play a game of chess). The computer reads the instructions in a program and does what they tell it to do. For the Commodore 64 instructions are stored on the cassette in a code that is made up of a series of tones. If you listen to a tape, you hear only a buzzing sound. That is, however, music (or instructions) to the ear of the computer. If you use diskettes to store program instructions, they are stored as a series of electrical signals on the magnetic surface of the diskette.

There are two more I/O ports on the Commodore 64. One is the Cartridge Port which is on the back that lets you insert cartridges like those on video games. These cartridges contain additional memory. Some cartridges contain a type of memory that can be used to store programs. Other cartridges have "preprogrammed" memory that contains instructions for the computer. Several video games, for example, for the Commodore 64 computer are available in cartridge form. To use them you plug them into the Cartridge Port and turn the machine on. The instructions in the memory of the cartridge then control how the Commodore 64 operates.

Finally, the Commodore 64 has a "User Port" on the back next to the cassette port. The user port is called an "expansion buss" or

“expansion port” on some computers. It is a convenient means of connecting several types of accessories to the computer.

MEMORY

When you type something on the keyboard, or load a program into the computer from a cassette, there must be a place to put that information. Each letter or number you type on the keyboard is converted to a code and stored in the computer’s memory. Each character has its own code, a series of ones and zeros. A few of the codes are shown here:

Keyboard Character	Internal Code
A	01000001
B	01000010
1	00110001
2	00110010

The universal code today is ASCII, or American Standard Code for Information Interchange. Most small computers use this code to translate characters and symbols into signals that can be stored in their memory.

Each of the ones and zeros in the codes shown above is called a “bit.” Seven of those bits define the code for each letter or character. Another bit, the eighth, is usually added to the code for each character so the computer can check for errors. This process, called “parity checking” will not be discussed here. Those eight bits are called a “byte.” Bytes are the fundamental code units for the Commodore 64 and for most other small computers. Memory inside the computer is also divided into bytes. One byte of memory can hold the electrical impulses that represent eight ones and zeros.

The Commodore 64 computer contains two types of memory, RAM and ROM. ROM stands for Read Only Memory, which is usually programmed at the factory and cannot be changed by the user. The Commodore 64’s ROM contains thousands of instructions for the computer. If there were no preprogrammed ROM you might have to give instructions in patterns of ones and zeros. Fortunately, the computer automatically begins to follow the instructions in its ROM. Instead of ones and zeros, you can use English-like words in the BASIC computer language.

All computer memory cannot be ROM, however. Much of the memory in the Commodore 64 is RAM, or Random Access Memory. The standard Commodore 64 has a little over 64,000 bytes of RAM. Since each byte can store the code for one character, the 64,000 bytes of RAM can theoretically hold up to 64,000 characters. The 64 absorbs some of its RAM for its own housekeeping work, which means you have less. In addition, the Commodore 64's circuits are designed so that when the computer operates under control of the BASIC language in its ROM, only a little over 38,000 bytes of memory are available for use. However, this exceeds the capacity of other small computers, which have as little as 1,000 bytes of memory for use.

RAM is also known as "volatile" memory. It is general-purpose memory. You can store data or instructions in RAM, tell the computer to use the information you've stored there, and then replace the material in RAM with something new. You can put data in RAM (write to memory), and you can take it out (read from memory). You can only read ROM. The biggest problem with RAM is the fact that whatever is in it disappears when the computer is turned off. If you need to save something in RAM for later use, you must store it on a cassette (or diskette) before turning the computer off. Material in ROM, on the other hand, remains there forever but cannot be changed or modified.

Computers have 5K of RAM, or 32K, or 128K, even 512K. Each K of memory is 1024 bytes. Thus 16K would be 1024 times 16 or 16,384 bytes. To determine the number of bytes of memory multiply the number of K by 1024.

THE CPU

The CPU, or central processing unit, is the heart of a computer system. Although most CPU chips are smaller than an Oreo cookie, the electronic components they contain would have filled a room a few decades ago. Using advanced microelectronic techniques, manufacturers cram thousands of circuits into tiny silicon chips that work dependably and use less power than an electric razor.

The CPU does most of the actual processing inside the computer and sends signals to the other parts of the computer to synchronize operation of each component and permit the system to perform thousands of operations a second. These operations are fairly simple (*e.g.*, adding two numbers together), but the speed at which

they are done allows the computer to do some very complex things by breaking complicated jobs down into hundreds of simple steps.

The Commodore 64 uses the 6510 microprocessor chip manufactured by MOS Technology. This “computer on a chip” is an enhanced version of the 6502 chip used in the VIC, Apple II, PET, and Atari 800/400 computers. The 6510 understands the same set of instructions as the 6502 and a few more. In addition, this chip can use more memory than is generally feasible with the 6502. In the Commodore 64 the 6510 chip has some help from two other chips, the 6566 Video Display Chip and the 6581 Sound Interface Device. The 6566 chip is a specially designed integrated circuit that does most of the work to create and produce the computer’s color video display. The 6581 handles sound and music synthesis. In older personal computers, sound, graphics, and actual computing were all performed by a very busy 6502 chip. In the Commodore 64, three chips divide the work with the result being a much faster operating computer and one that has very sophisticated sound and graphics features. (Chapter 8 discusses these features in detail.)

CHAPTER 3

How to Get Your Computer Up and Running

At this point you should have all your cables correctly connected with the computer and the television plugged in. If they are not on, switch the Commodore 64 and the television on with the On/Off switch on the right side. If everything is operating correctly you should now have the initial screen display photo. If it is not, return to Chapter 2 for instructions and suggestions.

THE KEYBOARD/DISPLAY CONNECTION

The keyboard on the Commodore 64 uses the layout on standard typewriters and should be familiar to any touch typist. The top row of letters begins with Q on the left and has WERTYUIOP across. Figure 3.1 illustrates the keyboard.

The Commodore 64 keyboard is busier than a typewriter keyboard, however. For example, the A key, like most of the keys, has a letter printed in the middle of the keypad. There are also two graphic symbols on the front of the A key. This one key can produce four different symbols on the screen—a lowercase a, a capital A, and two graphic symbols.

The Commodore 64 keyboard contains a number of keypads with unfamiliar legends. There are keys with arrows on them, keys with words or phrases such as RUN/STOP, CLR/HOME, INST/DEL, and RESTORE, and there are keys with symbols such as the Commodore logo (bottom left) and arrows (top left and bottom right). Finally, there are four keys on the right labelled F1 through F8. You will learn the functions of all the Commodore 64 keys in the next few chapters.



Figure 3.1 The Commodore 64 keyboard.

YOUR FIRST PROGRAM

Now let's work with the keyboard. Turn the computer on. Now type in the following after reading the Note just below the material to be typed:

```
10 INPUT N$
```

Note: The Commodore 64 normally produces capital letters without the necessity of holding down the shift key. The line above can be typed correctly without using the shift key until you need the \$ symbol.

Now press the **RETURN** key. This key tells the computer you are finished typing material on a particular line. The Commodore 64 will move the blue square which is called a "cursor" down one line. It should be flashing now on the left side of the screen. The Commodore 64 is now waiting for you to type in more information.

IF YOU MAKE A MISTAKE

If you make errors as you type this first program, there are several things you can do to correct them.

Before Pressing RETURN

If you spot an error before pressing the **RETURN** key, you can use the **RIGHT ARROW** and **LEFT ARROW** keys to move the cursor around on the line you are writing. The cursor is the blue square that tells you where you are on the screen. The **RIGHT ARROW** and **LEFT ARROW** keys are on the bottom right of the keyboard. Note that one key has an arrow that points to the right and one that points to the left. If you press this key, the cursor will move to the right. If you hold down the **SHIFT** key and press it, the cursor will move to the left. Use the **SHIFT** and Right/Left Arrow key to move the cursor to the point in your line where the error occurred. For example, if you typed 10 INPUR N and pressed the R key instead of the T, hold down the **SHIFT** key and then press the Right/Left Arrow key until the cursor is in the space just to the right of the R. Now find the **INST/DEL** key at the top right of the keyboard. Press this key once. (DEL is short for delete.) Pressing it with the cursor to the right of the R should make the R disappear. Now hold down the **SHIFT** key and press the **INST/DEL** key again. (INST is short for INSERT.) When you deleted the R the computer moved material on the right over one space. Pressing the **INST** key once (actually **SHIFT INST/DEL**) told the computer to make room for a character. Now type a T, use the Right/Left Arrow key to move to cursor to just past N and type a \$.

You can use the Right/Left Arrow key and the **INST/DEL** key to correct many errors. If you hold down the Right/Left Arrow key the cursor will continue moving in the direction of the arrow. If it gets to the edge of a line, the cursor will wrap around to the next line. Try it and see how the cursor travels.

After Pressing RETURN

If you do not notice an error until you have already pressed the **RETURN** key, there are several things you can do.

“Retype the Line.” Suppose you typed in “10 IBPUT N\$” and pressed the **RETURN** key before noticing the error. If you type in “10 INPUT N\$” and press the **RETURN** key again the new version of line 10 will replace the old version in the memory of the computer. This is often the simplest and easiest approach to correcting an error. If you accidentally type in a line that shouldn’t be there at all, just type the number of the line and press **RETURN**.

That causes the entire line to be deleted from the program you are writing (but not from the screen). For example, if you accidentally put a line 99 into the program, typing 99 and pressing **RETURN** will delete the line. There is another process called "editing" that we will deal with later. You can correct any mistakes you make, however, with the methods described above.

Now type in the following (Note: Press the **RETURN** key when you finish typing each line and remember that you get capital letters without pressing the shift key.):

```
20 PRINT N$; " CAN YOU WRITE"  
25 PRINT "COMPUTER PROGRAMS?"  
30 INPUT A$  
40 IFA$ = "YES" THEN GOTO 100  
50 IFA$ = "NO" THEN GOTO 110  
100 PRINT "GREAT, WE'LL HAVE FUN."  
105 GOTO 200  
110 PRINT "NOT TO WORRY, YOU WILL LEARN"  
200 END  
5 PRINT "HI WHAT IS YOUR NAME?"
```

Use the Right/Left Arrow key and the **INST/DEL** key to correct any errors you make while typing the lines above. Because the Commodore 64 puts up to 40 characters on each line of the screen display, all the program lines above will fit on one line. A program line, however, can be as long as 88 characters and may fill several lines on the screen. When you typed line 110, which has exactly 40 characters, the cursor automatically moved to the next line when you typed the last quotation mark. Even though the cursor is on another line, you must still press the **RETURN** key to tell the computer you are finished with that program line because a program line can be much longer than a screen display line.

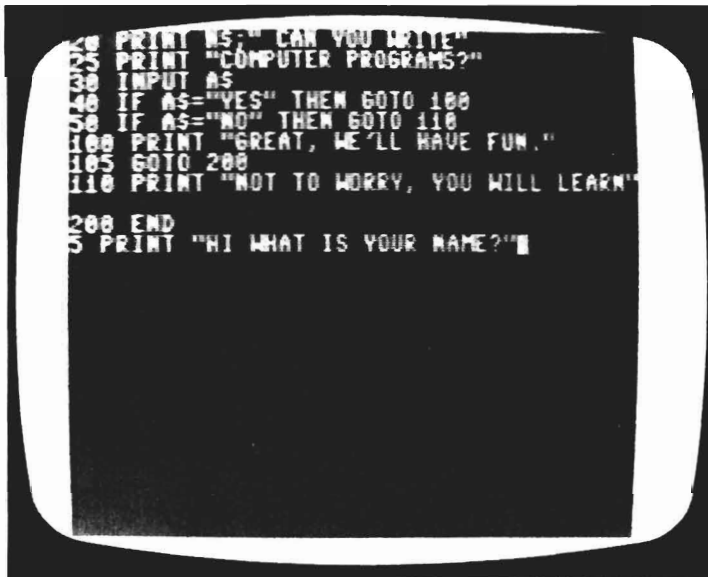
YOU HAVE A PROGRAM!

At this point your first program is in the computer. Now hold down the **SHIFT** key and press the **CLR/HOME** key. That will **CLEAR** the screen. You should have a completely blank screen with a flashing cursor in the upper lefthand corner. The key word **CLEAR** (**SHIFT CLR/HOME**) tells the computer you want the material on the screen erased and the cursor moved to the **HOME** position (top left corner). If you press **CLR/HOME** without holding

down the **SHIFT** key, the cursor will move to the home position but the screen will not be erased.

Now type the word **LIST** and press **RETURN**. Your program should be listed on the screen in numerical order. Even though you typed line 5 after all the other lines it is placed first on the screen since it has the smallest line number. Lines 5, 20, 25, 40, 50, 100, and 110 are too long to be displayed on one line on your screen.

The material you have just typed in is a computer program which was written in BASIC, a very popular computer language. The Commodore 64 uses a relatively powerful dialect of BASIC which has much in common with the versions of BASIC used in other personal computers such as the Apple II, Atari 800/400, and Radio Shack Color Computer. The Commodore 64's version of BASIC is very close to the BASIC used on other Commodore computers. If you learn BASIC on this Commodore 64 it will not be difficult to learn the version used on other popular computers should you move on to another model.



```
20 PRINT AS: " CAN YOU WRITE"  
25 PRINT "COMPUTER PROGRAMS?"  
30 INPUT AS  
40 IF AS="YES" THEN GOTO 100  
50 IF AS="NO" THEN GOTO 110  
100 PRINT "GREAT, WE'LL HAVE FUN."  
105 GOTO 200  
110 PRINT "NOT TO WORRY, YOU WILL LEARN"  
  
200 END  
5 PRINT "HI WHAT IS YOUR NAME?"
```

Figure 3.2 The screen display after the program is typed in.

A program is a set of instructions you want the computer to follow. The instructions are organized into program lines, and each line begins with a line number. When the computer is told to follow or "execute" the instructions in a program it begins with the

instructions in the line with the smallest number. It executes the instructions in that line first; then it moves on to the line with the next smallest number and executes the instructions there. It thus moves sequentially through the program, from lowest line number to highest line number. If you do not want the computer to follow this standard or "default" pattern, you must tell the computer what line you want it to do next. If you don't, it will move sequentially through the program.

How do you get the computer to execute your program? It is really very simple. The BASIC version used by the Commodore 64 has a key word **RUN** to tell the computer to follow the instructions in the program currently in its memory. To run your first program hold down the **SHIFT** key and press the **CLR/HOME** key. That will clear the screen and put the cursor in home position. Now type **RUN** and press **RETURN**. The sentence **HI WHAT IS YOUR NAME?** appears at the top of the screen. On the line below the question a **?** appears along with the cursor. Type in your first name and press **RETURN**. If you type in **JERRY** and press **RETURN**, the computer will reply, **JERRY CAN YOU WRITE COMPUTER PROGRAMS?** Another **?** indicates the computer is waiting for you to answer the question. Type in either **YES** or **NO** and press **RETURN**. Let's say you type in **NO**. The computer should respond with **NOT TO WORRY, YOU WILL LEARN**.

You have now typed in a program and it has been **RUN** or executed by the computer. If it didn't work as expected, look back over the lines you typed in and try to find the problem. For example, if the number 0 was typed for the letter O in line 50 the program won't work correctly when you type **NO**. When the program runs correctly, read the explanation of each line in the section that follows.

A LINE BY LINE EXPLANATION OF THE PROGRAM

Chapters 6, 7 and 8 deal with the BASIC language used by the Commodore 64 in greater detail. This section will explain what happens as each line of your first program is executed.

5 PRINT "HI, WHAT IS YOUR NAME?"

This line contains the key word **PRINT** and the sentence **HI, WHAT IS YOUR NAME?** The key word **PRINT** instructs the computer to print whatever appears on the line just after **PRINT**.

The quotation marks before and after the sentence tell the computer to print the material just as it appears in the program.

```
10 INPUT N$
```

The key word **INPUT** tells the computer to look for something to be typed in on the keyboard. If you type in **AMY** and press **RETURN** the computer sets aside a section of its memory and puts the characters **AMY** there. It then makes **N\$** equal to **AMY**. **N\$** is a “variable name” and **AMY** is the content of that variable. Later in the program if you tell the computer to **PRINT N\$** the computer will print **AMY** rather than **N\$**. On the other hand, if you tell the computer to **PRINT “N?”** the computer would actually print **N\$** because you enclosed it in quotation marks.

```
20 PRINT N$“ CAN YOU WRITE”  
25 PRINT “COMPUTER PROGRAMS?”
```

The key word in both lines is **PRINT**. Since **N\$** is not enclosed in quotation marks the computer checks to see what **N\$** stands for. If it is **AMY**, that is what is printed on the screen. The rest of the phrase in line 20 is between quotation marks and will be printed as is. All of the material after **PRINT** in line 25 is within quotation marks so the sentence **AMY CAN YOU WRITE COMPUTER PROGRAMS?** will appear on the screen with **AMY CAN YOU WRITE** appearing on one line and **COMPUTER PROGRAMS** on the next.

```
30 INPUT A$
```

This line is just like line 10 except that whatever you type in will be assigned to the variable name **A\$** rather than **N\$**. We chose the letter **A** because this is an answer to a question. The **N** in **N\$** is short for name.

```
40 IF A$ = “YES” THEN GOTO 100
```

There are three key words in this line: **IF**, **THEN**, and **GOTO**. The line tells the computer to check what **A\$** stands for. If **A\$** equals **YES** then the instruction after **THEN** is executed. In line 40 the key word **GOTO** instructs the computer to skip to line 100 and execute the instructions there **IF** and only if **A\$** equals **YES**.

```
50 IF A$ = “NO” THEN GOTO 110
```

This line, like line 40, tells the computer to check what A\$ equals. If A\$ equals NO, the computer jumps to line 110 for instructions. If A\$ equals YES, the computer never reads or executes the instructions in line 50 since it jumps from line 40 to line 100. If A\$ equals NO, the computer never reads or executes the instructions in lines 100 and 105 since it jumps from 50 to 110.

```
100 PRINT "GREAT, WE'LL HAVE FUN."
```

The key word in line 100 is **PRINT**. The line causes the sentence in quotation marks to be printed on your screen.

```
105 GOTO 200
```

If A\$ equals YES and line 100 prints its cheery comment, the program has accomplished its purpose. You don't want the other comment in line 110 to be printed so **GOTO 200** sends the computer from line 105 to line 200.

```
110 PRINT "NOT TO WORRY, YOU WILL LEARN."
```

If A\$ equals NO, the computer jumps from line 50 to line 110 where an encouraging comment is printed on the screen.

```
200 END
```

The key word **END** tells the computer that, once it gets to line 200, it has finished its work. It stops executing instructions and prints **READY** which indicates it is waiting for additional instructions. **END** is optional in many programs because the computer will also stop and tell you it is **READY** when it has executed the instructions in the line with the largest line number.

At this point you may be thinking that writing programs for a computer is far more complicated than this. True, there are some key words that are more difficult to understand than the ones used so far. And there are complex, sophisticated procedures used by professional programmers that can't be picked up easily in a weekend. But it is also true that you can quickly and easily learn to write your own programs in a language like BASIC. The skills you learn are some of the same ones used by programmers who spend most of their waking hours at the keyboard of a computer. In addition, if you learn the key words in BASIC and how programs are written, you will be able to use programs from magazines and adapt and personalize programs.

GRAPHICS AND THE 64

The Commodore 64 displays a number of graphics characters, as well as letters, numbers, and symbols. The graphics symbols shown on the keyboard are obtained by holding down either the **SHIFT** key or the Commodore Logo key and pressing the key with the desired graphic symbol embossed on the front. If, for example, you hold down the Commodore Logo key (bottom left of keyboard) and press the **B** key you get a small checkerboard pattern. Hold down the **SHIFT** key and press the **B** key to produce a thin vertical line. Many of the keys on the Commodore 64 keyboard can produce two very different graphics symbols. The graphics symbol on the left side of the keyfront is obtained by holding down the Commodore Logo key while the symbol on the right is produced by holding down the **SHIFT** key.

You may want to play with the keyboard a bit to get a feel of how graphics symbols are produced and how they look on the screen. When you have finished clear the screen with the **CLR/HOME** key. Then type **NEW** and press **RETURN**. **NEW** is a key word that tells the computer to erase whatever program is in its memory. **NEW** thus clears out the computer's memory so that you can type in a new program. If you type in lines for a new program without using **NEW** some of the lines from the old program might remain and cause confusion when you tried to run the new program. Now type in the following program:

```
10 FOR X = 1 TO 800
20 PRINT "♠";
30 NEXT X
```

The graphics symbol in line 20 is obtained by holding down the Commodore Logo key and pressing the **X** key. **RUN** the program and you should get 800 blue spades on the screen. Since there is space for 24 lines of 40 characters or 960 characters on the screen, this program puts spades in all except 160 locations.

This simple program illustrates another feature of the 64. There is a large key labelled **CTRL** on the left side of the keyboard. Hold it down and press the **3** key. Did the color of the cursor change to red? Now run the program again. This time you should have a screen of 800 red rather than white spades. You can change the color of the cursor and of the material printed on the screen by holding down the **CTRL** key and pressing a number from 1 to 8. Try this again

with another number (*e.g.*, color) to see how it works. In addition, retype line 20 using different graphics symbols to see how they look in different colors. Note, however, that the background color of the display screen is blue. If you press CTRL-7 you will make the foreground (*e.g.*, the characters to be printed) blue too. They will be there but you won't be able to see them.

The Commodore 64 lets you control the color of the border, the background, and the foreground. More information on color and graphics, as well as sound and music and key words, will be presented in Chapter 8.

CHAPTER 4

LOADing Information from Cassette or Disk

LOADing is the process of getting information stored on either a cassette or diskette back into the computer's memory. This chapter explains how to **LOAD** programs using cassettes or diskettes. Chapter 5 describes how to **SAVE** programs using a cassette or diskette and how to use the printer.

CASSETTE STORAGE

Making the Connection

To use a cassette player with the Commodore 64, you must connect the cable from the recorder to the back of the computer. (See Chapter 2 if your recorder and computer are not connected.)

Examine the recorder for a minute. Two very helpful features of the Commodore recorder are the memory counter and the **SAVE** indicator (see Chapter 5).

The Commodore recorder has another valuable built-in feature. Commodore preset the correct volume and tone levels in their specially designed recorder so there is no volume and tone adjustment. It's a reliable cassette.

When you turn on the television and the Commodore 64, the screen looks like Figure 4.1.

Once the cassette is connected and the screen looks like Figure 4.1, you can **LOAD** programs into the memory of your Commodore 64.

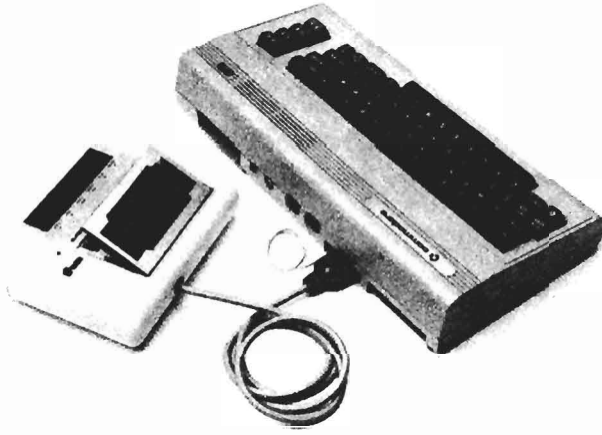


Figure 4.1

The C2N recorder plugged into the Commodore 64.

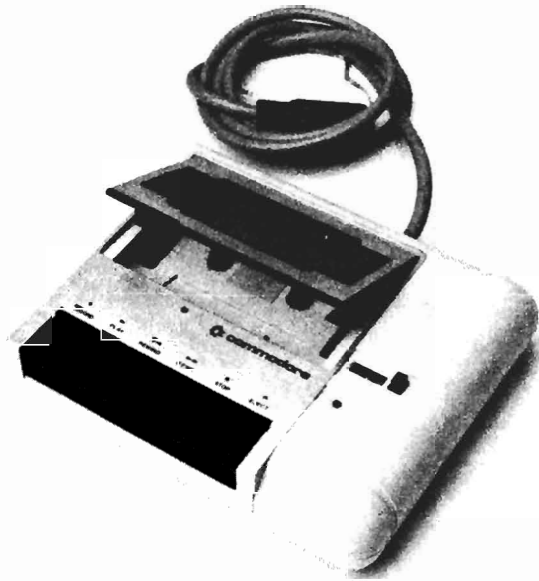


Figure 4.2 The C2N cassette recorder.



Figure 4.3 The initial screen display.

LOADing Steps

The easiest way to **LOAD** a program from cassette uses the **SHIFT** and **RUN/STOP** keys. Follow these guidelines to **LOAD** the first program on a tape:

- rewind the tape completely.
- hold down the **SHIFT** key and press the **RUN/STOP** key.

The screen reports:

PRESS PLAY ON TAPE

When the **PLAY** button on the recorder is pressed, the screen displays:

OK
SEARCHING

and when the first program, name it SUM, is located, the message on the screen is:

FOUND SUM
LOADING
READY
RUN

and the program begins to do what it is supposed to. Using this method to **LOAD** does two things. First, the program is loaded into the memory of the computer. Second, the program is automatically **RUN** or executed. In the following instruction, you must type **RUN** to execute the program. There may be times when you don't want the program to execute automatically (you might want to get the list of the program line by line, for example, but most of the time **SHIFT** and **RUN/STOP** are a quick and easy way to get a program up and running.

Suppose you have a cassette that contains several programs. The first program, called **TEST**, is the one you want to transfer from tape into the computer. The following steps show you how to **LOAD** the program into memory:

STEP 1. Rewind the tape in the recorder to the beginning of the tape.

STEP 2. Type

LOAD "TEST"

and press the **RETURN** key on the keyboard. Be sure to type quotation marks (" ") before and after the name of the program. If these are left out, the following message appears:

```
?TYPE MISMATCH  
ERROR  
READY
```

To correct this mistake, retype **LOAD**, the quotation mark, name of the program, and the last quotation mark. Then press **RETURN**.

It is also important to spell the name of the program correctly. When the computer reads information on the tape, it looks specifically for the name of the program you typed. For example, if **TRST** is typed instead of **TEST**, the Commodore 64 searches the entire tape looking for a program named **TRST** which doesn't exist.

If you discover an error before pressing **ENTER**, use the **DELETE** key to backup and type the correction.

After the name of the program is correctly entered and you press the **RETURN** key, the screen displays a message to tell you what to do next. That message is:

PRESS PLAY ON TAPE

STEP 3. Press the **PLAY** button on the recorder. As soon as you press **PLAY**, the screen displays:

OK
SEARCHING FOR TEST

to let you know the computer is looking for the program you specified. The Commodore 64 lets you know when it finds the program by displaying:

FOUND TEST
LOADING
READY

This means the program has been read, or transferred, from cassette into the memory of the computer. The **READY** message appears when the entire program is in memory so it may be a few seconds before it appears.

STEP 4. To execute the program, type

RUN

press the **RETURN** key, and that's all there is to it! The program is ready for you to use.

There are a couple of other ways to type instructions to **LOAD** programs. The instructions above are probably the safest, but what if you forget the name of the program you want to **LOAD**? Can you **LOAD** a program without knowing the exact name? Yes, you can. If the program you want is the first one on the tape, the format is as follows:

LOAD "

(notice only one set of quotation marks) or:

LOAD

with no quotation marks. This only works, though, if the program you are looking for is first on the tape. The instruction tells the computer to **LOAD** the first program it finds. You could go through this procedure several times to locate your program, but it gets to be very time consuming.

What if you don't remember the name of the program and it isn't first? Suppose, for example, that you have a cassette with the programs **COLOR**, **DEMO**, **TEST**, and **SAMPLE** on it and **TEST** is

the one you want to **LOAD**. There are two ways to deal with this. If you know the name of the program, you can follow steps 1 through 3 above. As the computer reads the tape and finds the first program it lets you know what is happening by displaying:

```
SEARCHING FOR TEST  
FOUND COLOR
```

As the tape progresses and the computer locates the next program, the display is:

```
FOUND DEMO
```

When **TEST** is found, it is **LOADed** and you can go to Step 4 (type **RUN**) to use the program.

That method is fine if **COLOR** and **DEMO** are short programs. Some programs, however, can take five minutes or longer to be read and that is a long time when you are sitting at the keyboard waiting for **TEST** to be located. For lengthy programs, you may want to use an alternative method.

Using steps 1 and 2 above, you will need to go through the entire tape at least once using the counter on the recorder. The counter lets you locate where each program begins and ends on the tape. For instance, the program **COLOR** might begin at location 009. When the screen display reports that **COLOR** has been found, write down the numbers on the memory counter (*i.e.*, **COLOR-009**). Each program on the tape should be found and a note made of the ending location. Use the following guidelines for noting tape locations:

- rewind tape completely
- press the counter button so that there are all zeros in the display
- in order to locate all the programs on the tape, tell the computer to **LOAD** the last program (that way the Commodore 64 searches the entire tape, tells you when each program is found), or if you instruct it, searches for a program that doesn't exist and the computer searches all the cassette for that non-existent program
- when the screen display indicates a program has been found, make a note of the program name and its location.

When all of the programs have been found and you have a list of where each one begins and ends, you have a valuable tool for **LOADing** programs later. For example, say **TEST** ends at counter location 57 and **SAMPLE** begins at 60. You can rewind the tape to

the beginning and set the counter to 0. Next, use the fast forward key on the cassette to advance the tape and counter to about 58. Now you are ready to **LOAD SAMPLE** using steps 2-4 above.

DISKETTE STORAGE

Making the Connection

Just as the cassette player must be connected to the Commodore 64, the disk drive must also be properly plugged in. Take the six pin DIN plug on the disk drive's cable and insert it in the matching connector on the back of the computer. Then plug the power cord from the disk drive into a suitable wall outlet. The VIC printer, also used on the model 64, uses the same 6 pin DIN plug on the back of the computer, or it can be plugged into a spare connector on the disk drive.

The power up sequence, or order in which the computer and disk drive are turned on, is very important with the Commodore 64. When the drive is connected, turn it on first. Always make sure the computer is turned on *after* the disk drive.

This section assumes your disk drive is the 1541 model and that you have at least the demo diskette that comes with the drive. Figure 4.2 and Figure 4.3 illustrate a diskette and its various parts as well as the correct way to insert a diskette into a disk drive.

The material you can see through the oval window of the diskette is where information is stored. Treat diskettes with care and *never* touch the plastic platter that is visible in the oval window. Oil from your finger can prevent the computer from reading information stored there. Diskettes are sensitive and respond poorly to dust, cigarette smoke, dirt, or food crumbs. Because the platter inside the protective envelope is covered with a magnetic film, diskettes are also highly sensitive to magnetic fields. Keep them from the tops of television sets and disk drives because the transformers in them generate a magnetic field.

The notch on the right side of the diskette is called the *write protect* notch. If this notch is as it appears in the picture, the diskette is not write protected, and the computer can store or change information on it as well as read the information already stored there. If you have a diskette that contains a valuable program and you don't want anyone to change it, you can place a piece of tape (most diskettes come with tabs for this purpose) over the notch. It's enough here to say that information can not be put on a

diskette while the write protect notch is in place. (See Chapter 5.)

When you insert a diskette into the drive, always put it in with the window edge first as in Figure 4.3, with the label on top. Notice the green light on the left of the drive. This is the power light and indicates when the disk drive is on. The red light next to it has two functions: it comes on when information is being written to or read from the diskette; when there is an error (*i.e.*, program not found), it flashes to let you know there is a problem.

To open the Commodore 1541 disk drive door, press the lip gently in and up with your thumb. Place your thumb on the lip with the rest of your fingers on the top of the drive and press up with your thumb. Slide the diskette in until it seats. To close the door, press down from the same position.

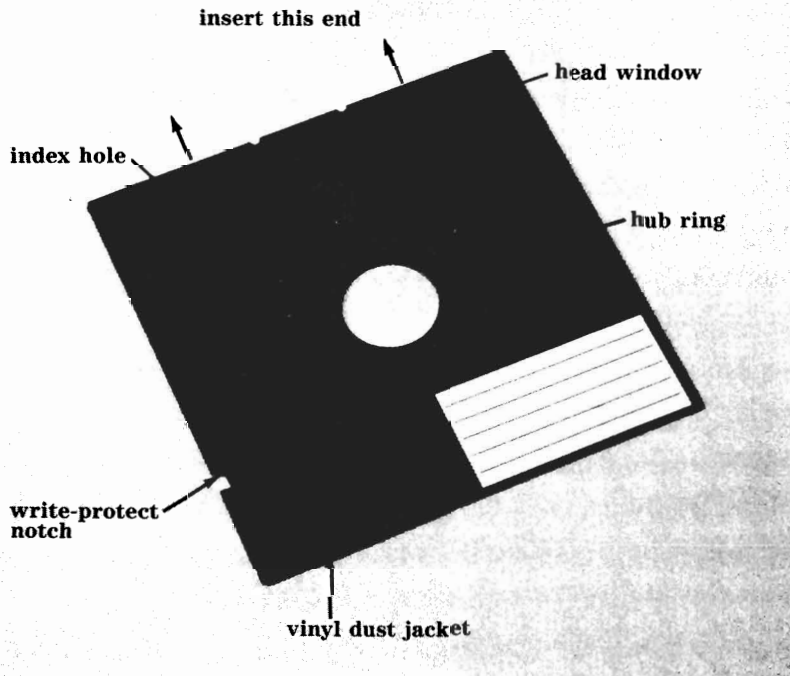


Figure 4.4 A floppy diskette.

LOADING A PROGRAM

LOADING programs from diskette is somewhat like using a cassette, only faster! The Commodore 64 assumes you are using a

cassette. For example, even with the drive connected to the Commodore 64, if you type **LOAD "TEST"** the Commodore 64 responds with the "PRESS PLAY ON TAPE" message because it expects a cassette to be used. You must tell the computer to expect information from the serial port where the drive is connected. Anything that is attached to the Commodore 64 is called a device and has a device number. The cassette player, for example, is device number 1. The disk drive device number is 8 and the printer is 4. Each time you want to get information from a device, you must include the number of the device. But you didn't do that with the cassette you say? Right. The Commodore 64 has a default value of 1; that is, it assumes you are using a cassette and doesn't require a device number with the recorder. It is necessary, however, to include device numbers when using disk drives and printers (more on printers in the next chapter).

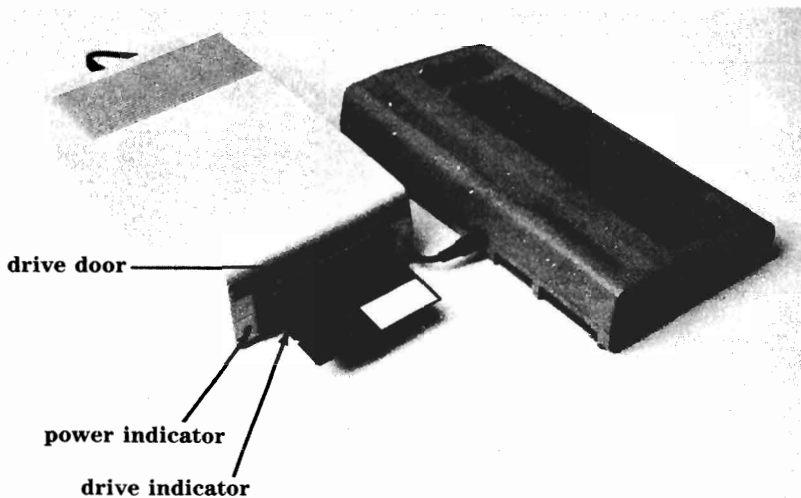


Figure 4.5 The correct method for inserting a diskette into the Commodore 64.

LOADing Instructions

There are really only two steps for **LOADing** programs from diskette. Suppose you have inserted a disk that contains the program you want and have closed the disk drive door securely. To **LOAD** a program named **SAMPLE**:

STEP 1. Type:

LOAD "SAMPLE",8

and press **RETURN**. The screen displays

**SEARCHING FOR SAMPLE
LOADING
READY**

Although the message is the same for the cassette, the disk performs much quicker than the cassette. Notice that the comma comes *after* the quotation mark and is followed by the device number of the disk drive (8). What happens if the comma is placed inside the quotation marks? The display reads:

**? SYNTAX
ERROR
READY**

What if another number besides 8 is entered for the device? The following message appears:

**SEARCHING
? DEVICE NOT PRESENT
ERROR
READY**

If an error is made, however, the Commodore 64 usually tells you what kind of mistake was made and when it is ready for you to take control. All you have to do is retype the instruction. But if the name is misspelled or a file is requested that isn't on the diskette, this message appears:

**? FILE NOT FOUND
ERROR
READY**

At this point, the red light on the disk drive blinks on and off. This happens any time there is an error having to do with the drive. It is a handy little feature to let you know what's going on (or isn't going on in this case).

STEP 2. After the program has been successfully **LOAD**ed into the computer type:

RUN

and press **RETURN**. The program executes. When you are finished with a program and want to empty the 64's memory, type:

NEW

and press **RETURN**. Now you can **LOAD** and **RUN** another program. If you get the message:

? OUT OF MEMORY

there is not enough room to **LOAD** and/or **RUN** the program you requested. If this happens, you can type **NEW** and clear the memory of any programs already there. Always clear the memory between programs to keep programs from accidentally overlapping. Try **LOADing** the program again. If you still get the message that there isn't enough memory, then the program requires more memory than your Commodore 64 has. Make sure the programs you purchase do not require more memory than you have *before* buying the program.

Some of the features of the disk drive that make it preferable to the cassette are evident when it comes to **LOADing** programs. For example, it is necessary to know where on tape a program is located to speed up the **LOADing** process when using a cassette. On a disk, however, you don't have to know where on the disk a program is because the computer takes care of that. Another advantage of a disk drive is the ability to get a list, or directory, of the programs on a diskette that lets you know what programs are on a diskette and about how much room each one will take up in the memory of the computer. Suppose you have a diskette that hasn't been used in a while and you don't remember what's on it. To find out, type:

LOAD "\$",8

and press **RETURN**. This doesn't mean to **LOAD** money into the computer as it might appear. The dollar sign means "directory" to the computer. The instruction then means to **LOAD** the directory from device number 8 (the disk drive). When you press **RETURN**, the screen displays:

SEARCHING FOR \$
LOADING
READY

The disk drive sets aside space on each diskette for the directory. When the screen indicates the computer is ready, the Commodore 64 has read the directory into its memory. In order to see the list of program names, type:

LIST

and press **RETURN**. Quickly the names roll by on the screen. Sometimes the program names go by so quickly you can't read them. The speed can't be changed, but you can control how many flash by. If you want to stop the **LISTing**, press the **RUN/STOP** key. The computer will display the following message after the last program name on the screen:

BREAK
READY

When you're ready to see the rest of the directory, type **LIST** and press **RETURN** again. You can do this as many times as you want.

Take just a minute to look at your directory. While the numbers and names might be different, suppose your directory looks something like this:

```
5 PRG "TINYMON1 FOR C64"  
8 PRG "TINYMON INST"  
7 PRG "PROGRAMBLE CHAR"  
9 PRG "C64 CHAR GENR"
```

On the left side of a line is a number. This number represents the amount of "blocks" the program uses. A block is 254 bytes. The first program, then, is five blocks or 1270 (5×254) bytes long. If a program is 255 bytes long, it takes two blocks, so the block concept is approximately how much room a program needs.

The first file, again, is about 1270 bytes long. You can also tell that this file is a program by the designation **PRG**. If it is a data file, the letters **SEQ** follow the block numbers. Data files are found most often when you have written your own program that requires data. The last piece of information about the file is its name—**TINYMON1 FOR C64**. This is the title you use to **LOAD** the program. If you forget the name or spelling of a program, check the information in quotation marks in the directory.

Now that you have **LOADed** a few programs you are ready to **SAVE** the programs you create. Chapter 5 tells how.

CHAPTER 5

Output to Cassette, Diskette or Printer

If you have a program in the computer's memory and want to store it for later use, you can **SAVE** the program. The program stays intact when the computer is turned off and the memory is erased. This chapter describes how to **SAVE** on cassette, on diskette, and on a printer. The cassette and disk store information in electronic codes that make sense primarily to the computer. The printer provides "hard copy," the program transferred to paper. If you store on a cassette, insert a good quality tape in the cassette recorder. If you store on a diskette, use a 5¼ inch, soft-sectored diskette that has been prepared (formatted) for a program to be stored on it. The section on **SAVEing** with diskettes tells how to format a diskette.

Suppose you have typed in a program and want to show off your new programming talents. But it will be a week or so before you see your skeptical friends. Other than leaving the computer on all that time, you can **SAVE** the program. Type in a program if you don't have one already in memory. The one in Chapter 3 will work nicely. Now follow the instructions in this chapter for **SAVEing**.

SAVE TO CASSETTE

First, make sure the recorder is properly connected to the computer, as explained in Chapter 4. Next, name the program using up to 16 characters (including spaces). You do not need to name the program on a cassette, but **LOADing** is simpler when several

programs are on one cassette. Name this program DEMO for the experiment.

Rewind the tape to the beginning; set the memory counter to zero; and follow these steps to SAVE the program called DEMO:

STEP 1. TYPE:

SAVE "DEMO"

and press **RETURN**. The screen displays:

PRESS RECORD & PLAY ON TAPE

STEP 2. Press **RECORD** and **PLAY** and the recorder starts automatically. Be sure both **RECORD** and **PLAY** are pressed as the computer can't sense if only one of them is pressed. Nothing is recorded (or **SAVED**) on a tape player if both **RECORD** and **PLAY** are not used. The **SAVE** light on the recorder (just below the memory counter) comes on. The screen displays:

SAVING DEMO
READY

When the program is **SAVED**, the light goes off and the tape stops automatically.

STEP 3. Note the program name and where it stops on the memory counter. Write the beginning and ending numbers on the cassette label, for example, if DEMO starts at location 5 and ends at 12. When **LOADING** the program later, you will know where the program is located and when you put another program on the tape you'll know where previously saved programs end.

To practice, **SAVE DEMO** again, but this time call it **SAMPLE**. Where on the tape should you **SAVE** it? You could rewind the tape and **LOAD DEMO** again, find where it ends and **SAVE SAMPLe** there. Or, since you know the location on the memory counter, you can rewind the tape, reset the counter to zero and fastforward the tape to a safe point beyond 12. This is faster than **LOADING** each time.

Because strange things can happen when programs are **SAVED**, Commodore 64 has a **VERIFY** command that lets you check the program just **SAVED** on tape with the program still in memory. Why is this such a nice feature? Suppose you have a very important financial program you have just **SAVED** because you'll need it next

week when your income tax return is audited. You're in a hurry, but decide to **VERIFY** the program anyway. To your horror, you discover that the program was not **SAVEd** correctly. You get a new tape, **SAVE** and **VERIFY** again and everything is fine. Without the **VERIFY** feature, you would not have known that the program was not **SAVEd** until you tried to **RUN** it after the original had been erased. The crucial element here is that **VERIFY** checks the **SAVEd** version while the *original* is still in the computer's memory.

To use the **VERIFY** command, rewind the tape to the beginning of the program, in this case it's the first one on the tape. Type:

VERIFY DEMO

and press **RETURN**. (When the program you want to **VERIFY** is first on the tape, you need not include the program name.) After typing **VERIFY**, you can press **RETURN** and the first program is checked. The screen replies:

PRESS PLAY ON TAPE

Press **PLAY** and the screen displays:

SEARCHING FOR DEMO
FOUND DEMO
VERIFYING
OK
READY

with a few seconds between each message. When the **VERIFYing** is completed, the tape recorder automatically stops.

As long as the **SAVEd** version matches the version in memory, the above message appears. But what if there's a problem? If the name is misspelled, the computer searches the entire tape and can't complete the **VERIFY** command because the program isn't there. If, however, a problem occurred during the **SAVE** procedure itself, the following message appears:

?VERIFYING
ERROR
READY

It may be difficult to determine the problem, but you may want to **SAVE** the program on another section of the tape or on the other side, or use another tape altogether.

With the tape that has DEMO and SAMPLE on it, rewind the tape to the beginning and **VERIFY** SAMPLE. After typing **VERIFY** SAMPLE and pressing **RETURN**, the screen displays:

```
SEARCHING FOR SAMPLE
FOUND DEMO
FOUND SAMPLE
VERIFYING
OK
READY
```

You can **VERIFY** a program regardless of its location on the tape, provided it is still in memory. **VERIFY** all your important programs before erasing them from memory.

SAVE TO DISKETTE

The advantage of a disk system in terms of time was mentioned earlier in relation to **LOADING** programs. When you **SAVE** and **VERIFY** programs on diskettes, you realize the same advantage of time.

FORMATTING DISKETTES

To **SAVE** a program on diskette, you need to format a diskette. This involves writing an electronic map on the diskette. Later, when the computer wants to store information on the diskette, the electronic map tells it where to place the information.

Before you can use a new diskette to store programs or data, you must also use the format procedure. Use blank diskettes because the formatting process erases anything on the diskette. If you put an unformatted diskette into the drive and ask for a directory, the message

```
SEARCHING FOR $
```

appears on the screen. The red light on the disk drive comes on (indicating it is trying to read information on the diskette) for a few seconds and then goes off. After a few more seconds, the light goes on and then off again until you do something about it. The computer looks for instructions on the diskette and doesn't find any. Assuming it has made a mistake, the computer goes back and tries to read the blank diskette again and continues trying to do what it

was instructed to do. The only thing you can do to stop the cycle is pull the diskette out of the drive. Try to do this when the red light is *off* because when the red light is on, the computer is trying either to get information from or to the diskette. After you remove the diskette from the drive, turn the computer off, wait a few seconds, and then turn the computer back on.

To format a diskette, follow these steps:

STEP 1. Insert a diskette in the drive

STEP 2. Close the drive door securely

STEP 3. Type

OPEN 15,8,15

and press **RETURN**. The screen displays

READY

OPEN 15 tells the disk drive to prepare to communicate with you on a particular part of the diskette. This is called **OPENing** a "channel" and is necessary before you can write or read from a device other than the computer. In this case, the number of the other device is 8 which stands for the disk drive. The last number in the **OPEN** statement (15) is called the secondary address and this establishes limits for the device that was specified.

STEP 4. Type

PRINT #15,"N:DISKNAME,DI"

and press **RETURN**. The diskette is being formatted and the identifiers **DISKNAME** (any name with a combination of characters and/or spaces such as: **MY PROGRAM**) and **DI** (any two characters like **DW**). **DI** is short for Disk Identifier. These identifiers differentiate between disks when files are being read to or from them. The red light on the drive light comes on for about 85 seconds during the formatting process.

STEP 5. When the red light on the drive goes off, type:

CLOSE 15

and press **RETURN**. This tells the computer you have completed your communication with the disk drive and the communication channel can be closed. That's formatting, or preparing a diskette for use. It's a good idea to prepare several formatted diskettes so they are ready to use when you have a program in the memory of the computer and want to store it.

SAVEing INSTRUCTIONS

To practice **SAVE**ing to diskette, use the same program from Chapter 3 as you used in the cassette explanation. Continue using the names **DEMO** and **SAMPLE**. When you have a program typed or **LOAD**ed in, follow these steps to **SAVE** it:

STEP 1. Insert a formatted diskette into the drive and close the door securely (if you are unsure how to insert the diskette, see Chapter 4).

STEP 2. Type:

SAVE "DEMO",8

and press **RETURN**. The red light on the drive comes on while the program is being **SAVE**d. When the light goes off the program has been **SAVE**d and the screen displays:

READY

If you get a message like:

PRESS RECORD & PLAY ON TAPE

one set of quotation marks was omitted or the device number 8 is missing. Remember to put the name of the program inside the quotation marks and the comma outside the quotation marks. The 8 represents the device number for the disk drive and must always be used when **LOAD**ing or **SAVE**ing with diskettes. If the device number is left out, the Commodore 64 assumes you are using a cassette and the keyboard locks up. If this happens, press the **RUN/STOP** key to get control again and retype the instruction.

Another frequent message is:

?DEVICE NOT PRESENT

This appears if a number other than 8 (for the disk drive) is entered.

Although a file name is not required when using cassettes, it is impossible to **SAVE** a program on disk without one. If you were to type **SAVE " ",8** to try not to give a name, the following appears on the screen:

**?MISSING FILE NAME
ERROR
READY**

Just as you can **VERIFY** cassette **SAVEs**, you can also **VERIFY** disk programs in much the same way. Again, the purpose is to compare a program just **SAVEd** with the original still in memory. To **VERIFY DEMO** type:

VERIFY "DEMO",8

and press **RETURN**. The computer checks to make sure the **SAVEd** version is exactly like the original. The following message is displayed:

```
SEARCHING FOR DEMO
FOUND DEMO
VERIFYING
OK
READY
```

if everything is as it should be. If, for some reason, the **SAVEd** version is different, the message is:

```
SEARCHING FOR DEMO
FOUND DEMO
?VERIFY
ERROR
READY
```

and you can be sure there is a problem. If the program isn't a crucial one, you may want to try to **SAVE** it again on the same diskette. If you get the message again, or the program is important, the best suggestion is to use another diskette and go through the **SAVE** and **VERIFY** routines again.

Now, **SAVE** the program in memory again and call it **SAMPLE**. Type

SAVE "SAMPLE",8

and press **RETURN**. You actually have two copies of the same program now, one called **DEMO** and one called **SAMPLE**. Next, **VERIFY SAMPLE**. The screen will tell you when **SAMPLE** is found and how the **VERIFY** procedure goes (an error message appears if there is a problem, "OK" is reported if all is well).

OUTPUT TO A PRINTER

The last method of **SAVEing** information is using a printer to get a "hard copy," or printed program. The Commodore 64 printer

connects to the back of the computer in the serial port (see Chapter 4 for an installation diagram). Make certain the printer is properly connected and turned on before the Commodore 64 is turned on.

You can print one or more lines of information in a program, or you can have the lines of a program printed out. When using a printer, the word to get a permanent copy of your program is **LIST**. The example in this chapter explains how to get a line by line **LISTING** of the program as well as how to print material from within a program.

Suppose you have a program like the one below. When the program is typed in, the word **RUN** is typed, and the **RETURN** key is pressed, the program executes. In the case of the program below, the phrase "THIS IS A TEST" should appear on the screen and then the **READY** message. Type in the following program to use in the examples for making use of the printer as a device to **LIST** your programs.

```
20 PRINT "THIS IS A TEST"  
30 END
```

Press **RETURN**, type **RUN** and press **RETURN** again. The screen looks similar to this:

```
20 PRINT "THIS IS A TEST"  
30 END  
READY  
RUN  
THIS IS A TEST  
READY
```

The computer sends the output to the screen. That is, when you type **RUN** and press **RETURN**, the execution of the program is displayed on the screen. You want the information inside the quotation marks sent to the printer. To do that, a few pieces of information must be given to the computer by adding three lines to the program. To begin, type

```
5 OPEN 9,4
```

and press **RETURN**. This tells the computer that you want to **OPEN** a file numbered 9 (this often called the logical file and can be any number from 0 up to 255) and that you want to access device number 4 (the printer can be either 4 or 5, your 64 manual lists all the devices and their corresponding numbers). This line tells the computer to **OPEN** file number 9 and to send the information for

logical file 9 to a printer (device 4), Next, it is necessary to use the **CMD** instruction to tell the Commodore 64 that the printer is to be the primary output device instead of the screen. Type

```
10 CMD9
```

and press **RETURN**. **CMD** initializes the command channel and 9 tells the computer that file number 9 is the output channel. The file number always follows the **CMD** instruction. If by mistake **CMD** is followed by the device number, the following message appears:

```
?FILE NOT OPEN  
ERROR
```

and the program won't execute. You won't know this, however, until you type **RUN**. If the **FILE NOT OPEN** message appears, you know that you have entered a file number other than the number specified in the **OPEN** statement. This can be corrected by either retyping the line or moving the cursor up to the **CMD** line and typing over the wrong number with correct one.

After a file is **OPENed** and you are finished with it, the file must also be **CLOSEd**. If a file is not **CLOSEd**, an error message may result later on. To close a file in our example program, type

```
25 CLOSE9
```

and press **RETURN**. As it stands now, the program seems to be a mess of line numbers in the wrong order. Fortunately, the Commodore 64 straightens out the jumble and puts the program in proper numerical sequence. To see the program correctly, type

```
LIST
```

and press **RETURN**. The screen looks like the following:

```
5 OPEN9,4  
10 CMD9  
20 PRINT "THIS IS A TEST"  
25 CLOSE9  
30 END
```

and in a couple of seconds adds

```
READY
```

By using the **LIST** command, you have a line by line copy of your program displayed on the screen. Notice that the program line numbers are in the correct order now.

To get the information inside the quotation marks in the program to go to the printer, type **RUN** and press **RETURN**. The printer types:

THIS IS A TEST

and a few seconds later the computer responds with

READY

Everything that normally appears on the screen is displayed by the printer when **CMD** is used. Now type

LIST

and press **RETURN**. Notice that the **LISTing** of the program lines that earlier appeared on the screen now are displayed on the printer. After **RUNning** a program to the printer, the **LIST** command sends the program lines to the printer. This is the way the printer is used as a **SAVEing** device. When the computer is turned off, the program and all the hard work that went into it is gone. With **LISTing** to the printer, however, the creative work of developing a program is not lost when the computer is turned off because you have a duplicate of the program on paper.

What happens if you take out line 10 (where the **CMD** instruction is given)? Try it by typing

10

and press **RETURN**. Now use **LIST** and press **RETURN**. The program **LISTing** is only on the screen. Type **RUN** and notice that **THIS IS A TEST** is displayed only on the screen. It is the use of **CMD** then, that provides the opportunity to use the printer as a **SAVEing** device.

In any program that you have, the following additions are necessary to send a **LISTing** or **RUN** a program to the printer:

```
5 OPEN 1,4
10 CMD1
50 CLOSE 1
```

where 5, 10 and 50 are program line numbers (the line numbers in your program might be different). The file number (1 here) can be any number from 0 to 255, and the 4 represents the device number of the printer. **CMD1** opens the command channel to file number 1 (this file number must be the same as the file number in the **OPEN** statement), and **CLOSE** is used to close the file number 1.

There is a way to get information to the printer without using the **CMD** statement. This method lets you send some information to the screen and some to the printer. For example, if you are working on a home budget program, you may want questions to appear on the screen, such as "How much did you spend on groceries this week?" and a printout of the summary chart of expenditures. Use of **PRINT#** allows this distinction. Type in the following short program to illustrate how this works:

```
10 OPEN 1,4
20 PRINT "THIS MATERIAL IS TO BE DISPLAYED ON
    THE SCREEN"
30 PRINT#1, "THIS MATERIAL IS TO BE PRINTED BY
    THE PRINTER"
35 CLOSE 1
40 END
```

Line 10 **OPENs** file number 1 and sends it to device number 4 (the printer). Line 20 is a standard **PRINT** statement. Line 30 tells the computer that the information in quotation marks is file number 1 and that it is to be sent to the printer. Line 35 **CLOSEs** the file and Line 40 **ENDs** the program. **PRINT#** always transmits information to the device number specified in the **OPEN** statement.

To see this work, type **RUN** and press **RETURN**. The message in Line 20 is displayed on the screen and Line 30 is printed out for you.

If line 20 is displayed and then the following message appears:

```
?SYNTAX ERROR IN 30
```

the most likely culprit is the comma after **PRINT#**1. Always include a comma after the file number in this statement. If this message appears, you can retype the line or use cursor keys and the **INST/DEL** key to edit the line.

When **SAVEing** programs on a cassette or diskette, the program can be **LOADed** (using directions from Chapter 4) at a later time. With a printer, the line by line **LIST** of the program is protected but, to be used, it must be typed in again on the Commodore 64.

CHAPTER 6

Programming in BASIC

This is the first of two chapters that deal specifically with the popular computer language called BASIC. At this point, you are familiar with the procedures for correcting and editing programs as you type them into the computer.

PURPOSE OF CHAPTERS 6 AND 7

There are at least two hundred books currently in print that teach you how to write programs in BASIC. At least 10 books deal specifically with the programming the Commodore 64 in BASIC. We will not try to teach you how to write BASIC programs on your computer in two chapters. Instead, the goal of Chapters 6 and 7 is to introduce the concept of BASIC and its use on the Commodore 64. In addition, the chapters provide you with enough knowledge to take BASIC programs written for the Commodore 64 and use them yourself. There are many books and several magazines of BASIC programs for the Commodore 64. After reading the next two chapters, you should understand how BASIC programs work and you should be able to take a Commodore 64 BASIC program from a book or magazine, type it in on your computer and use it. You will know enough BASIC to debug a program that isn't working as it should and even make changes or enhancements that interest you.

HOW TO USE THIS SECTION

It would be possible for you to plow through these two chapters without a computer and learn some BASIC. Our advice, however,

is to read the BASIC chapters with a computer in front of you. Learn what the new terms mean, and use your computer to do the examples scattered through the reading. Don't try to memorize definitions and instructions. Practice enough so that working in BASIC is comfortable. It is not necessary to be able to say off the top of your head what **GOSUB 140** means; you only need to know where to look for that information. BASIC becomes a familiar second language through experience, not memorization.

INTRODUCTION TO BASIC

There are several important differences between the language spoken by a computer and regular English. In English the meaning is usually clear even if a word or two is mispronounced or misspelled. BASIC and other computer languages are not so forgiving. You must say precisely what you intend to, and say it exactly as the computer expects.

Another difference is in the way punctuation is handled. Commas, semicolons, and periods clarify the meaning of written English, but the rules vary. You would be able to understand a letter from Uncle Harry even if he managed to write a whole page without a comma or period. In BASIC, these punctuation marks are often as important as the letters and numbers. Leaving out one comma can prevent an entire program from running properly.

Finally, English is a very rich language. There are usually several words that have similar meanings; a 1956 Chevrolet could appropriately be called a car, auto, automobile, or vehicle. BASIC is not so well endowed. Often there is only one way of saying something.

The words BASIC understands are called "key words." To make the computer do something, these key words are used to make up statements. Statements correspond to sentences in English. The term statement actually has two uses in computer programming. It can mean the program line, and it is often used in much the same way "key word" is used here. Three examples of statements (*i.e.*, program lines) are given below:

```
10 LET A = 2
20 LET B = 4
30 PRINT A + B
```

Each statement is preceded by a line number. When a computer is told to **RUN** a program it begins with the statement that has the lowest line number and follows the instructions given in that statement. It then goes through the program until all the statements have been executed. When it reaches line 30 in our example, it does the arithmetic requested ($2 + 4$) and prints the result. Type in this program on your 64. If some of the details of typing in programs are fuzzy you may want to review Chapter 3.

The three statements you typed in make up a simple program. A program is a set of instructions or statements that tell the computer how to solve a particular problem. When the three-statement program is typed in, type **RUN**, then press the **RETURN** key. If the computer responds by printing 6, you are off and running.

At this point we have several new terms. BASIC is a popular computer language. It enables the computer to understand a number of special or "key words" such as **LET** and **RUN**. BASIC also has a set of rules about how punctuation marks are used. Using these rules and the key words BASIC understands, you can write programs that the computer can execute or carry out. Programs are made up of lines or statements, each of which begins with a number (the line number). Statements contain key words, used according to the rules of BASIC.

You tell a computer you want it to execute a program (follow the instructions in the program) by giving it the key word **RUN**. This is the normal way a computer is used. There is, however, another way. If you simply type in the following line:

PRINT 2 + 2

the computer prints the result of adding 2 and 2 immediately after you press the **RETURN** key. If you do not put a line number in front of a statement, the computer assumes you want the instructions carried out immediately. This method of using a computer is sometimes called the calculator mode or the immediate mode. It is handy since it lets you use the computer as you would a calculator. The key word **PRINT**, if it is used to tell the computer to do something immediately, is called a *command*. If **PRINT** is part of a program it is called a *statement* in most textbooks. Most of the key words in BASIC can function as either commands or statements. That is, they will work in either the calculator mode or the program mode.

In the next section you will learn more about some of the most frequently used keywords in the Commodore 64 version of BASIC.

PROGRAM CONTROL COMMANDS

Several of the key words available in BASIC are used to tell the computer how to manage its work. They are listed below with a description of the job they do:

RETURN. This command has its own key on the keyboard. RETURN is used to tell the computer you have finished typing in a line of instructions. When you press the RETURN key the computer enters the line in its memory if it begins with a line number. If there is no line number the computer executes the instructions immediately.

RUN. This command has already been used several times. When you use the RUN command, it tells the computer you want it to execute the instructions contained in the program currently in the computer's memory. Normally, RUN tells the computer to begin with the instructions in the line with the lowest number. There is a variation of RUN, however, that specifies where to begin. If you tell the computer RUN 70, the computer starts executing instructions in line 70 and above. (Any instruction with line number less than 70 would not be executed.)

NEW. There are times when you want to stop what you are doing and start over. One way to accomplish that is to pull the plug on the computer. Another way is to use the command NEW. NEW erases any program currently in the computer's memory. You can then start fresh with nothing left of the old program. Before you use NEW be sure you really do want to erase everything in memory. Once it is done, it's done.

HOME/CLEAR. The instructions HOME and CLEAR have their own key at the top right of the keyboard. (See Chapter 3). In essence, when you use HOME or CLEAR you are using these key words in the calculator mode. The instruction is carried out immediately. You do not even have to press the RETURN key. Can you clear the screen and/or move the cursor to its home position by using these key words in a program? Yes, you can, but not by using HOME or CLEAR. Type the following program into your computer:


```
10 PRINT "THIS IS THE TOP LINE."  
20 PRINT "THIS IS THE SECOND LINE."  
30 PRINT "SHIFT(CLR/HOME)"  
40 GOTO 10
```

Line 30 should be read as an instruction to type 30 **PRINT** " and then hold down the shift key and press the **CLR/HOME** key. Finally, type the last quotation mark and press **RETURN**. Hold down the shift key and press the **CLR/HOME** key to produce an odd graphic character on the screen that looks like the heart on the S key, except the heart is blue and the border around it is light blue. This is a reverse graphic character. Most of the keys on the Commodore 64 that perform a particular function, such as clearing the screen or changing the color of the border or background, will produce a particular reverse graphic character when that key is pressed within quotation marks. When the computer comes to one of these characters as it executes a program, it acts as if the key that produced the character had been pressed. **RUN** the program now and watch what happens.

The two lines of print produced by lines 10 and 20 seem to flash on and off rapidly. After 10 and 20 produce their message, the computer reads line 30 (**PRINT "SHIFT(CLR/HOME)"**). If you hold down the **SHIFT** key and press the **CLR/HOME** key, the computer executes a **CLEAR** command. This erases the screen and moves the cursor to the home position. Line 30 causes that to happen. The screen is erased and the cursor goes to the top of the screen. Line 40 tells the computer to go back to line 10 and execute the instructions again. The two lines of print are produced again; line 30 erases them again, and the process begins again. What would happen if line 30 were deleted? Delete 30 by typing 30 and pressing **RETURN**. Now **RUN** the program again. This time, since the lines are not erased and the cursor does not go to the home position, the screen fills up with the print message from lines 10 and 20. As each new line is added to the bottom of the screen, a line at the top is pushed off the screen.

BREAK. On most computers, this command has a key of its own that is labelled **BREAK**. On the Commodore 64 you can tell the computer to stop executing a program (**BREAK** the program) by pressing the **RUN/STOP** key. If your computer is still running the example program that illustrated how **CLR/HOME** works in a program, press the **RUN/STOP** key now. The program will stop

executing and the computer will display a message that says **BREAK IN 20**. (It may give the number 10, 20, 30 or 40.) That means the computer stopped while it was working on line 20.

END. This key word tells the computer it has reached the end of the program. It causes the computer to stop executing instructions.

SAVE and **LOAD**. These key words are used to load and save material on cassettes or diskettes. They are dealt with in detail in Chapters 4 and 5.

LIST. This command tells the computer that you want to see the lines of the current program displayed on the screen. You get a listing of the program. **LIST** causes the entire program to appear on the screen, if there are no more than 23 lines. If your program is more than 23 lines long, the computer can't get it all on the screen at once. **LIST** will show you the final 23 lines.

There are several other formats for **LIST**, however:

LIST#- begins the listing at the line number specified. **LIST 30-** begins the listing at line 30.

LIST# without the dash (-) after the number, gives the line specified. **LIST 30** causes only line 30 to be displayed.

LIST - #, with the minus (-) preceding the line number. This lists all the lines from the beginning of the program up to a specified line number.

LIST #-# prints a section of the program. **LIST 30-95** causes all the lines from 30 to 95 to be listed.

EDIT This is not a key word in Commodore 64 BASIC but is a useful feature. You have already learned how to edit a program line using the Right/Left Arrow Key and the **INST/DEL** key. Up to this point if you typed a line incorrectly and pressed **RETURN** the only thing you could do is retype the entire line.

The Commodore 64 actually has another method of editing lines after the **RETURN** key has been pressed. Find the UP/Down Key on the bottom, right of the keyboard. Use that key to move the cursor anywhere you want on the screen. Pressing that key alone moves the cursor down. Holding down the **SHIFT** key and pressing the Up/Down key causes the cursor to move up. If you move the cursor up to a line that has already been entered in a program (by pressing **RETURN**) you can still use the Right/Left Arrow key and the **INST/DEL** key to correct an error. Then press the **RETURN** key. Use the Down Arrow key to move the cursor to a line with nothing printed on it and type **LIST**. You will see that the line you corrected on the screen is now listed in the corrected form.

You can even change the line number of a set of instructions. Suppose there is a line 20:

```
20 PRINT "THIS IS A DEMO LINE"
```

If you move the cursor over the 2 in 20 and press the 4 key the line number changes to 40. Now if you press **RETURN**, move the cursor to a clear line, and type **LIST** you will have a line 20 and a line 40:

```
20 PRINT "THIS IS A DEMO LINE"  
40 PRINT "THIS IS A DEMO LINE"
```

If the program already had a line 40, the one you created replaces the original line 40. Creating a new line from an old one, without destroying the old one, is handy when you are typing in many similar lines. You can create a new line by changing the number of an old line and then edit the new one where necessary.

CONT. This is short for **CONTInue**. If you interrupt execution of a program by pressing the **RUN/STOP** key, you can tell the computer to pick up where it left off by typing **CONT** and pressing the **RETURN** key. You can also override a **STOP** statement in a program by typing **CONT** and pressing **RETURN**.

STOP. This key word is a statement that works a lot like the command **BREAK**. A program line like 45 **STOP** causes program execution to stop at line 45. If you want the computer to continue executing the program after it gets to the line containing the **STOP** instruction, just use the **CONT** key word. The computer will pick up where it left off and continue execution.

CLERICAL INSTRUCTIONS

PRINT, COMMA, SEMICOLON

Now we'll pick up the pace a bit. In the following sections additional key words and programming rules are presented and explained. Generally, the explanation is associated with a short demonstration program. If this is your first venture into programming we advise you to read the explanations carefully and **RUN** each of the sample programs.

PRINT. This statement causes the computer to print out or display whatever follows **PRINT**. If material following **PRINT** is

enclosed in quotation marks, the computer prints it exactly as typed. The material inside the quotation marks is called a string or a literal string. The instruction **PRINT**, "THIS IS A TEST" would cause the computer to print the string THIS IS A TEST on the screen.

If there are no quotation marks around the material to be printed, the computer behaves a little differently. Consider the program below:

```
10 LET THIS = - 30
20 PRINT THIS
```

Type in and **RUN** the program. Line 10 tells the computer that you want to use a variable named THIS and that you want to make THIS equal - 30. Line 20 does not print the name of the variable (THIS). Instead, it prints the value of THIS which is - 30. Whether you use quotation marks or not will make a big difference in what the computer does.

COMMAS and SEMICOLONS. Add the following lines to the two lines you typed in to see how **PRINT** works:

```
30 PRINT "THIS IS";THIS
40 PRINT "THIS IS ";THIS
50 PRINT "THIS IS ",THIS
60 PRINT "THIS IS",THIS
```

Note the only difference between line 30 and line 40 is the space between the S in IS and the quotation mark. Now **RUN** the program. Your display should look like this:

```
- 30
THIS IS - 30
THIS IS - 30
THIS IS   - 30
THIS IS   - 30
```

The - 30 on the first line of the display came from line 20. Lines 30 and 40 produce almost identical displays. The literal string THIS IS was printed first by both lines. Then the value of the variable THIS was printed. In line 40 a space inside the quotation marks makes the line neater and easier to read because the number 30 is not directly up against the S in IS. The semicolon after the literal string tells the computer to print whatever comes next immediately after the last item printed. If you use a semicolon to separate

two items that are to be printed, you must add any spaces needed yourself (e.g., inside the quotation marks of the literal string).

Lines 50 and 60 produced exactly the same results even though one has a space inside the quotation marks and one does not. Why? The key lies in using the comma rather than the semicolon as a separator. A comma instructs the computer to begin printing whatever follows in the next print zone rather than right up against the material already printed. Thus, a comma generally separates material while a semicolon puts it together. Type **PRINT 1,2,3,4** and press **RETURN** to see how the computer spreads out material that is separated by commas.

Both commas and semicolons show no respect for the end of a line. If you tell the computer to print too much material to fit a line, the computer will type all it can on one line and put the rest on the next line. The program below illustrates the point. Use the key word **NEW** and then type it in:

```
5 FOR X = 1 TO 20
10 PRINT "AND";
20 PRINT "YET";
30 PRINT "ANOTHER";
40 NEXT X
```

RUN the program and you will see that it fills up 6 lines on the display and spills over a bit to the seventh line. The three literal strings **AND**, **YET**, and **ANOTHER** are printed end to end until the program loop has been executed 20 times. Now add the following lines:

```
50 PRINT
60 PRINT
70 FOR B = 1 TO 7
80 PRINT "EVEN",
90 PRINT "MORE",
100 NEXT B
```

RUN the expanded program. You get the same letters at the top of the screen. Then lines 50 and 60 give us a blank line (50 gets us off the seventh line, 60 skips one line). Then the second **FOR NEXT** loop prints a pattern of **EVEN MORE** on the screen seven times. The comma on lines 80 and 90 causes the computer to space over and print material at the beginning of each of the four print zones.

You may want to experiment with the **PRINT** instructions and with commas and semicolons a bit to get familiar with how they work.

A final note on **PRINT**. If you tell the computer to **PRINT** something and don't put a comma or semicolon after it (*e.g.*, **PRINT "HELLO"**) the computer will automatically go to the beginning of the next line when it encounters the next **PRINT** instruction.

Another final note on **PRINT**. Since **PRINT** is used so often, Commodore 64 BASIC lets you substitute a question mark for **PRINT**. Typing **50 PRINT** and **50 ?** will produce the same thing. Most of the key words used in Commodore 64 BASIC can be abbreviated with the first letter of the word plus the next letter typed with the SHIFT key depressed. **CONT**, for example, can be typed by typing a C, then holding down the shift key and pressing the O key. Appendix A lists the shortened versions of all the keywords in Commodore 64 BASIC that have abbreviations.

CHAPTER 7

Getting Information Into the Computer

There are three major ways Commodore 64 BASIC allows you to assign “values” to “variables.” You can do this with the **LET**, **GET**, and **INPUT** statements.

LET AND TYPES OF VARIABLES. You have already used **LET** several times in the sample programs. The general format for **LET** is:

LET variable name = value

The **LET** key word tells the computer what value to associate with a particular variable name. For example, **LET B = 22** tells the computer the variable named B will equal (have the value of) 22.

There are two basic types of variables, numeric and string. In the example above, B is a numeric variable. **LET B\$ = “HELLO”** tells the computer that the string variable B\$ is to equal HELLO. You must enclose string variables in quotation marks (*e.g.*, **LET B\$ = “FINE”**, not **LET B\$ = NOT FINE**). The \$ after B designates the variable as a string variable. If the variable name does not end with a \$, the computer assumes it is dealing with a numeric rather than a string variable. The Commodore 64 will not accept **LET A\$ = 22** or **LET A = “HELLO”** because there is a mismatch between the type of variable specified (numeric or string) and the type of value assigned. If you type **LET A\$ = 22** and press the **RETURN** key, the computer responds with:

?TYPE MISMATCH
ERROR

This is the way the 64 tells you it does not understand the instructions given. See Appendix B for an explanation of the Commodore 64’s error messages.

If you assign a value to B (or B\$) in a program (e.g., 10 **LET B = 22**) and then assign another value to B later (e.g., 30 **LET B = 66**) the value of B is the last one assigned, 66 in this example.

Up to this point the value assigned with the **LET** statement has been a simple numeric value (22) or a literal string value ("HELLO"). **LET** is much more versatile. **LET** can use another variable to assign the value to a new variable (**LET B = A** or **LET B\$ = A\$**) and it can use an expression (**LET B = 2 + 5**). These **LET** instructions are also acceptable:

```
LET B = B + 2  
LET B = B + A
```

You must define any variable name on the right side of the equal sign. Program line 30, **LET B = B + A**, is fine if an earlier line defined the value of A (e.g., 10 **LET A = 95**).

There are two additional limitations on the way you name variables. The BASIC used by the Commodore 64 has a number of key words that it treats as instructions. You have used key words such as **PRINT**, **LET**, and so forth already. You cannot use a key word to name a variable. **ABS**, for example, is not an acceptable name since it is a key word. The same is true for **ATN**, **CLR**, **COS**, **SQR**, **SPC**, and many more. The key words used by the Commodore 64's version of BASIC are listed in Appendix A.

As mentioned above, you can only use two characters in a numeric name and two characters plus a \$ in a string variable name. However, you can actually use an instruction such as **LET TEST = 24** or **LET INVENTORY\$ = "CAR POLISH"**. The Commodore 64 will accept these instructions, but it will use only the first two characters in the name. The variables in the examples above thus have the names **TE** and **IN\$**. That means **TEST** and **TEXT** are the same variable to the Commodore 64 since they both begin with **TE**. Avoid problems by selecting names that meet the Commodore 64's expectations.

INPUT. When the computer comes to an **INPUT** statement, it stops and waits for the operator to give it some data. If **INPUT** is followed by a numeric variable name (e.g., **INPUT B** or **INPUT BALANCE**) the computer expects you to type in a numeric value and press **ENTER**. If **INPUT** is followed by a string variable name (e.g., **INPUT B\$** or **INPUT NAME\$**), the computer expects you type in a string (don't enclose it in quotation marks) and press **ENTER**. If you give a numeric value when the computer expects a string value, or vice versa, the computer will reject your input. Here is a short example program:


```
10 PRINT "TYPE IN A NUMBER"  
20 INPUT A  
30 PRINT A  
40 PRINT "TYPE IN A STRING"  
50 INPUT A$  
60 PRINT A$
```

RUN this program. The screen will display "TYPE IN A NUMBER" on one line and a ? will appear on the next line. The computer is waiting for you to type in a numeric variable. If you type in a number and press **ENTER**, the computer will set A equal to the number you typed in. Then it will print that value (line 30). Line 40 tells you to **TYPE IN A STRING**; line 50 tells the computer to look for a string. Type in **TESTING** and press **RETURN**. The computer will print **TESTING** and then print **READY**.

What happens if your input is not what the computer was expecting? **RUN** the program again and type in **TESTING** when the computer expects a numeric value. When you press **RETURN** the computer prints the message **?REDO FROM START**. That means it did not expect you type in a string and wants you to try again. The problem is a mismatch between the variable name (A) and the type of data input (**TESTING**). Type the number 34 and the program will continue. The computer accepts the number when you press **RETURN** and prints it on the screen. It goes on to lines 40 and 50. Now it is looking for a string variable. Type in 34 and press **ENTER**. Surprised by the results? The computer accepted 34 as a string with no problems and printed 34. That means A equals 34 and so does A\$. There is a difference in the two variables, however. Add the following two lines to the program:

```
70 PRINT A + 12  
80 PRINT A$ + 12
```

RUN the program and give both A and A\$ the value of 34. When the computer gets to line 70 it will compute $34 + 12$ and print 46. Then it will print the following:

```
?TYPE MISMATCH  
ERROR IN 80
```

You cannot add a string variable to a numeric variable. Even though A\$ equals 34 you can not use it in math operations. To the computer the 34 value of A\$ is simply a two-character string—not a number. The variable A can be manipulated mathematically while the variable A\$ cannot.

Up to this point we have used simple **INPUT** statements in which the value of one variable was requested. An **INPUT** statement can be used to assign values to more than one variable. **INPUT A,B,C** would cause the computer to print a ? on the screen. If you type in 44 and press **RETURN** the computer will type a double question mark (??) to indicate it expected more. Type in 66, press **RETURN**, and another ?? will appear since there are three variables to be assigned values. Type in 77, press **RETURN**, and the computer will be happy: A will equal 44, B will equal 66, and C will equal 77. You can also input all three values at once. If you had typed 44, 66, 77 and pressed **RETURN** the computer would have accepted all three values at once. You can even mix numeric and string variables (**INPUT A\$, B, C\$**) but you must be careful to enter the values in the correct order (*e.g.*, **STRING, NUMBER, STRING**) and separate them with a comma.

There is one final point about **INPUT** statements. The **RUN/STOP** key has already been mentioned as a means of telling the computer to stop executing a program. **RUN** your program one more time. When the computer asks you to input a value for A, press the **RUN/STOP** key. Normally, you would be able stop execution of a program by pressing the **RUN/STOP** key. That will not work when the computer is waiting for input. You can stop execution of the program during execution on an **INPUT** statement by holding down the **RUN/STOP** key and pressing the **RESTORE** key. This causes the computer to execute a "reset" instruction. It clears the screen and prints **READY**. The program you are executing is not erased from memory, but the value of all variables in the program is set to 0. **LIST** and **RUN** will now allow you to view the program lines on the screen or execute the program again.

GET and **GOTO**. One of the problems of using **INPUT** is the requirement that you press **RETURN** after typing the necessary data. That is usually no great burden, but there are times when pressing **RETURN** creates problems. Many games, for example, begin with instructions. You tell the computer whether you want instructions, or not, by pressing one of two keys. You tell the computer you have finished reading the instructions by pressing another key. Novices often forget to press the **RETURN** key in such situations and end up wondering why the program doesn't work. You can avoid this frustration if you use **GET** instead of

INPUT. Type **NEW** to get rid of any program in the memory of the computer and then enter the program below. It will illustrate the use of **GET**:

```
10 PRINT "NEED INSTRUCTIONS?"
20 PRINT "PRESS Y FOR YES"
25 PRINT "PRESS N FOR NO"
30 GET A$
35 IF A$ = "Y" THEN GOTO 60
40 IF A$ = "N" THEN GOTO 100
50 GOTO 30
60 PRINT "INSTRUCTIONS PROVIDED HERE"
70 PRINT "PRESS G TO BEGIN GAME"
80 GET B$
85 IF B$ = " " GOTO 80
100 PRINT "THE GAME BEGINS HERE"
```

The **GET** statement in line 30 lets you define the value of the string **A\$** simply by pressing a key on the keyboard. Line 10 in the example asks if you need instructions. Lines 20 and 25 tell you to press the **Y** key for instructions, and the **N** key, if you don't want them (*e.g.*, you've played the game before and are ready to begin). Lines 35 and 40 check to see if the string **A\$** equals **Y** or **N**. If it equals **Y**, the computer obeys the **GOTO** instruction that tells it to go to line 60 where the instructions would be printed. If **A\$** equals **N**, the **GOTO** instruction in line 40 tells the computer to jump to line 100 and begin executing instructions from that point on. As soon as you press **Y** or **N**, the computer follows one of the **GOTO** instructions in lines 35 and 40. The computer will execute lines 35 and 40 before you press **Y** or **N**, however. When it does, **A\$** will equal nothing since you have not pressed a key. Line 50 sends the computer back to line 30. The program will thus "loop" through lines 20 through 40 until a key is pressed.

If you press the **Y** key, the computer prints the material in lines 60 and 70. Normally, that might be a screen of instructions. You are also told to press **G** when you finish reading the instructions and are ready to play the game. Line 80 gets the value of **B\$**. Then we come to line 85:

```
85 IF B$ = " " THEN GOTO 80
```

As long as you do not press a key on the keyboard the string variable **B\$** will equal nothing. Nothing is indicated by " " which is

also called a "null string" since there is nothing between the quotation marks. Thus, unless you press a key, the computer will execute the instructions in line 70 and 80 over and over. When you do press a key, the **GOTO 80** instruction in line 85 will not be executed since **A\$** does not equal " ". The computer goes on to the next line of instructions.

GET does have its limitations. It can only have a one character value (**GET A\$** can equal A, or B or 1 and so on, but it cannot equal **AVERAGE**). In addition, the computer quickly scans the keyboard to see what key is pressed when it encounters **GET**. That means you must write the program in such a way that the computer will check the keyboard over and over until a key is pressed.

MATH EXPRESSIONS AND ORDER OF CALCULATION

BASIC uses the normal symbols to indicate addition (+) and subtraction (-). Multiplication is indicated by an asterisk (*) and division by a slash (/). The expression:

LET A = 5 * 4 + 1

means A equals five times four plus one. The expression:

LET B = 20 / 4 - 1

means B equals twenty divided by four minus one. The variable A thus equals 21 and B equals 4. There is one more math symbol that is frequently seen in programs. Look at the two examples below:

LET R = 5 * 5 * 5 **LET R = 5 ↑ 3**

Both these expressions equal 125. The first uses a series of multiplication symbols to obtain the answer. The second expression is read as five raised to the third power or five cubed. The symbol for powers is an up arrow which is just above the **RETURN** key on the right side of the keyboard.

When **BASIC** is used to do a series of math operations, it follows a standard sequence in doing them. All the power computations are done first, then the multiplication and division is finished. Finally, the addition and subtraction is computed. A handy mnemonic to remember the standard sequence is Please My Dear Aunt Sally for Power, Multiply, Divide, Add, and Subtract. When there are several of the same types of math to be done, the computer will do the work on the left first and then work across to the right.

Sometimes the standard order of computing is not the order needed to solve the problem correctly. It can be changed by the use of parentheses. The computer will do all the computations inside parentheses before doing the work outside the parentheses. Consider the expressions below.

PRINT $6 * 4 - 3$ **PRINT** $6 * (4 - 3)$

The first expression produces 21 because the multiplication is done first, then three is subtracted from 24. The second expression produces 6 because the subtraction is done first ($4 - 3$), then the multiplication ($6 * 1$). If an expression has several sets of parentheses the computer will do everything inside the innermost set of parentheses first, then work in the next and so on, and, finally, do the work outside any parentheses. When using parentheses, be sure there is one right parenthesis for every left parenthesis, otherwise the computer will report an error.

TYPES OF NUMBERS

The BASIC used in the Commodore 64 can deal with several types of numbers. Here are examples of the types of numbers it accepts:

- 5 positive integer or "whole" number
- 5 negative integer or "whole" number
- 345.55 positive decimal number
- 345.55 negative decimal number
- 4.33E + 5 positive number in scientific notation
- 4.33E - 5 negative number in scientific notation

CHAPTER 8

More BASIC

MAKING DECISIONS AND COMPARISONS

BASIC has several ways of comparing one variable to another and making decisions. The most important of these are **IF THEN**, **GOSUB**, and **FOR NEXT**.

IF THEN

Remember the explanation of the key word **GET** in the preceding chapter? The example program used several **IF THEN** statements to make decisions about what to do next. The key words **IF** and **THEN** are used together to enable the computer to make many different types of decisions.

IF THEN is more a family of instructions than just one statement. Here is a typical example of an **IF THEN** statement:

```
50 IF X < N THEN PRINT X$,X
```

Line 50 above looks at the value of X. **IF** X is less than (<) N, **THEN** the computer will **PRINT** the string X\$ and the value of the numeric variable X. The **IF** in line 50 is used to specify a "condition." If that condition is true the action specified in the **THEN** part is performed. There are several types of **IF THEN** statements.

FOR NEXT TO

You have already used several simple **FOR NEXT** loops in Chapter 6. The program listed below will be used to illustrate some of the finer points of **FOR NEXT** loops:

```

10 PRINT "DOLLARS TO INVEST"
15 PRINT "EACH YEAR"
20 INPUT D
30 PRINT "YEARS TO INVEST"
40 INPUT Y
50 PRINT "RATE OF INTEREST"
60 INPUT R
70 LET R = R/100
80 PRINT "YEAR  INVEST  TOTAL"

```

Note: type in three spaces between YEAR and INVEST and three spaces between INVEST and TOTAL.

```

90 LET B = 1 + R
100 PRINT 1;TAB(7);D;TAB(15);D
110 FOR L = 1 TO Y
120 LET B = B*(1 + R)
130 LET S = (D*(B - 1))/R
140 IF L < Y THEN PRINT L + 1;TAB(7);(L + 1)*D;
    TAB(15);INT(S)
200 NEXT L

```

Type in the program exactly as it is listed above. Most of the key words used in the program will be familiar to you.

This program lets you indicate how much money you will invest per year, how many years you will invest that amount, and the interest rate you will receive. The computer takes that information and computes the accumulated investment per year (under INVEST) and the total value (including accumulated interest) of your investment (under TOTAL). The amount under TOTAL is rounded to whole dollars by the INT key word (you'll learn how INT works later in this chapter).

RUN this program using 600 for the Dollars Invested per Year, 10 for the Years to Invest, and 9 for the Rate of Interest. Here is what you should get as a printout:

Table 8.1 Run of Investment Program

```

DOLLARS TO INVEST YEARLY
?600
YEARS TO INVEST
10
RATE OF INTEREST
9

```


YEAR	INVEST	TOTAL
1	600	600
2	1200	1253
3	1800	1966
4	2400	2743
5	3000	3590
6	3600	4514
7	4200	5520
8	4800	6617
9	5400	7812
10	6000	9115

Thus, if you invest \$600 a year for ten years you will end up with \$9,115 of accumulated interest and principle. Lines 110 to 200 define a LOOP. This is the heart of the program since it computes the amount of money that will be earned and prints the results. It is called a loop because it is used several times, once for each year you plan to invest. The **FOR NEXT** statement controls how many times the computer moves through the loop. The **NEXT** in line 200 defines the lower boundary of the loop. When the computer comes to a **NEXT** it returns to the line where **FOR** occurs and goes through the loop again. The way **FOR** operates is a little complicated. In this case, L is the "control" variable in the loop (there is no significance to the label L, it could be any other acceptable variable name). The expression on the other side of the equal sign tells the computer where to start and how many times to run through the loop. Line 110 sets L equal to 1 (the initial value for the first loop). Each time the computer runs through the loop, it increases the value of L by one. When L is equal to Y (the final value), it goes through the loop one more time and moves on to the line immediately after the **NEXT** statement. Y is the number of years you plan to invest your money, so there will be one line of results for each year you invest. Since in this program there is no line after line 200 where **NEXT** appears, the computer stops when it finishes the **FOR NEXT** loop.

GOSUB RETURN

FOR NEXT loops let you use the instructions inside the loop over and over. The number of times they are used is determined by the control values provided. You will find **FOR NEXT** loops used in many programs.

Another handy procedure that is common in BASIC programs is the *subroutine*. A subroutine is a set of instructions that is used at several points in the program. The example program below will make the concept of a subroutine clearer:

```
10 GOSUB 100
20 PRINT "GUESS MY NUMBER"
25 PRINT
30 INPUT G
40 GOSUB 200
50 GOTO 20
100 REM RANDOM NUMBER GENERATOR
120 LET N = INT(RND(0)*100)
130 RETURN
200 REM GUESS CHECKING ROUTINE
210 IF G > N THEN PRINT "TOO HIGH TRY AGAIN"
220 IF G < N THEN PRINT "TO LOW TRY AGAIN"
230 IF G = N THEN GOTO 280
240 GOTO 300
280 PRINT "GREAT, YOU GOT IT!"
285 PRINT "I WILL THINK OF A NEW NUMBER"
290 GOSUB 100
300 RETURN
```

This program prints GUESS MY NUMBER at the top of the screen. Line 25 is a **PRINT** instruction with nothing to print that causes the computer to skip a line each time line 25 is executed. You then type in a number between 0 and 100 and press **ENTER**. The computer compares your guess with the number it has generated and tells you if your guess is high, low, or correct. If it is high or low the computer tells you to try again and asks you to make another guess (don't forget to press **ENTER**). When you guess the number, the computer tells you "GREAT, YOU GOT IT." Then it says "I WILL THINK OF A NEW NUMBER" and asks you to input another guess. To stop playing this game, you should press the **RUN/STOP** key and **RESTORE**.

This program begins with the instruction **GOSUB 100** in line 10. The key word **GOSUB** tells the computer to stop executing instructions sequentially. Instead it is to go to a subroutine that begins at the line number following **GOSUB**. In this case the subroutine begins on Line 100. Actually line 100 contains a **REM** statement that tells us what the subroutine does. If you begin a line with the key word **REM**, the computer ignores the line, but

you can type a message after **REM** that provides information to the programmer. The subroutine that begins at line 100 generates a random number. The subroutine includes lines 100 to 130. We know line 130 is the end of the subroutine because it contains the key word **RETURN** to tell the computer that the subroutine has been executed. The computer then returns to the instruction just after the **GOSUB** key word, in this case, line 20. The computer has branched to a subroutine that did a particular task and then returned to continue normal execution of the program. Line 20 prints **GUESS MY NUMBER**; then line 30 looks for a number that you type in. When you press **ENTER**, the computer takes the number you typed in and assigns it to the variable named **G** (for guess).

Now we come to line 40 where there is another **GOSUB**, this time to line 200. The subroutine includes lines 200 through 300. This subroutine compares the number generated by the computer (variable **N**) with your current guess (variable **G**) and gives you either a hint or a congratulatory message. If you are incorrect, the **GOTO** in line 240 sends the computer to line 300 which has a **RETURN** statement. The computer returns to the instruction just beyond the **GOSUB** that brought it to the subroutine. That is line 50 where there is a **GOTO** statement which sends the computer back to line 20. The computer tells you to **GUESS MY NUMBER** and looks for another input. When you type in another number, the computer comes to the **GOSUB 200** instruction in line 40 once more, and things start all over again.

What happens if you guess correctly? Line 230 in the subroutine identifies the guess as correct and the instruction **GOTO 280** is executed. The computer offers praise in 280 and tells you it will think of a new number to guess in line 285. Then line 290 does something interesting. We are already in a subroutine, but there is another **GOSUB** in 290. The **GOSUB** there sends the computer to the random number generator subroutine beginning in line 100. When the computer has a new number for variable **N**, it returns to the instruction just past the **GOSUB** instruction in 290. That is line 300 which is another **RETURN**. This time the computer goes back to line 50 because line 40 sent it to this subroutine.

There are several advantages to the use of subroutines. You can write all the instructions to do a particular task in one place in the program and then use the **GOSUB** instruction any time you need that task done. For many people, writing programs is easier if you break the job down into a series of subroutines. (One theory of

programming, however, criticizes the use of subroutines. The theory feels a program should be written so that it executes sequentially, from top to bottom, with no jumping around from one place to another.) In addition, when you must do a particular task at several points in the program, it is very easy to put a **GOSUB** at the appropriate place and avoid writing the same set of instructions at several points in the program.

CHAPTER 9

Graphics, Sound, and More BASIC

Some of the fancier features of the Commodore 64 computer are outstanding color graphics and music synthesis. These features are difficult to use without buying one or more programs that assist you with graphics and sound. Some of the color capabilities are easy to use.

CONTROLLING BACKGROUND, BORDER AND CHARACTER COLORS

A light blue border surrounds the standard Commodore 64 screen. The background field where characters are typed is dark blue, and the characters themselves are light blue. Some people find this pattern pleasing while others prefer a different pattern of background, border, and character colors. To change the color of the border and the background (sometimes called the playfield or display window) select from the 16 options for the border, screen, and character color. Special locations in the Commodore 64's memory control the colors used. The location for border color is 53280. Type in the following instruction and note the result:

```
PRINT PEEK (53280)
```

Your Commodore 64 should print 254. The instruction above tells the computer to look at memory location 53280, determine what number is stored there, and then print that number on the screen. When the number 254 is in memory location 53280 the computer uses a light blue border. Now type the following:

```
10 INPUT A  
20 POKE 53280,A  
30 GOTO 10
```

The program above allows you to type in a number which becomes the value of variable A. Line 20 uses the instruction **POKE** that tells the computer to place the value after the comma in the memory location specified just after **POKE**. If you **RUN** this program and type in 0 the computer treats this as an instruction to change the border to black. Now try that.

Below is a list of the border and background color options and their numbers:

0	black	8	orange
1	white	9	brown
2	red	10	light red
3	cyan	11	gray 1
4	purple	12	gray 2
5	green	13	light green
6	blue	14	light blue
7	yellow	15	gray 3

Do you wonder why the computer returned the number 254 when the border was light blue? Light blue is obtained by **POKE**ing the number 14 into memory location 53280. That location always has at least the number 240 in it. To determine what number the current color represents, you must subtract 250 from the number you obtain. Thus $254 - 240$ is 14, the number for light blue.

Another memory location, 53281, controls the color of the playfield. If you change line 20 above to:

```
20 POKE 53281,A
```

and run the program again, the numbers provided determine the color of the background. Use the table above to select background color.

The Commodore 64's ability to easily change background and screen colors can create some startling visual effects. Type in the following program:

```
10 PRINT "{CLR}"
20 PRINT "***** ALERT ALERT *****"
30 POKE 53281,2
40 FOR X = 1 TO 150:NEXT X
50 POKE 53281,6
60 FOR X = 1 TO 150:NEXT X
70 GOTO 30
```

Note: Line 10 tells you to type a special sort of print instruction. The "CLR" in the brackets is obtained by holding down the shift key and pressing the **CLR/HOME** key. Do not type the brackets or the letters **CLR**. The brackets tell you a control key on the keyboard is to be pressed. The **CLR** tells you which one. **CLR** is at the top of the keyboard on the right. To get the **CLR** function you must hold down the shift key and press the **CLR/HOME** key. For your effort, you get a small square inside the quotation marks that contains a heart shape rather than the {CLR}. The heart is an "inverse" video character that indicates it stands for one of the control keys on the keyboard. Line 10 causes the computer to clear the screen and move the cursor to the home position. Typing a control key after **PRINT** and inside quotation marks tells the computer to execute an instruction rather than actually printing something on the screen. Most of the control keys (*e.g.*, cursor up, down, left, right; backspace, home) can be given as instructions in a program using this format.

If you **RUN** this program it will print ***** ALERT ALERT ***** on the top of the screen and then alternately change from blue to red. The **FOR NEXT** loops in lines 40 and 60 are there to slow the computer down. It counts to 150 each time it gets to lines 40 and 60. If they were not there the screen would change colors so quickly the result would be displeasing. Changing the terminal value in the **FOR NEXT** loop to something other than 150 changes the rate of flashing

Push the **RUN/STOP** key to terminate execution of this program and type **POKE 53281,6**, then press **RETURN**, if your screen is red and you prefer the standard blue. Another way to return the computer to standard, or default screen, colors is to hold down the **RUN/STOP** key and press the **RESTORE** key.

CONTROLLING CHARACTER COLOR

Examine another aspect of color on the Commodore 64 with this short program:

```
10 PRINT "{BLK}THIS IS IN BLACK{WHT}"
20 PRINT "THIS IS IN WHITE"
30 PRINT "{YEL}THIS IS IN YELLOW"
40 PRINT "{RVS ON WHT} THIS IS IN REVERSE
   WHITE{RVS OFF}"
```

There are some odd instructions in this program. Line 10 should be interpreted as instructing you to type in the number 10 followed by **PRINT**. Then type the quotation mark. The {BLK} should be interpreted as an instruction to hold down the **CONTROL** key and pres the key with BLK embossed on it. That is the 1 key. Now type THIS IS IN BLACK. Finally, {WHT} means “hold down the **CONTROL** key and press the key with WHT embossed on the front (the 2 key).”

Line 30 asks you to type {YEL} which is obtained by holding down **CONTROL** and pressing the 8 key. Finally line 40 has {RVS ON WHT}. Hold down **CONTROL** and press the 9 key, then hold down **CONTROL** and press the 2 key. At the end of line 40 you set {**RVS OFF**} by holding down **CONTROL** and pressing the 0 key.

As you type in the **CONTROL** instructions note that certain graphic symbols appear on the screen. **CONTROL-BLK** produces a small square; **CONTROL-WHT** produces a square with an inverse E in it, and **CONTROL-REVS ON** produces an inverse R symbol. You will become familiar with these symbols and what they designate as you use the graphics instructions. It is important to remember that these symbols indicate the computer is to follow a particular instruction when the program is executed. **RUN** the program now and observe the results. There are four lines of text—one with black letters, one with white letters, and one with yellow letters. The final line of print is in inverse video—that is the letters are surrounded by a darker background. Using the color and **RVS ON** keys inside quotations after a **PRINT** instruction is one way of controlling how material is printed on the screen. The {BLK} instruction tells the computer to print characters in black until a new instruction directs otherwise. Characters can be in any of the colors printed on the front of keys 1 through 8 and you can get more variety by printing some characters in reverse (*e.g.* inverse) video via the **RVS ON** key. In addition, seven other colors are available. If you hold down the key with the Commodore logo on it (bottom left of keyboard) and press a number from 1 to 8, a different set of colors is available. Here are the colors produced when the Commodore logo key is used instead of the Control key:

1 Orange	2 Brown	3 Light Red	4 Gray 1
5 Gray 2	6 Light Green	7 Light Blue	8 Gray 3

Note that the use of control characters inside quotation marks can cause problems under some circumstances. For example, when you are editing or correcting a program the cursor keys let you

move from one point on a line to another and from one line to another. Those same cursor control keys, however, are treated as instructions after a quotation mark. If you type **PRINT XR + RT** when you really meant to type **PRINT XF + RT**, you could use the left cursor key to move the cursor back over R and replace it with an F. If you type **PRINT "THIS AS** when you meant to type **PRINT "THIS IS"** the cursor key, if pressed before the end quotation marks, will be treated as an instruction to be inserted into the program. The left-arrow shows up as a rectangle with a narrow line down the left side. Each time you press the left-arrow cursor key, the computer prints one of those symbols. When the program is executed, the symbol is interpreted as an instruction to move the cursor back one space. Therefore, you can program cursor movement by typing cursor control graphic symbols inside quotation marks, but you cannot always use the cursor control keys to move about inside a string to make corrections. If you must make corrections inside a string, the **DEL** key will work as expected and pressing **RETURN** moves you off the line where the quotation marks are so you can retype the line correctly. In the example **PRINT "THIS AS** can be corrected by adding the second quotation mark (**PRINT "THIS AS"**). Now the cursor is outside the quotation marks. You can use the cursor to move back inside the marks without adding unwanted symbols.

The points made above can be illustrated with this line:

```
10 PRINT "THIS IS{CRSR DOWN CRSR DOWN}A TEST"
```

After typing **THIS IS** press the **CURSOR DOWN** key twice; you should see two inverse-Q's on the screen. Then type **A TEST"**. **CLEAR** the screen and **RUN** this program. The result should be:

```
THIS IS
```

```
    A TEST
```

The computer treated the two inverse-Q's as instructions to move down two lines on the screen. Now **LIST** the program. Use your cursor keys to move the cursor up inside the quotation marks of line 10. At this point the cursor keys let you move freely anywhere on the line. Edit the string "THIS IS A TEST" in any way you wish.

SCREEN MEMORY AND CHARACTER CODES

Variations of the **PRINT** instruction have been used to tell the Commodore 64 to display material on its video screen. Another

somewhat more complicated way to get material displayed on the screen is to use the **POKE** keyword to put 25 lines of 40 characters on the screen at one time. Each of those 1000 character locations on the screen (25 lines by 40 characters) has its own location in memory. The number stored in that location determines what the Commodore 64 displays on the screen. The memory location, for example, controls what is displayed in the top left hand corner of the screen is memory location 1024. Clear the screen and type in the following instruction:

POKE 1024,65

When you press the **RETURN** key the computer should display a graphic symbol in the shape of a spade in the top left corner of the screen. Memory location 1024 controls that screen location and number 65 is the code for a spade graphics symbol. Table 9.1 shows the screen memory locations for the entire Commodore 64 screen. Display a symbol at any location on the screen by poking the proper screen memory location with the character code you want to display.

Table 9.2 shows the codes for each of the Commodore 64 characters. Note that two characters are associated with each code. This computer has the patterns stored in ROM memory for two different sets of characters, one that produces capital letters and graphics symbols (Set 1), another that foregoes graphics symbols for upper and lower case letters (Set 2). The computer automatically selects the capitals/graphics set when it is switched on. You can tell it to select the other set in two ways. Hold down the Commodore Logo key and press the **SHIFT** key; do it again and it will flip to the other set. Memory location 53272 actually controls character set selection. If 53272 contains the number 21 the computer will use upper-case/graphics; if it contains 31 it will use the upper/lowercase set. If you give the instruction **POKE 53272,21** the computer will use the upper case/graphics character set. **POKE 53272,23** tells the computer to use the upper/lower case character set.

How does the computer “know” that the code number 65 represents a graphic symbol in the shape of a spade? The computer has a special area of **ROM** memory that contains patterns for video characters which begins at memory location 53248. These locations in **ROM** contain codes for the video display characters. The pattern for each character is stored in eight bytes of memory. The characters on the video display are actually dot patterns that

create the desired letter or character. The dots for one character are in a pattern of 8 lines of 8 dots. The letter A, for example, is created by the following dot pattern:

```
0 0 0 0 0 0 0 0
0 0 1 1 1 1 0 0
0 1 0 0 0 0 1 0
0 1 0 0 0 0 1 0
0 1 1 1 1 1 1 0
0 1 0 0 0 0 1 0
0 1 0 0 0 0 1 0
0 0 0 0 0 0 0 0
```

The “1’s” in the diagram above tell the computer to put a dot on the screen. Every character the computer can display has a dot matrix pattern in this section of **ROM**. Each of the lines of ones and zeros listed above can be stored in one byte of memory. (All data stored in **RAM** or **ROM** is actually stored as patterns of ones and zeros). One byte of memory is made up of eight bits. A bit of computer memory can store one piece of data, and that data can be either 1 or 0. To be completely accurate, ones and zeros are really the presence of an electrical signal (On equals 1) or the absence of a signal (Off equals 0). Thus the dot patterns for the video characters are stored in memory as eight bytes of data with each byte indicating the pattern of dots to be displayed on one of the eight lines in the dot matrix. When you tell the computer to display a particular character on the screen it looks in this section of memory to determine just what that character should look like on the screen.

To this point two important sections of memory have been discussed—the area where patterns for each character are stored and the area that controls the character to be stored on the screen. There is one more area which should be mentioned. The Commodore 64 lets you control what goes in a particular screen location and also decide what color the character should be. The memory location that controls the color of anything displayed at a particular location is determined by adding 54272 to the screen location number shown in Table 9.1. For example, the top left corner of the screen is memory location 1024. Thus location $1024 + 54272$ or 55296 will control the color of the character in the top left corner of the screen. You can POKE a number from 0 to 15 in 55296 and control the color:

0 Black	1 White	2 Red
3 Cyan	4 Purple	5 Green
6 Blue	7 Yellow	8 Orange
9 Brown	10 Light Red	11 Gray 1
12 Gray 2	13 Light Green	14 Light Blue
15 Gray 3		

This short program illustrates how all this works:

```

10 INPUT A
20 PRINT "{CLR}"
30 POKE 1524,83
40 POKE 1524 + 54272,A
50 GOTO 10

```

This program displays the graphic character that resembles a heart in the middle of the screen. The number 83 in line 30 tells the computer which number is to be POKED into memory location 1524. Table 9.1 tells us that 1524 controls the twentieth character position in the thirteenth line of the screen. Table 9.2 tells us that code 83 will produce a heart when the computer is using the #1 character set.

Line 30 puts the character code on the screen memory location. Line 40 tells the computer to POKE the value of variable A into the memory location that controls the color of character location controlled by memory location 1524. When this program is run the computer asks for input for the value of A. Type in 0 and press the **RETURN** key. Then line 20 clears the screen. Line 30 **POKEs** the character code into screen memory, and line 40 **POKEs** the number of the color for the character. If you input a 0, that is the code for black so you should see a black heart in the middle of the screen. Line 50 sends the program back to line 10 where you can input another value for A. Use any number between 0 and 15 to see how the color changes. Note that since the screen color is blue, giving A the value of 6 (blue) will make the character disappear since it is now a blue character on a blue background.

After you experiment with changing the colors of the heart, you may want to change lines 10 and 30 to the following:

```

10 INPUT A,B
30 POKE 7932,B

```

Now run the program. The computer will ask for two numbers when it executes line 10. Type in a number between 0 and 7 (remember 6 makes things "disappear"), type a comma, and then

type in another number. Use Table 9.2 to select the second number. This version of the program lets you select the color of the character (variable A) and the character itself (variable B). If you type in 0,7 you will get a black capital G. Now type in 0,83. Did that produce a black heart? Now type in 0,211. Did that also produce a heart? This one should be a reverse video heart. You can create reverse video images of all the characters by adding 128 to the character code. Since a heart has code 83 you add 128 to it and get 211.

ADVANCED COLOR GRAPHICS ON THE COMMODORE 64

This computer is capable of several types of advanced color graphics that make it outstanding compared to other computers in its price range. Each character on the screen is made up of an 8×8 matrix of dots. With the built-in graphics characters you can create some interesting graphics figures on the screen. Holding down the **SHIFT** key causes the Commodore 64 to print the graphic character on the right front of the key. Holding down the Commodore logo key causes the Commodore 64 to print the graphic character on the left front of the key.

Control of each of these dots individually, rather than as a set of 64 (8×8) dots, is usually referred to as high resolution graphics. You can create a much finer grained display if you can control each dot, or pixel, individually. A pixel, short for picture element, is the smallest element that the computer can independently control. The Commodore 64, with great effort, can control 200 lines of pixels with each line having 320 picture elements.

Standard Commodore 64 BASIC does not contain built-in key words that make high resolution graphics easy to use. Most of the work must be done by **POKE**ing screen locations. Unless you are highly motivated, do not tackle high resolution graphics without some additional software to make the job easier.

Another feature of the Commodore 64, sprite graphics, can be created and manipulated as one element rather than as a composition of many different picture elements. For example, you could create an airplane image made up of perhaps, hundreds of pixels. To move the plane on the screen, you have to instruct each of the pixels to move. Sprite graphics lets you create the image of a plane, define it as Sprite 1 or 2 or whatever, and then give instructions to the sprite rather than to individual picture elements that make up

the sprite. In other computers a sprite may be called a MOB, or movable object block. The Commodore 64 lets you create many different sprites (e.g., a set of Chess players, different types of aliens in a space game). Once created, the sprites can be moved about on the screen, expanded or reduced in size, even given priorities. If you have a cloud sprite and an airplane sprite, one can be given priority over the other. If the airplane has priority it will be in front of the cloud if they occupy the same area on the screen. The Commodore 64 also has provisions for collision detection so you know when two sprites are in the same screen location. That is very handy for many types of video games. Many programs are in print that will help you use the Commodore 64 color graphics. Some of those programs are described below:

FACEMAKER (Spinnaker Software, 215 First Street, Cambridge MA 02142). This program is for children 4 to 12. It is, in many ways, an electronic version of Mr. Potato Head, since it lets children create all sorts of funny faces with the aid of the program.

Sprite Designer (Academy Software, P.O. Box 9403, San Rafael, California 94912, cassette—\$17, diskette—\$22). This program provides help in designing and creating sprites.

Sprite-64 (CrossTech Graphics, 2133 North Fremont Street, Chicago, Illinois 60614 Phone 313-871-3555, tape or diskette—\$50). **Sprite-64** adds a set of keywords to BASIC that lets you use sprite graphics without using all those **PEEKs** and **POKEs**.

SpriteMaster 64 (Access Software, 925 East 900 South, Salt Lake City, Utah 84105 Phone 801-532-1134). This \$36 program helps you create sprites. It also creates a sprite subroutine that can be appended to other programs so you can use your sprites in other BASIC programs.

SpryteByter (FoxSoft, P.O. Box 507, Deer Park, Texas 77536 Phone 713-473-6723. Cassette—\$30). This program adds 60 new key words to the Commodore 64's sprite graphics vocabulary and allows you to use the joystick to create sprites.

SCREEN GRAPHICS 64 (Abacus Software, P.O. Box 7211, Grand Rapids, Michigan 49510 Cassette—\$25). This program adds 24 new key words to the Commodore 64's BASIC. It makes it much easier to develop, print and save on diskette or cassette your color graphic creations.

64 PANORAMA (Midwest Micro Associates, 311 W. 72nd Street, Kansas City, MO 64114, cassette—\$20). This is a program that turns the computer into a drawing board controlled by your joystick and some of the keys on the keyboard. It is much like a

souped up Etch-A-Sketch. Normally the joystick can be used to draw narrow or wide lines on the screen by moving the joystick handle and holding down the fire button. There is also an “erase” mode to undo anything you don’t want on the screen and a “connect” option that lets you specify two points on the screen and have the program connect them with a straight line. There is also a separate program that lets you draw circles. A set of 19 digitized pictures demonstrates just how good the graphics on this computer are. The pictures, which are high resolution black and white, can be printed on a standard Commodore printer and are quite well done.

Midwest Micro also sells a BANNER/HEADLINER program that lets you create and print large banners on your printer (cassette—\$20) and a program called 64 GRAFIX SAMPLER (cassette—\$20) which shows some of the 64’s color graphics features via demonstration screens.

SOUND ON THE COMMODORE 64

Control of five different memory locations creates sound on the Commodore 64. Each of these memory locations handles a separate aspect of the sound created and provides Commodore 64 owners with some of the best music and sound effects available on a personal computer. The Commodore 64 can control three separate tone generators which means it is a “three voice” synthesizer. Here are the details of one of the three tone generators:

Memory Location

54295 – Volume control. The number in this memory location can be between 0 and 15 with 15 specifying the loudest volume and 0 indicating no sound at all.

54272

54273 – Tone. These two memory locations specify the tone to be played. The Commodore 64 User’s Guide and the Programmer’s Reference Guide (Commodore, \$20) contain tables that provide a note to number chart.

54277 – Attack/Decay Pattern. The uniqueness of a particular sound is not simply a characteristic of the tone. Sounds rise to a maximum volume and then trail off in different patterns. A sharp or sudden sound has a rapid attack rate—it goes from no sound to

its maximum volume rapidly. This memory location controls the Attack and Decay pattern of a sound.

54278—**Sustain/Release**. This memory location determines how long a note is held at its maximum level.

54276—**Waveform**. If we hooked up an oscilloscope to a source of sound we could “see” the pattern of the sound on the oscilloscope’s screen. The tone generators in the Commodore 64 can generate sounds with four different waveforms—triangle, sawtooth, square wave, and noise.

The five memory locations described above collectively define the envelope of the sound generated. That is, they determine the picture of the sound we would see on an oscilloscope and they all contribute to the uniqueness of a sound. Different musical instruments, for example, could play the same musical note and still have distinctive sounds because their attack/decay patterns or waveforms were different. To use these memory locations directly to create music or sound effects is difficult. A number of utility programs make sound generation on the Commodore 64 easier. Commodore has even announced a piano style keyboard that can be connected to the Commodore 64 that will let you play the computer in much the same way you would an electronic organ.

Table 9.1 Screen memory locations.

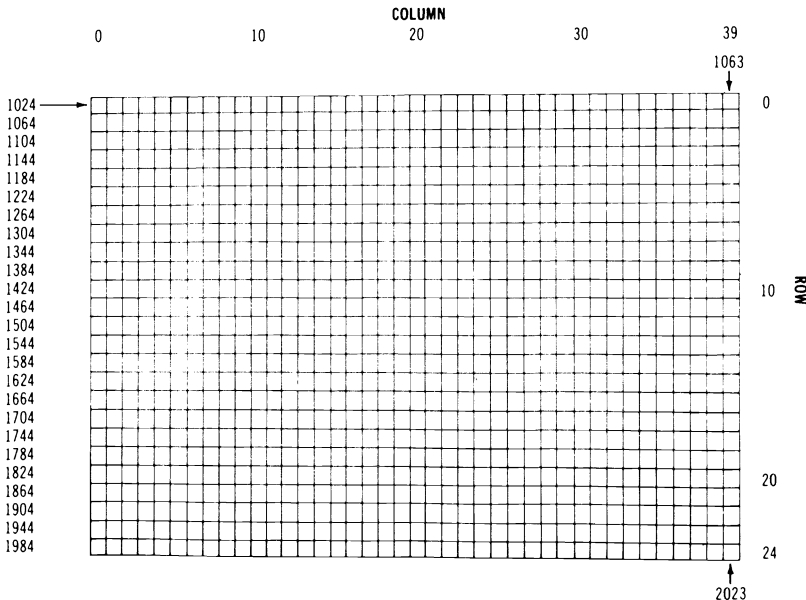


Table 9.2 Character codes.

SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE	SET 1	SET 2	POKE
@		0	SPACE		32			64	SPACE		96
A	a	1	!		33		A	65			97
B	b	2	"		34		B	66			98
C	c	3	#		35		C	67			99
D	d	4	\$		36		D	68			100
E	e	5	%		37		E	69			101
F	f	6	&		38		F	70			102
G	g	7	'		39		G	71			103
H	h	8	(40		H	72			104
I	i	9)		41		I	73			105
J	j	10	.		42		J	74			106
K	k	11	+		43		K	75			107
L	l	12	,		44		L	76			108
M	m	13	-		45		M	77			109
N	n	14	.		46		N	78			110
O	o	15	/		47		O	79			111
P	p	16	0		48		P	80			112
Q	q	17	1		49		Q	81			113
R	r	18	2		50		R	82			114
S	s	19	3		51		S	83			115
T	t	20	4		52		T	84			116
U	u	21	5		53		U	85			117
V	v	22	6		54		V	86			118
W	w	23	7		55		W	87			119
X	x	24	8		56		X	88			120
Y	y	25	9		57		Y	89			121
Z	z	26	:		58		Z	90			122
[27	;		59			91			123
£		28	<		60			92			124
]		29	=		61			93			125
↑		30	>		62			94			126
←		31	?		63			95			127

Several of the programs for creating music and sound are the *Music Machine* and *Music Composer* by Commodore; *Sound Shaper* by Quality Computer; and *Musicmaster* from *Compute!* Magazine, June, 1983.

CHAPTER 10

Software for Your Computer

As the Commodore 64's popularity grows, its software becomes increasingly available. Software for the Commodore 64 can be divided into four categories:

1. Recreational.
2. Business and home software.
3. Languages and Programming Aids.
4. Telecommunications.

PUBLIC DOMAIN SOFTWARE

One type of software that deserves special attention is Public Domain software, available in reasonable quantity for the Commodore 64. Several groups have collected a large number of programs written by Commodore 64 owners and user groups and donated them to the public. Users only pay a small handling and media fee, *i.e.*, \$10 for a diskette containing 10 to 20 programs. Quality varies from terrible to great. The major difference between public domain and commercial software is the general lack of documentation with public domain programs. Commercial software generally comes with some sort of instructions, while you may spend considerable time figuring out a public domain program. Many PET and 64 users groups have large libraries of public domain software, free to members of the group. Names and addresses of several sources of public domain software for the Commodore 64 are listed below:

Commodore dealers, especially in Canada, have been very active in collecting and distributing public domain software, particularly educational programs. Many dealers have 50 or more diskettes of

programs to loan customers who wish to copy them. Commodore has a newly packaged line of public domain software which can be purchased for a small charge.

Public Domain, 5025 South Rangeline Road, West Milton, Ohio, 45383, 513/698-5638. This company collects public domain software for the PET, VIC, and Commodore 64 computers on either cassette or diskette. Two sets of programs (over 25 in each set) are currently available on tape or diskette for \$10 a set.

Toronto PET Users Group (TORPET, P.O. Box 100, Station "S", Toronto, Ontario, Canada, M5M 4L6). Membership, \$20 a year, includes monthly issues of *THE TORPET* magazine and access to over 3,000 public domain programs from the club's library for \$10 a diskette.

Microcomputer Applications (1517 W. Church, Marshalltown, Iowa, 50518, 515/752-8845) give special emphasis on the school market and has one of the largest collections in the U.S. of educational programs and some utility software for \$10 a diskette. Some program instructions are in Dutch or German, but you can generally figure out how the programs work.

THE PET CONNECTION

A vast library of software that will run on the Commodore 64 has been developed in past years for older and more expensive computers. Commodore sells a PET emulator program for the Commodore 64 which allows it to run virtually all the BASIC programs written for the popular PET computers. With the \$30 emulator program, you can buy programs written for the PET, or type them into your computer from listings in magazines and books. Education Circuit (P.O. Box 333, Landing, New Jersey 07850) also has a PET emulator program on cassette that is \$24. It appears to work in much the same way as the emulator from Commodore.

RECREATIONAL SOFTWARE

Recreational software is available in three different formats: as cartridges that plug into the cartridge slot; programs on diskettes; and programs on cassettes. Cassette programs are generally cheaper but are less convenient to use. Disk programs are easier and faster to use but you must purchase a disk drive to use them. Diskettes are also somewhat fragile and require careful handling.

Cartridges, the most convenient form for recreational software, are also expensive. Cartridge games generally sell for between \$25 and \$55. To give you an idea of what is available for the Commodore 64 we have included descriptions of some of the popular game programs.

SKI MAN (Pacific Coast Software, 3220 S. Brea Canyon Road, Diamond Bar, California 91765 Phone 714 594-8210. \$20—diskette). The object of this game is to guide an intrepid skier down a slope strewn with obstacles. You determine the direction of movement with the joystick. The graphics in this game are good as is the sound. Each time you hit an obstacle the skier falls prostrate in the snow and an ambulance, siren wailing, comes onto the course, picks up the skier, and takes him away.

KNOCKOUT (Pacific Coast Software. \$20—diskette). This is a two player game with excellent color graphics. The screen becomes a boxing ring with stands of fans and two boxers in a ring. Each player has a joystick that controls the movement and punches (body and head blows) of one of the boxers. The game is played by rounds with points being awarded for punches that actually hit the other boxer. A very adept player can land enough punches to “knockout” an opponent and win the game.

Technically this is an excellent game. However, the focus on physical aggression and conflict may be regarded as unhealthy.

GRIDRUNNER (HES, 71 Park Lane, Brisbane, California 94005 Phone 415 468-4110. \$39.95—cartridge). This is a very fast moving game. The screen display is a grid of horizontal and vertical lines. You can move your player (a small triangle) anywhere in the bottom seven lines (with the joystick) and you must shoot down all sorts of meanies who move rapidly down the screen toward you while avoiding all sorts of deadly devices such as “plasma beams.”

The action in this game is extremely fast. Novices often have a lot of difficulty getting the hang of Gridrunner. The speed and the several levels of difficulty make it a challenging game, however, and one that is likely to be very popular with video game freaks.

SQUISH 'EM (Sirius Software, 10364 Rockingham Drive, Sacramento, California 95827 Phone 916 366-1195. \$34.95—diskette). The graphics for this game are outstanding. The object of the game is to “climb to the top of a 48 story building and collect a suitcase full of money. You must avoid being knocked off the building by a variety of Creepy Creatures or by falling objects.” You use your joystick to control the progress of a climber who scales the building story by story. You have four climbers and can select from several levels of difficulty.

This is an outstanding game, one of the best. Sirius has many video games for the Commodore 64 and several other computers.

FAST EDDIE (Sirius Software, \$34.95—diskette). This is another video game with excellent graphics. The objective of Fast Eddie is “to help Eddie capture as many floating Prizes as he can jump up and grab. You must keep Eddie hopping over the pesky little Sneakers while guiding him up, down, and around the screen.” The screen looks a bit like the Donkey Kong screen but the game plays differently. Fast Eddie is appropriately named because the action is very swift.

This is a colorful, well executed game. Another top notch product from Sirius. Playing this game makes you adjust your standards because you realize how good the color graphics features on the Commodore 64 can be.

Video games are available by the hundreds for the Commodore 64. There are card games like blackjack and poker, space games like Jupiter Lander (from Commodore), adventure games, puzzle and maze games, and much, much more. You can keep up with what is currently available by reading reviews in the magazines or visiting a well stocked video game or home computer store.

BUSINESS AND HOME SOFTWARE

The Commodore 64 is an in between computer that is good at most computer applications and great at some. It is a great home computer, a very good school computer (will be great when more software is available), and a good business computer. The limitations so far as business applications go, have to do with the keyboard (only four function keys), the display (25 × 40 instead of 25 × 80), the speed and capacity of the disk drive, and the availability of good business software (there is some, but not as much as for the CP/M oriented business computers such as the Sanyo 1000, Eagle II, and Morrow MicroDecision). Even when those limitations are considered, however, the Commodore 64 can do a decent day’s work in many small businesses and it can do many important financial and record keeping jobs around the home. It is very competitive with other computers in the \$1,200 to \$2,000 range although higher priced computers will generally have features that are desirable for business oriented computing.

WORD PROCESSING SOFTWARE

A word processing program turns your computer into an electronic text processing system to create all types of written material—from letters to books. On May 12, 1983, the last American-made manual typewriter, once the loyal companion of people from journalists and secretaries to students, rolled off the New York Smith Corona assembly line. Electric typewriters cut into the manual typewriter market first, followed by electronic typewriters and then word processors. Word processors create, edit, and revise documents in a fraction of the time that would be required if a typewriter were used. In the future personal computers, using word processing software and inexpensive printers, may well replace typewriters in many small businesses and in homes.

The Commodore 64, when used with good word processing software, is an adequate word processing computer. If you add one of the 24 × 80 video enhancement boards to it, the result can be a very inexpensive, but powerful, word processing computer for the home or business.

RAPIDWRITER (H.D. Manufacturing, 91 Long Hill Road, Leverett, Massachusetts 01504 Phone 413-549-3744, tape or diskette—\$40). This program is available for the VIC 20 and Commodore 64 computers. It is easy to learn to use, has features normally found only in more expensive programs, and can be used to print material on several different printers (some are limited to the 1525 printer). RapidWriter is written in BASIC and thus does not operate as fast as word processing programs written in machine language. All in all, however, it does most of the things you would expect in an inexpensive word processing program.

QUICK BROWN FOX (548 Broadway, Suite 4F, New York, New York 10012, Cartridge—\$65). Advertised extensively, this program is probably the best known of the VIC 20 and Commodore 64 word processing programs. It is written in machine language and compares favorably with word processing programs for computers such as the Atari 800 and the Apple II. Quick Brown Fox (QBF) is a cartridge based program which is available in versions for the VIC and Model 64 (and the IBM PC). It is a professionally produced program that has much to offer. The manual is comprehensive, well illustrated, and understandable. The manual is oriented toward the novice user who has no word processing experience.

There are a few complaints about QBF which should probably be

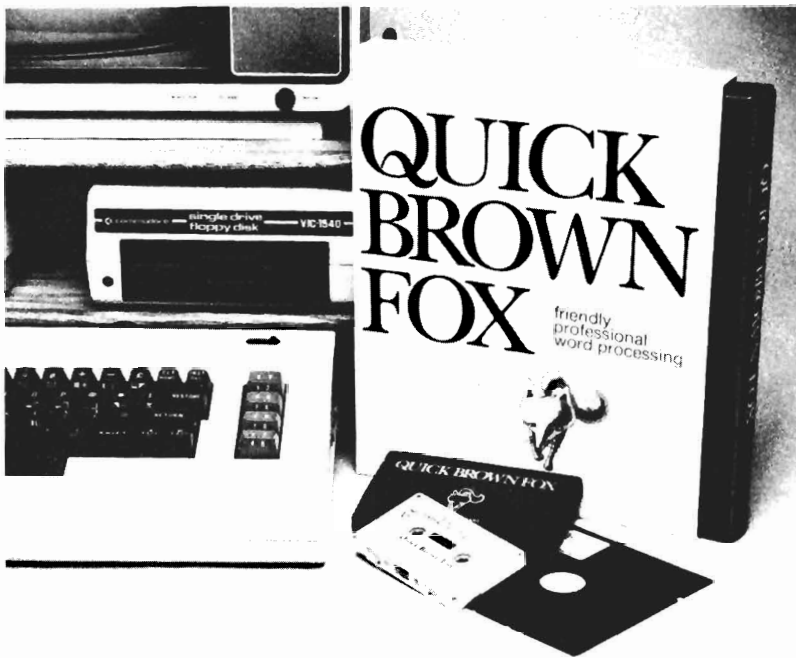


Figure 10.1 The Quick Brown Fox word processing program.

mentioned. A major problem of QBF is the way it operates. On sophisticated word processing systems you type in material, view it on the screen, and make any changes you want as you write. The process takes place with the screen full of material and with the program in a standard “create and edit” mode. QBF has a “create” mode that lets you type in material for the first time. In the create mode the screen is not full of material, however. You must press several keys and go to the “view” mode to see a full screen. To edit that material you press more keys and go to “edit” mode. More expensive word processors permit you to create, edit, and view material simultaneously.

QBF is sometimes awkward to use but it has the features of programs that cost over \$200. For example, you can tell it to center a heading, underline or bold face selected words, and delete a word.

EASY SCRIPT (Commodore, diskette—\$49.95). Commodore’s word processing program for the Commodore 64 comes with a good manual that provides all the details on the program although it tends to be more complicated and confusing than is necessary. It

works with a standard 40 character per line display and will store documents on cassette or diskette. It has most of the features a home word processor should and competes acceptably against many more expensive programs that run on similar computers such as the Apple II. One feature we particularly liked was the ability to delete a character, word, or sentence by simply pressing a key. On many word processing programs deleting a sentence with 100 characters means that you press the delete key 100 times. Easy Script also has search and replace features that let you correct a spelling error that appears in many places in the document with one set of instructions. It also takes advantage of the color capabilities of the Commodore 64. You can change the color of the border, the text, and the background if you wish.

The program also lets you specify whether you will be using a serial or parallel printer and there is even a section in the manual that shows you how to hook up many of the more popular printers to the Commodore 64—a very helpful feature. Easy Script has a method for sending special codes to the printer that lets you take advantage of special printer features.

We have only one major complaint against this program. It does not have a true “word wrap.” On a typewriter you must watch for the end of the line and press the carriage return key to start another line. A word processor should do that for you and “wrap” around to the next line when there is not enough room to place the last word typed on the current line. If you run out of room on a line as you are typing “splendiferous,” most word processors will automatically take the whole word down and make it the first word on the next line. Easy Script breaks the word up which means you could have “sple” at the end of one line and “ndiferous” at the beginning of the other. Easy Script gets high marks in many areas but the lack of word wrap lowers our overall evaluation to “good” but not great. Commodore has two other programs that work with Easy Script. Easy Spell 64 lets you check spelling in your documents and Easy Mail 64, a program that lets you keep track of mailing lists (e.g., church or club memberships, catalog mailing lists, Christmas card lists).

TOTL.TEXT (TOTL Software, 1555 Third Avenue, Walnut Creek, California 94596.) This program comes on diskette or cassette and is available in two versions—one for \$25 and a more powerful version for \$35. The manual is poorly written and likely to be confusing to novices. The program itself, however, is powerful. Most of the features beginners want in a word processing

program are found in this one. It is easy to use once you figure out what the manual is trying to say, and it works with many different types of printers. It is, however, written primarily in BASIC and is thus much slower than programs such as Quick Brown Fox. Totl.Text has a simpler and more straightforward method of editing material than QBF and will be preferred by some. It does not, however, have word wrap. The same company markets a Mailing List and Label program.

Before you select a word processing program, we suggest you think through the way you will be using the program (*e.g.*, to write letters, do term papers, create book chapters) then get demonstrations of several programs that look attractive. As you try the program out keep in mind the way you plan to use it and look for features that are desirable for that application.

ACCOUNTING AND FINANCIAL SOFTWARE

Several companies offer accounting software that runs on the Commodore 64. Many programs are designed for family financial applications; some will work satisfactorily in a very small business environment. Powerbyte Software (2 Chipley Run, West Berlin, New Jersey, 08091), for example, has a complete line of business and home software. Programs from this company include Accounts Receivable/Payable, Business Inventory, Order Tracker, My Profit Margin, Business Calendar, Billing Solver, Cash Flow Model, Liner Regression, Stock Ticker Tape, Home Budget, Medical Records, and Mother's Recipes. Some typical accounting software is reviewed below:

THE BUSINESSMAN (Southern Solutions, P.O. Box P, McKinney, Texas. Phone 214/542-0278). Southern Solutions markets a comprehensive line of accounting software for Commodore computers, including the Commodore 64. Software packages generally cost \$100 or less and are available for all the major business accounting areas as well as for many home financial applications. The Businessman is a comprehensive general ledger system that works with the Commodore 64, a disk drive, and a printer. This program is well written, easy to use, and relatively powerful considering the low cost. It also has several nice features not found on many accounting programs. The F1 function key, for example, is a "whoops" key. That is, if you type something in and then realize it

was correct and now you've changed it to an incorrect entry, pressing the whoops key will return things to the way they were before.

The manual for *The Businessman* is thorough, as comfortable to read as it can be considering the topic, and well organized. It is one of the best manuals for accounting software we have ever seen. The program has a cash journal that keeps track of checks written and deposits made, a general journal that performs the same jobs a double entry bookkeeping system would, and provisions for creating several types of reports. Consider *The Businessman* if you plan to use the Commodore 64 for accounting applications.

ACCOUNT PAC (Pacific Coast Software. Diskette—\$30). As you might expect this program is not as comprehensive as the one described above. *Account Pac* was written for people who want to manage their household finances or the finances of a very small business.

The manual for this program is an explanation of how the program works and a tutorial on how to set up and use a double-entry accounting system. You are shown how to set up a "chart of accounts" which contains a list of all sources of income and expenses as well as assets and liabilities. The program works with two data diskettes—a current month diskette and a backup diskette that stores information on a year's transactions.

This program provides all the program most families will need for financial management. A small business that requires a minimal, relatively routine set of records will find this program attractive because it is simple to use and provides most of the information required. Pacific Coast Software's *General Ledger* program also runs on the Commodore 64 and provides all the software needed to set up a double entry bookkeeping system for a small business.

EASY FINANCE series (Commodore). Commodore distributes five different packages in the *Easy Finance* series. These programs are designed to help you with some of the most commonly encountered family financial issues. *Finance I*, for example, helps you evaluate various options for borrowing money (*e.g.*, interest rates, term of loan). Other programs in the series help you deal with investment decisions. There is also a business oriented program that helps with business decisions such as lease versus purchase options and the selection of depreciation options.

Commodore, as well as several other companies, plans to offer programs for the Commodore 64 that will help you manage and

make decisions about your investment portfolio, and programs that let you keep an accurate inventory of your household possessions.

SPREADSHEET SOFTWARE

Companies with large or mainframe computers have routinely used spreadsheet software for years. These programs let you automate the process of performing tasks that require calculations where the formulas and formats remain the same, but the actual numbers used change. Examples of such tasks are job cost estimates, expense reports, loan and mortgage amortization tables, salary grids, break even analyses, inventory management, routine accounting tasks, and statistical analyses.

Several years ago some young fellows wrote and marketed a spreadsheet program called VisiCalc that ran on the Apple II and several other personal computers. Computer owners have now spent over 40 million dollars on that one program and there are at least 50 other programs in the VisiCalc tradition.

PRACTICALC (distributed by Micro Software International, 50 Teed Drive, Randolph, Massachusetts 02368, phone 617 961-5700, \$40 on diskette or tape). When we received our copy of PractiCalc our expectations were somewhat low. We have used VisiCalc and SuperCalc, two very powerful spreadsheet programs that cost as much as \$300 depending on the version. Could a \$40 program running on an under-\$400 computer compare favorably? PractiCalc does not display as much information on the screen as some of the spreadsheet programs for business computers, and it does not have some of the fancier features of the more expensive programs. It is, however, a thoroughly professional product that will function quite acceptably in a business environment. The manual is easy for novices to follow, yet it provides useful information for advanced computer users. The program itself has all the basic features of more expensive spreadsheet software and even has some features we preferred. For example, odd numbered columns are in a different color than even numbered columns and the "active" cell is blue. The company that distributes this program plans to market a database management program, a check writing program, and a program that lets you compose music on the Commodore 64. If PractiCalc is an example of the quality we can expect from Micro Software International, they should be well worth considering.

CALC RESULT (Distributed in the U.S. by Computer Market-

ing Services, 300 W. Marlton Pike, Cherry Hill, New Jersey 08002 Phone 609 795-9480, diskette and cartridge—\$150). Calc Result is expensive when compared to the typical Commodore 64 program, but it is inexpensive when you compare it to the cost of a similar program on other computers. It is a very powerful spreadsheet program. When you execute this program it asks you which language you want the messages to appear in—German, French, Italian, Swedish, English, Spanish, or Dutch. The program was developed by Handic Software of Sweden and is marketed internationally.

This is a premium quality spreadsheet with features lacking in spreadsheets programs that cost \$400 and more. Like PractiCalc it puts the color capabilities of the Commodore 64 to good use. It has a Help function that allows you to press a key and get an explanation of commands, it lets you divide your screen into several different zones and independently manipulate the material displayed there, and it will print a bar graph that represents the data in your spreadsheet. The only major complaint about this program relates to the manual. The manual is comprehensive and includes several examples of how a spreadsheet program could be used. It was written in another language and seems to have been poorly translated. The result is a manual that, in places, is confusing and unclear. All the information is there, but you may have to work a bit to find it and to understand exactly what the instructions mean. Even with that fault it is still a very good spreadsheet program.

Computer Marketing plans to distribute a complete line of Commodore 64 software including a statistical analysis program (\$50), a word processing program called Script-64 (\$140), a checkbook manager (\$35 on diskette, \$30 on tape), and an appointments organizer called 64-Diary (\$60).

OTHER BUSINESS AND PROFESSIONAL SOFTWARE

TIME MANAGEMENT (Totl Software, diskette or tape—\$30). If you and/or your colleagues perform work that requires precise scheduling of sequenced activities, this program (called **TOTL TIME**) may be of great help. It can plan the work of a team of individuals who are working on several different projects. Different types of work are recorded separately and reports or charts are generated in many different formats such as by project, by individual, or by activities. The program can produce several types of summary charts (in color on the display). These charts can be

printed on virtually any printer that can be attached to the Commodore 64.

This is not a simple program, and thus not something you are likely to use for relatively straightforward scheduling of appointments. If, on the other hand, your work involves something like building houses where teams or individuals with different skills (*e.g.*, electricians, carpenters, carpet layers) must be scheduled for work, then **TOTL TIME** may be very useful.

RESEARCH ASSISTANT (Totl Software, cassette or disk—\$30). If you keep card or folder files of information you must decide how those files are to be arranged. This program lets you type in information in much the same way you would write information on 3 × 5 cards. Then it permits you to search through your electronic “cards” by giving the computer a keyword or words that will guide the search. That means the same file can, in effect, be organized along several dimensions such as by student name, by major, and by the courses the students are taking (or have already completed).

We found this program somewhat complicated to use but it is one of only a few database management programs available for the Commodore 64 and worth considering. A program that appears similar is **DATA MANAGER** (MicroSpec, 2905 Ports O’Call Court, Plano, Texas 75075, disk—\$60).

INQUIRE PAC (Pacific Coast Software, diskette—\$70). This program allows the user to set up a sort of electronic file cabinet that contains organized information. It allows you to set up your own set of files and then search through them electronically and print out the information you are searching for.

This is not a sophisticated program, but its simplicity makes it easier to use than many and if you do not need the complex features, it may be just the thing.

The company that developed *The Businessman* software described earlier (Southern Solutions) also has several other programs:

The BILLPAYER. This program computerizes invoice entry, recording, and payment as well as check recording, and printing. Like the other programs listed below it can be used with *The Businessman*.

THE BILLCOLLECTOR. It keeps track of unpaid invoices, prints invoices and an aging report, and calculates finance charges on unpaid accounts.

THE PAYMASTER. This is a payroll program that calculates and stores salary and tax data and prints W2 forms.

LANGUAGES AND PROGRAMMING AIDS

The Commodore 64 computer speaks the computer language BASIC because it is the language installed in ROM at the factory. BASIC is a very good language, but it is not the only computer language, nor is it the best for every application. Software is available that lets your Commodore 64 speak other computer languages. Many programs also add flexibility and ease of use to the Commodore 64 when you program it in BASIC or another language.

C64FORTH (Performance Micro Products, 770 Dedham Street, S-2, Canton, Massachusetts 02021, phone 617-828-1209, disk or cassette—\$100). FORTH is a language which is growing in popularity among small computer users. It is more structured than BASIC and some people believe that it provides programmers with more power and flexibility than is available with BASIC. C64FORTH has all the key words of the FORTH 79 standard and also has provisions for using the Commodore 64's color and sound features. Another version of the language, 64-FORTH, is available in cartridge form from Computer Marketing Services for \$60.

PILOT (Commodore, \$59.95). PILOT is a language that was developed for educators who want to create computer assisted instruction programs. It is ideal for that application and is rapidly becoming a first language for elementary and junior high students because it is easy to learn to use. Commodore's PILOT for the Commodore 64 will have the standard PILOT keywords plus keywords for color graphics and music synthesis.

VANILLA PILOT (Tamarack Software, Box 247, Darby, Mt. 59829, diskette—\$30). We paid \$30 for this program at a store in Dallas and expected very little for our money. But we found that it is an enhanced, superior version of PILOT that takes full advantage of the Commodore 64. It even has commands for "turtle graphics" the most popular aspect of Logo. The manual is well written and informative, the program is one of the best implementations of PILOT we have seen. The program would have been a bargain even at \$60. If you want to learn a programming language but are a bit concerned about BASIC and its mathematical orientation, consider Vanilla Pilot.

LOGO (Commodore, \$59.95). Logo is "hot" right now because many people believe it is THE language for young children. A very positive feature of Logo is the ability to "teach" the computer to do something (*e.g.*, draw a red balloon and make it float across a blue

sky) and then create a new keyword for that activity. If you have young children Logo may be a language you want to explore. The version marketed by Commodore was created by the company that developed one of the versions for the Apple II computer. We understand the Commodore 64 version is even better.

PROGRAMMING AIDS

SUPEREXPANDER (Commodore, \$59.95). Superexpander is a utility program that adds several new key words to the BASIC used by the Commodore 64. This program makes programming sound and graphics much easier. Our only complaint is that the key words available when the Superexpander program is used should have been a part of the standard BASIC in the Commodore 64 and not an extra cost option.

PETSPEED (Small Systems Engineering, 222B View Street, Mountain View, California 94041, phone 415-964-8201, diskette—\$150). The BASICs used in most small computers, including the Commodore 64, are “interpreters.” That is, they look at each line of instruction in a BASIC program, convert that line to a set of machine language instructions the computer can execute, and then move on to another line. If a program goes to line 200 in a program 15 times during execution, line 200 will be converted to machine language instructions 15 times. Another type of BASIC, called compiler BASIC, makes the conversion only once and thus runs as much as 100 to 300 times faster than BASIC programs. PETSPEED is a popular compiler BASIC for Commodore computers and has been well received by programmers who want to develop programs in BASIC but need the speed of machine language for execution.

In addition to the software described above, several programs that help you develop graphic and music programs are described in Chapter 8. Because the Commodore 64 is growing in popularity, we can expect to see many more programs in this category soon. There should be several versions of languages such as Pascal as well as a number of programs that enhance or expand the capabilities of the BASIC used by the Commodore 64.

EDUCATIONAL SOFTWARE

There are several several hundred educational programs for the Commodore 64, and many of them are available free of charge. (See

the section on public domain software above.) In this section we will review some of the commercial educational software.

TYPING TUTOR plus WORD INVADERS (Academy Software, P.O. Box 9403, San Rafael, California, phone 415-499-0850, cassette—\$21.95). You get two programs for the price of one on this tape. Typing Tutor is a drill and practice program that teaches you how to touch type. It has 8 levels of difficulty and is a colorful, well written typing tutor program. We recommend it for anyone who would like to learn how to touch type. The program keeps track of your errors as well as typing speed and it highlights in red all the letters you typed incorrectly in an exercise.

Word Invaders is an excellent game that lets you practice your typing skills in a video game environment. Lines of words come marching down the screen toward the bottom. You must type them correctly to erase them from the screen. If any lines of words reach the bottom, you lose. The program keeps track of your speed and assigns a game score based on speed and accuracy. The program was highly rated by Creative Computing magazine.

THE HAPPY TUTOR (Pacific Coast Software, diskette—\$20). This is also a typing tutor program. It allows you to practice typing letters or sentences. It also has provisions for composing your own set of sentences for practice. The computer generates a sort of typing sound as you work and keeps track of errors made, your typing speed on the last sentence, and your average speed for the last set of sentences. This program does the job and makes typing practice fun. Our only criticism is about the way the instructions are slowly printed on the screen. We would rather see the instructions appear more quickly than listen to the simulated sound of a typewriter. It is still a program well worth the \$20, however.

VISIBLE SOLAR SYSTEM (Commodore, cartridge—\$40). This program lets you pilot your spaceship on a journey through the solar system. It makes good use of the color graphics features of the Commodore 64 to display a model of our solar system. As you pilot the ship you learn about the location of planets and their relationships to other planets. It is a good educational program with an informative manual.

EASY LESSON and EASY QUIZ (Commodore, \$30). These two programs are part of a complete line of educational software which will be marketed by Commodore for the Commodore 64. Easy Lesson and Easy Quiz let you build a file of questions (true, false, or multiple choice). Then when you want to create a test the computer will “build” one for you.

TELECOMMUNICATIONS SOFTWARE

An increasing number of personal computer owners are using their machines to gain access to information and services available from other computers over ordinary phone lines. To use the Commodore 64 for telecommunications you will need a modem and a telecommunications software package. Modems are discussed in the chapter on hardware and accessories. A telecommunications program lets you send instructions to another computer over the phone and receive information from the other computer. Incoming information is displayed on your video display. You may also send it to your printer, store it on cassette or diskette, or place it in the memory of the computer.

There are several telecommunications programs for the Commodore 64. One of the best programs is TERMINAL from Midwest Micro Associates (P.O. Box 6148, Kansas City, Missouri, 64110, cassette—\$30). The manual is a tutorial on telecommunications as well as the use of the program. Midwest Micro also sells a more sophisticated program called SuperTerm-64 that, among other things, lets you store material in the buffer on tape or diskette for use later.

This concludes our discussion of software. We hope that it has provided you with an idea of the range of software available for the Commodore 64. Remember—new (and often better) Commodore 64 software is appearing every week!

CHAPTER 11

Selecting Hardware and Accessories

This chapter explains what many of the accessories for the Commodore 64 do, what they won't do, and points out some of the pitfalls and problems associated with buying accessories.

There are many accessories for the Commodore 64. Shop for service and fair prices. Buy carefully.

ROM EXPANSION BOARDS AND PROM BURNERS

If you create a marvelous program for your Commodore 64 and want to install it permanently in a Read Only Memory chip, you can use an accessory to do so. The PromQueen, a device that plugs into your cartridge slot, lets you create the programmed ROM chips, which are the heart of video game cartridges. It is sold by Gloucester Computer Bus Company (6 Brooks Road, Gloucester, Massachusetts, 01930). The PromQueen has 4K of RAM to use as you develop your program. There is also a special socket for inserting blank programmable ROM chips (2716 or 2732) and a power supply. An accompanying program called Hexkit 1.0 makes it easier for you to write machine language programs and place your program in a programmable **ROM**. This product requires some professional experience in machine language programming to use. It retails for \$300.

PRINTERS AND PRINTER INTERFACES

The 1525 Graphic Printer, marketed by Commodore (\$400) is a relatively inexpensive Japanese import modified to operate cor-

rectly with the VIC and Commodore 64 computers. It can print all the Commodore 64's graphic characters as well as letters, numbers, and symbols. You will not need special cables or interfaces. Plug it in, turn it on, and it works. The 1525 is relatively slow and print quality is somewhat mediocre.

At least 200 different models of printers on the market today range in price from under \$100 to several thousand. Quality varies from barely legible print to that of a high quality office typewriter. (See *Computers for Everybody* by Jerry Willis and Merl Miller, dilithium Press, Beaverton, Oregon, for more information.)

To successfully connect a printer, you must overcome three major obstacles:

1. The printer must accept data in a form that the Commodore 64 uses, or you must place an interface between the Commodore 64 and the printer.
2. The cable between the computer and printer must have the proper connectors on each end.
3. The code used by the Commodore 64 and the printer must be the same, or a program must translate the codes for the printer.

To deal with the interface issue first, printers generally come in two forms—serial and parallel. Serial printers accept data a bit at a time with each character code being made up of seven or eight bits. Parallel printers accept data eight bits at a time over eight separate data lines. The Commodore 64 has a serial interface port (the round **DIN** plug with six holes on the back), where you can connect a serial printer. Every serial printer (except the 1525 from Commodore) uses an industry standard voltage level when sending data to the printer. It is called the RS232C standard and uses -12 and $+12$ volts. The Commodore 64 and the 1525 printer use 0 and $+5$ volts instead. That means this interface is not compatible with 99 percent of the serial printers available. If you intend to use a serial printer other than the one available from Commodore, buy an RS232C interface cartridge. The serial interface, sold by Commodore for about \$50, plugs into the user port on the back of the computer. It is wired to operate with a modem rather than a printer. Generally the cable to the printer will have to be rewired.

Parallel printers are also a problem since the Commodore 64 does not have a standard parallel interface connector. You can buy parallel interfaces, however, for \$50 to \$80. Cables are also a frequent problem. Most serial printers use a special connector called a 25 pin D connector. Most computer stores carry cables with

this type of connector on each end. The Commodore RS232 interface has a plug that fits this type of connector.

Most parallel printers use one of two types of connectors, the Centronics plug pioneered by Centronics, a U.S. printer manufacturer, and another type of connector that requires a cable with an "edge card" connector on it. These are often hard to find. In any case you will have to build or buy a correctly wired cable that plugs into your Commodore 64 on one end and fits your printer on the other. Finding a cable that works is sometimes the most difficult aspect of setting up the printer.

The code used by the 64, unfortunately, varies from the industry standard code, ASCII (short for American Standard Code for Information Interchange). The code for letters and numbers is the same, but there is no ASCII equivalent for the Commodore 64 graphics codes (most print as letters or symbols). A few of the Commodore 64 codes tell the printer to do things you would rather it not do (such as stop printing and go into a resting state). The 1525 printer is set up to accept the same code as the VIC and Commodore 64 computers. (Note: In 1983 Commodore announced the 1526, which is more expensive than the 1525 but a better value since its print quality is much higher and it prints faster than the 1525. It is code and format compatible with the Commodore 64 and comes with a cable that connects directly to the computer.)

SOME SOLUTIONS

A program called Smart ASCII from Midwest Micro Associates (P.O. Box 6148, Kansas City, Missouri, 64110, 816/254-9600, \$60), includes a program on cassette tape that can be transferred to disk, and a cable. The cable plugs into the Commodore 64 user port. The other end plugs into many of the popular parallel printers.

Smart ASCII solves all three problems noted above. A parallel port is provided, the cable works with many standard printers, and the program provides three different types of translation from which to select. The Commodore 64 code that tells the computer to clear the screen is translated as **CLR**, for example. The code for switching to reverse characters prints as **RVS** (Reverse Off prints as rvs), and the code to set character color to red predictably prints as **RED**. Smart ASCII comes with a clearly written and informative instruction manual. The program is compatible with several of the more popular word processing programs for the Commodore

64, an important point since some printer driver programs operate only with BASIC programs.

The Tymac Parallel Printer Interface is distributed by MicroWare Distributing. It is \$119 and comes with a printer cable that has a Centronic connector. The cable plugs into the serial socket on the Commodore 64. Its features include 2K of buffer memory which means the computer can send 2000 characters to the interface and then begin another task while the buffer takes over the job of sending character codes to the printer. This interface can duplicate Commodore 64 graphics on many printers, but you must specify which printer you will be using when the interface is ordered. MicroWare also sells a \$20 interface and cable that is a bare bones parallel interface that does not allow the printer to print the graphics symbols of the Commodore 64.

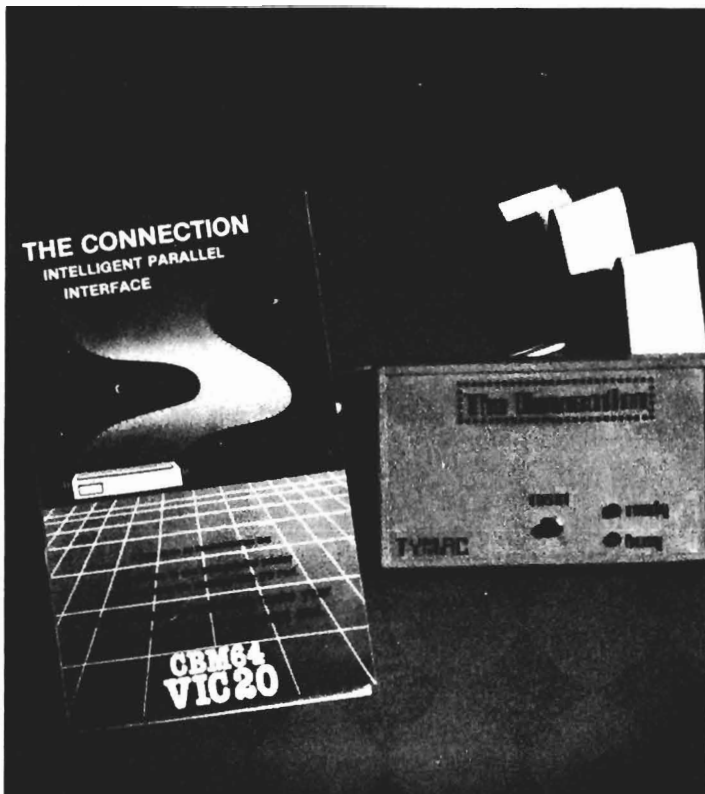


Figure 11.1 The Tymac parallel printer interface.

Other Companies that manufacture parallel printer interfaces include Cardco (\$80), OEM (\$100), and Micro World Electronix (\$120—6340 W. Mississippi Avenue, Lakewood, Colorado 80226, phone 303 934-1973). Serial printer interfaces are manufactured by Data20 (\$70, includes switches so you don't have to rewire the cable), and OEM (\$40).

Do not shop for interfaces simply on the basis of price. Decide what type of printer you will use, what software you plan to use it with (*e.g.*, only BASIC, accounting software, a word processor?) and determine if the interface is compatible with that software, and decide what you want the interface to do (*e.g.*, translate Commodore 64 code to ASCII, print Commodore 64 graphics). Then select the interface that does the job you need done.

VIDEO ENHANCEMENT HARDWARE

The Commodore 64's video display features are good for recreational use. Business applications, as well as word processing, would profit from a higher capacity screen display. A 24 line by 80 character screen display is relatively standard on most of the popular business computers today (*e.g.*, the IBM PC, Eagle II, Sanyo 1000, Radio Shack Model IV). When compared to the more expensive business computers, the Commodore 64's 25 line by 40 character display seems small.

Several companies have designed products for a 24 × 80 format. The VideoPak 80 from Data20 lets you use the Commodore 64 in a 24 × 40 color mode, or a 24 × 80 monochrome mode. You will need a video monitor if you use the larger screen display format. Data20 has another product, the Z80 VideoPak, that gives you a 24 × 80 display and the ability to run a popular operating system called CP/M used for business or professional purposes. CP/M (short for control program for microcomputers) is a program that has been customized to run on many different computers. The ability to run CP/M gives computer owners the opportunity to run literally thousands of programs which are CP/M compatible. Having the Z80 Video Pak does not mean, however, that you can run any CP/M program. CP/M software is available on diskettes, and, to operate on the Commodore 64, the material stored on the diskette must be in a format readable by the 1541 disk drive. Commodore has also announced a CP/M cartridge for the Commodore 64.

LIGHT PENS, TRACKBALLS, AND GAME PADDLES

Educational and recreational applications of the Commodore 64 sometimes call for input to the computer other than the keyboard. For example, many arcade games work better with a joystick than if you must press keys to control movement. Test drive several before buying. Joysticks vary considerably in feel and fit and can affect playing style and cause fatigue and low scores.



Figure 11.2 The Power Grip joystick from WICO.

Atari sells a joystick for about \$10 that is available almost everywhere. It will plug into the Commodore 64 with no modifications. A variety of formats and styles are available from such

companies as Balis, Datamaster, Discwasher, Spectravisio, Suncom, and Wico. Wico has a sturdy, well-built joystick with two strategically placed fire buttons. This same company also produces a delux joystick with interchangeable handles.



Figure 11.3 The WICO joystick has three different handles.

A trackball game controller marketed by both Wico and TG Products can be substituted for a joystick on the Commodore 64. Trackballs let you roll a large round ball set in the top of the controller to direct game action with more precision and speed than most joysticks. Trackballs are generally much more expensive than joysticks.

A light pen, much like a ball point pen, is another device that may interest you. When properly interfaced to the computer, it can determine where on the screen the user is pointing the pen. A pen from 3 G Company (Rt. 3, Box 28A, Gaston, Oregon, 97119, 503/662-4492, \$40) comes with a demonstration program and instructions for incorporating the pen in your own programs. The demonstration program displays multiple choice questions on the

screen. To select an answer you place the pen on the answer on the screen. Used in education, light pens give computer access to students who are too young to type and to handicapped students unable to operate a keyboard. Programs that deal with menu selection, inventory control, and data entry as well as video games can use light pens effectively.

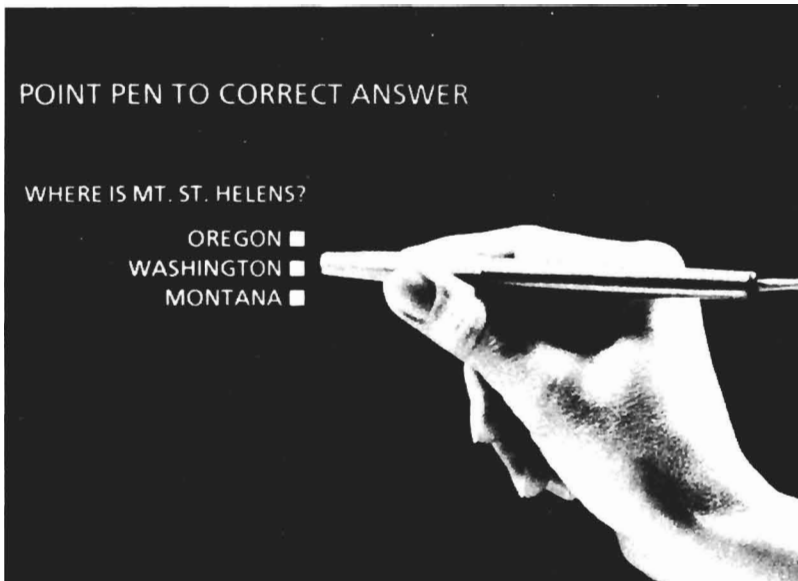


Figure 11.4 The 3G light pen.

MODEMS

A modem (modulator-demodulator) is a device that lets you connect your computer to other computers over the phone. A modem converts the signals from your computer into tones that can be transmitted and it also converts transmitted tones into signals the Commodore 64 can process.

Connect modems to the computer with a plug directly into the computer or a serial port. Since the Commodore 64 does not have a

standard RS-232 port, the most inexpensive way to attach a modem is to buy one with a direct connection such as Commodore's VICmodem that costs around \$100. The VICmodem has a direct connection on the phone end also. Acoustic modems convert computer signals into audible tones which are transmitted and received by placing the handset of a standard phone in rubber cups on the modem. They don't work as reliably as models which plug directly into a phone outlet. Noise in the room can be confused as data and transmitted. Standard speed on most national systems at present is 300 bits per second but that is likely to speed up soon, perhaps as high as 1200 bits per second. At present, modems of that speed cost between \$400 and \$600. A modem recommended for the Commodore 64 today is a 300 BPS model for under \$100.

You can buy more expensive 300 BPS modems with fancy features such as auto dial, that tells the modem, "Call Joe." The modem dials the number for you and signals when the phone is answered.

To use your computer as a telecommunications device, you need software to convert your computer into a terminal that can communicate with other computers. (See Chapter 9.)

DISK DRIVES, CASSETTE RECORDERS AND SUBSTITUTES

Many computer manufacturers sell their basic computer at a very low price and charge high prices for accessories. Commodore has priced the 1541 disk drive for the Commodore 64 very competitively from the outset. The low priced Commodore drive has discouraged other manufacturers from designing drive systems for the Commodore 64. The 1541 can store over 160,000 characters of information on one side of a diskette and, although it is a somewhat slow disk drive, is a very reliable unit. The drive capacity can be increased by adding high capacity disk drives to the Commodore 64. (See IEEE Interfaces below.)

One commercially available alternative to the 1541 disk drive is the Stringy Floppy from Exatron (181 Commercial Street, Sunnyvale, California, 94086, 800/538-8559). It uses specially designed tape cassettes called wafers to store programs and data. It sells for under \$200, and works reliably and quickly.

Most of the software for the Commodore 64 is on cassette, disk, or cartridge. The Exatron unit gives you fast, reliable storage, but you won't be able to buy many programs in the wafer medium.

The Commodore has a very reliable tape recorder system designed specifically for the Commodore 64. Standard cassette recorders rarely **LOAD** and **SAVE** tapes reliably. Several companies sell hardware that lets you use a standard tape recorder with your VIC. MicroWare, for example, sells a \$50 interface device that allows you to connect two tape recorders to the interface and duplicate a tape. After long use, cassettes wear out so it is good to have "back up" ability for a tape that contains an expensive program. In general, however, be skeptical of tape interfaces for the Commodore 64.



Figure 11.5 The tape recorder interface from Tymac.

Cardco and Integrated Controls also sell recorder interfaces. Eastern House Software (3239 Linda Drive, Winston-Salem, North Carolina 27106, phone 919 924-2889) markets the CBM 64 Rabbit, cartridge (\$40) that speeds up the standard Commodore

recorder. It loads and saves data and programs at approximately six times its normal rate, a valuable feature to individuals who use the cassette frequently. Note: The Rabbit Cartridge speeds up loading of commercially produced software when you load it into the computer at its regular speed and then use the Rabbit Cartridge to save it on another cassette at the faster speed.

NETWORKS AND IEEE INTERFACES

Commodore's 1541 disk drive and 1525 printer both connect to the Commodore 64 through a serial port, a different method of connecting printers and disk drives than is used on other Commodore equipment. On the PET and CBM models peripherals are connected via an interfacing approach called IEEE-488, a very reliable and fast method preferable to the serial method used on the Commodore 64. The IEEE-488 standard is more expensive than serial port input/output which is why it is not used on the Commodore 64.

Three other companies offer an interface cartridge that lets the Commodore 64 operate disk drives and printers designed for IEEE-488 interfaces. You can use high capacity disk drives, such as the Commodore 4040, 8050, and 8052, on your Commodore 64 as well as high quality printers such as the Commodore 8023 and 6400. The IEEE-488 interface for the Commodore 64 plugs into the expansion port. Then a cable is run from the interface to printers and disk drives. Micro Systems Development (11105 Shady Trail, Suite 104, Dallas, Texas, 75229, 214-241-3743) sells an interface board for \$100. Although accompanying instructions are inadequate, the board is well constructed. This company sells mostly to dealers. (Southwest Micro Systems, 2554 Southwell, Dallas, Texas, 75229, 214-484-7836, sells to end users). The mail order price for the C64 IEEE-4888 interface is \$95.

Small Systems Engineering (1056 Elwell Court, Palo Alto, California 94303, 415/964-8201) sells a different type of interface called Interpod. This \$180 device connects to the Commodore 64 serial port by a cable. It connects Commodore disk drives and printers which use the IEEE-488 protocol to the Commodore 64 as well as standard serial printers, plotters, and speech synthesizers, which use the RS-232C standard. It has provisions for selecting the correct serial speed (called BAUD rate or BPS—bits per second) from the interface.

Richvale Engineering (10610 Bayview, Richmond Hill, Ontario, L4C 3N8, 416-884-4165) also manufactures hardware and software for the Commodore computer line. They produce a product called the RTC C64-LINK, which functions in much the same way as the Interpod described above. Both provide an IEE-488 interface for the Commodore 64 and both have standard RS232 serial interfaces. C64-LINK also has a parallel interface and additional software that lets you use the same disk drive commands that are used on the PET computers with 4.0 BASIC. The disk drive commands in 4.0 BASIC are much easier and more convenient to use than the ones available in Commodore 2.0 BASIC, the version used on the VIC and Model 64 computers. C64-LINK also has a program that makes it much easier to write and revise machine language programs. At \$185, it even comes with software that lets you plug in a standard modem to the serial port and access information utilities.

Networking

The networking concept may not be familiar to many readers, but it is an important one if you plan to use several computers in one location. Under normal circumstances if you planned to use eight Commodore 64 computers in a classroom where computer literacy courses were taught, you would need to provide at least one disk drive, or one cassette unit, for each Commodore 64. Peripherals such as disk drives generally cost much more than the computer! One way to save is to use a resource sharing network system. That means you can connect several Commodore 64's to one or more disk drives and a printer. Since the disk drive is actually used less than 2 percent of the session and printer usage generally averages less than 5 percent of a typical computing session, it is practical to let the computer share these resources rather than have a disk drive and printer for each keyboard. It is possible to unplug the disk drive and/or printer after use and plug it into another computer which needs access. However, this wears out the connectors. There are networking accessories that let you interface to one or more disk drives and printers. One of the least expensive systems is the VIC/64 Switch, manufactured by Computer Marketing Services (300 West Marlton Pike, Cherry Hill, New Jersey, 08002), that sells for \$150 and allows you to connect up to 8 VIC and/or Model 64 computers. You can then connect 1540 or 1541 disk drives and a compatible Commodore printer to the VIC/64 Switch and start computing. Whenever one of the eight

computers needs printer access or the disk drive, the interface connects the computer to the peripheral and proceeds as if the disk or printer was connected directly to the computer. If another computer asks for access at the same time, the interface puts that computer in a wait state until the first computer finishes. This is a very inexpensive means of resource networking. The only expense involved is the VIC/64 Switch and cables from the computers to the interface.

NEW COMMODORE PRODUCTS FOR THE COMMODORE 64

Color Monitor—Commodore has begun to distribute a 12" color monitor that is compatible with the VIC 20 and the Commodore 64. It is very well designed, especially when you consider the \$299 price tag. The monitor has been specially designed to take advantage of all the video control signals which are generated by the VIC 20 and the Commodore 64. It provides clear, crisp color displays and good sound. Since monitors can cost as much as \$900, we felt the quality of this monitor was more than adequate considering its cost.

Color Plotter—This is another aggressively priced peripheral that works with the VIC 20 and the Commodore 64. For \$200 you get a printer/plotter that will print standard characters and draw high resolution charts, graphs, and illustrations in up to four colors. It uses 4.5" wide paper and prints in black, purple, green, and red. The 1520 plotter can be used to create high resolution graphics and text in color. Very nice for the price.

Music Keyboard—Until now music synthesizers that had their own piano-style keyboard cost at least \$600, usually more. Commodore plans to sell a music system with just such a keyboard for under \$100. Early prototypes seem to be well designed and come with very good software for creating and playing your own compositions.

ORGANIZERS AND USEFUL ACCESSORIES

Some optional accessories can make using your computer easier, more productive and efficient. Desk top organizers, cases, and dust covers are made expressly for the Commodore 64.

Carrying Cases—Computer Case Company (5650 Indian Mount Court, Columbus, Ohio, 43213, 800/848-7548) manufac-

tures and distributes a case for a Commodore VIC or 64 computer and 1540/1541 disk drive for \$129. These well-made cases protect the computer in transport but are not to be checked as luggage on an airplane.

Southern Case Company (2315 Laurelbrook Street, P.O. Box 28147, Raleigh, North Carolina, 27611, 800-334-0551) manufactures cases for the 64 with space for the keyboard, a cassette recorder, the power supply, four cassettes, and two joysticks. A rugged unit of twin walled plastic with foam rubber between the two walls, the case costs \$80 and can be checked baggage if necessary. Another model from Southern Case has provisions for the keyboard and a disk drive, is also \$80.

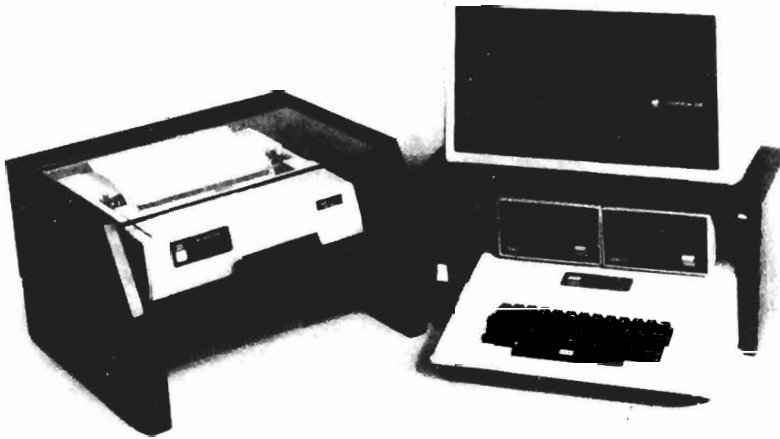


Figure 11.6 A printer enclosure from Cab-Tek.

A desk organizer is another handy accessory. The Micro Powerbench from Cab-Tek (Riverside Street, Nashua, New Hampshire, 03062, 603/889-1961) slips over your computer and provides a place for the television, or monitor, to sit and has room for the disk drives inside the unit. It has an on/off switch and a power strip on the back to plug in the power supplies for your computer, several disk drives, and a printer. That means one switch shuts everything off. The basic unit costs \$160 to \$180 and, for an additional cost, a power line filter and fan can be added. The same company also has

printer enclosures to keep dust and debris out of your printer and provides sound insulation for \$99 to \$199, depending on the size of the printer.

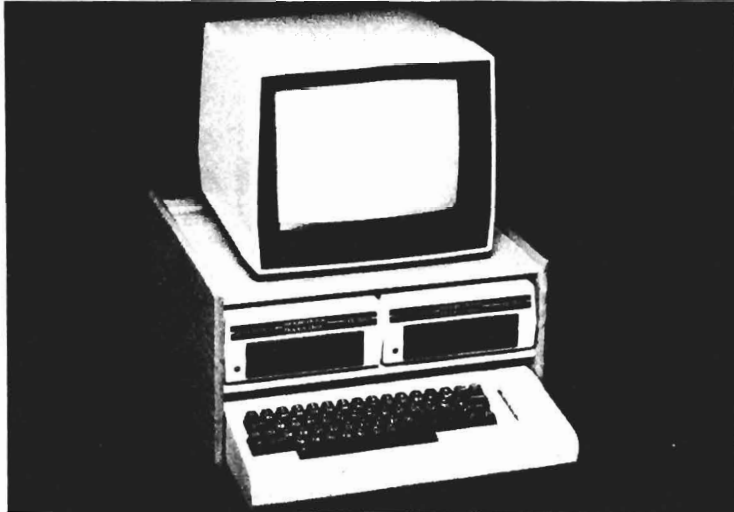


Figure 11.7 The DeskTopper from Madison Computer.

Madison Computer (1825 Monroe, Madison, Wisconsin, 53711, 608/255-5552) distributes a \$50 desk organizer with a shelf for your disk drive, room for manuals, and a flat top surface for the monitor.

CHAPTER 12

Sources of Additional Information

We hope this book has served as a good beginning point, an introduction, to the use of your Commodore 64 and to the field of personal computing. Yet we have only scratched the surface. There is more, much more, to be learned about your computer and about personal computing. The final chapter will provide you with some clues about how to get the additional information you need.

MAGAZINES

COMPUTE! This is an excellent reference magazine for owners of the Commodore 64 computer. Published monthly, it carries articles on the Commodore computers as well as others. Most issues contain at least two or three programs that will run on the Commodore 64 and numerous articles for intermediate or beginning users. A few are quite advanced. *Compute!* also has several regular columns on educational computing and programming in Logo. A typical issue will have articles on programming, reviews of commercial software and hardware, general articles on topics such as word processing, games, and the social implications of computers, and articles that include programs you can type in and use. Subscriptions are \$20 a year and there is a toll free number—800-334-0868, *Compute! Magazine*, P.O. Box 5406, Greensboro, North Carolina, 27403.

COMPUTE!'s COMMODORE GAZETTE. First published Summer, 1983, this magazine is aimed at novice owners of the Commodore VIC and Model 64 computers and has an article mix similar to *Compute!* Subscriptions are \$15 a year (12 issues). (See phone and address above.)

POWERPLAY. Published four times a year by Commodore, this magazine deals with the VIC and Model 64 computers and is \$10 a year. It is one of the best company magazines published. Well-illustrated, it publishes articles for beginning and intermediate users, and focuses on home and recreational computing. Each issue contains one or two programs as well as articles on a variety of topics. Most issues also carry a list of active user's groups for Commodore computers. Subscriptions are \$10 a year (Commodore Business Machines, The Meadows, 487 Devon Park Drive, Wayne, Pennsylvania, 19087).

COMMODORE. Published by Commodore, this magazine concentrates on professional and business applications with the more expensive Commodore computers, but the magazine also published quite a few articles on the VIC and Model 64. It is \$15 a year for six issues.

MICRO. This magazine publishes well-written and edited articles relevant to owners of computers that use microprocessor chips in the 6502 and 6809 families, including Commodore's. It focuses on computer programming articles and on building accessories yourself. There are also many tutorial and general interest articles, however. Subscriptions are \$24 a year for 12 issues from Micro, 34 Chelmsford Street, P.O. Box 6502, Chelmsford, Massachusetts, 01824.

THE TRANSACTOR. This publication is aimed at the more advanced user. Although not as large as *Micro* all the articles are relevant to Commodore computers. While it publishes reviews of commercial software and hardware products, it focuses on technical articles that deal with programming issues and/or the hardware design of particular computer models. Many articles are about the VIC and Model 64. Published six times a year by Canadian Micro Distributors, 500 Steeles Avenue, Milton, Ontario, Canada L9T 3P7, subscriptions are \$15.

STRICTLY COMMODORE. This bi-monthly magazine, while not as flashy as others, carries lots of information for owners of Commodore computers. Subscriptions are \$18 a year for six issues. The address is 47 Coachwood Place N.W., Calgary, Alberta, Canada, T3H 2E1.

COMMANDER. This magazine is aimed at owners of Commodore computers. Subscriptions are \$22 a year from Micro Systems Specialties, P.O. Box 98827, Tacoma, Washington, 98498, toll free order number, 800/426-1830.

COMMODORE COMPUTING. An excellent British publication, it covers the entire line of Commodore products and has articles for beginning through advanced users. It is 12.5 British pounds a year if you live in England, 15 pounds overseas. (*Commodore Computing*, MAGSUB, Oakfield House, Perrymount Road, Haywards Heath, Sussex, RH15 3DH.)

Commodore 64 Users Group Newsletter. A new publication, dues are \$25 a year for cost of newsletter and magazine. For more information write to Commodore 64 Users Group, Suite 100, Corporate West, 4200 Commerce Court, Lisle, Illinois 60532, 312/369-6525.

BOOKS AND BOOK PUBLISHERS

Books on the Commodore 64 are plentiful. Some current publishers are:

dilithium Press, P.O. Box 606, Beaverton, Oregon 97075, 800-547-1842. Specializes in personal computing books for beginners and intermediates. Free catalog.

Creative Computing Press, 39 East Hanover Avenue, Morris Plains, New Jersey, 07950, 800/632-8112. A division of Ziff-Davis; good line of books.

Reston Publishing Company, 11480 Sunset Hills Road, Reston, Virginia, 22090, 800/336-0338. Division of Prentice-Hall, books for novice and intermediate computer users.

Howard W. Sams, 4300 West 62nd Street, P.O. Box 558, Indianapolis, Indiana, 46206, 317/298-5400. Publishes rather technical books in area of electronics, including computers; introductory and intermediate level books on personal computing; computer manufacturers' manuals; publishes the *Commodore Software Encyclopedia* (\$10) and *Commodore 64 Programmers Reference Guide*.

Macro Dynamics, 8950 Villa La Jolla Drive, Suite 1200, La Jolla, California, 92037, publishes *The Complete 64* (\$15) that describes hundreds of products for the 64 and reviews products and references for the Commodore 64.

In addition to books and magazines, users groups are excellent resources. One of the best is Toronto Pet Users Group, P.O. Box 100, Station S. Toronto, Ontario, Canada, M5M 4L6; annual dues \$20, covers club newsletter and free access to the club's library of

over 3,000 programs with a small handling and copying charge. M5M 4L6; annual dues \$20, covers club newsletter and free access to the club's library of over 3,000 programs with a small handling and copying charge.

SUPPLIERS

Hundreds of companies produce and/or sell products for the Commodore 64 computer. Should you hear about a great new product or program for the Commodore 64 and be unable to locate it, you can order products by mail magazine advertisements. Many products must be ordered from the manufacturer, but there are a few distributors who try to stock a wide range of products from many different suppliers. Some of these distributors even publish catalogs that describe the products they carry at a discount. Below is a list of some mail order distributors of Commodore 64 hardware and software.

Data Equipment Supply Corporation, 8315 Firestone Boulevard, Downey, California, 90241, 213/923-9361.

Olympic Sales Company, P.O. Box 74545, 216 S. Oxford Avenue, Los Angeles, California, 90004. 800/421-8045.

Mooseware Incorporated, P.O. Box 17868, Irvine, California, 92713.

SJG Distributors, 10520 Plano Road, Suite 206, Dallas, Texas, 75238, 800/527-4893.

Southern Audio Video Electronics, 1782 Marietta Boulevard, N.W., Atlanta, Georgia, 30318, 800/241-2682.

Computer Outlet, Park Place, Upper Level, 1095 E. Twain, Las Vegas, Nevada, 89109, 800/634-6766.

CompuSense, P.O. Box 18765, Wichita, Kansas, 67218, 316/684-4660.

Computability, P.O. Box 17882, Milwaukee, Wisconsin, 53217, 800/558-0003.

Programs International, Moravia Center Industrial Park, Baltimore, Maryland, 21206, 301/488-7719.

Southwest Micro Systems, 2554 Southwell, Dallas, Texas, 75229, 214/484-7836.

JMC, 1025 Industrial Drive, Bensenville, Illinois, 60106. Catalog, \$2, book list.

Universal Software, 185 Mulberry Street, Claremont, New Hampshire, 03743, 800/343-8019.

Harmony Video and Electronics, 2357 Coney Island Avenue, Brooklyn, New York, 11223, 800/227-8927.

Appendix A

Abbreviations for Basic Keywords

Command	Abbreviation	Looks like this on screen	Command	Abbreviation	Looks like this on screen
ABS	A SHIFT B	A	FOR	F SHIFT O	F
AND	A SHIFT N	A	FRE	F SHIFT R	F
ASC	A SHIFT S	A	GET	G SHIFT E	G
ATN	A SHIFT T	A	GET#	NONE	GET#
CHR\$	C SHIFT H	C	GOSUB	GO SHIFT S	GO
CLOSE	CL SHIFT O	CL	GOTO	G SHIFT O	G
CLR	C SHIFT L	C	IF	NONE	IF
CMD	C SHIFT M	C	INPUT	NONE	INPUT
CONT	C SHIFT O	C	INPUT#	I SHIFT N	I
COS	NONE	COS	INT	NONE	INT
DATA	D SHIFT A	D	LEFT\$	LE SHIFT F	LE
DEF	D SHIFT E	D	LEN	NONE	LEN
DIM	D SHIFT I	D	LET	L SHIFT E	L
END	E SHIFT N	E	LIST	L SHIFT I	L
EXP	E SHIFT X	E	LOAD	L SHIFT O	L
FN	NONE	FN	LOG	NONE	LOG

Com-mand	Abbrevi-ation	Looks like this on screen	Com-mand	Abbrevi-ation	Looks like this on screen
MID\$	M SHIFT I	M	SAVE	S SHIFT A	S
NEW	NONE	NEW	SGN	S SHIFT G	S
NEXT	N SHIFT E	N	SIN	S SHIFT I	S
NOT	N SHIFT O	N	SPC(S SHIFT P	S
ON	NONE	ON	SQR	S SHIFT Q	S
OPEN	O SHIFT P	O	STATUS	ST	ST
OR	NONE	OR	STEP	ST SHIFT E	ST
PEEK	P SHIFT E	P	STOP	S SHIFT T	S
POKE	P SHIFT O	P	STR\$	ST SHIFT R	ST
POS	NONE	POS	SYS	S SHIFT Y	S
PRINT	?	?	TAB(T SHIFT A	T
PRINT#	P SHIFT R	P	TAN	NONE	TAN
READ	R SHIFT E	R	THEN	T SHIFT H	T
REM	NONE	REM	TIME	TI	TI
RESTORE	RE SHIFT S	RE	TIME\$	TI\$	TI\$
RETURN	RE SHIFT T	RE	USR	U SHIFT S	U
RIGHT\$	R SHIFT I	R	VAL	V SHIFT A	V
RND	R SHIFT N	R	VERIFY	V SHIFT E	V
RUN	R SHIFT U	R	WAIT	W SHIFT A	W

Appendix B

Error Messages

Here is a list of the error messages generated by the Commodore 64 and a description of causes.

BAD DATA. String data was received from an open file when the program was expecting numeric data.

BAD SUBSCRIPT. The program was trying to reference an element of an array whose number is outside of the range specified in the DIM statement.

CAN'T CONTINUE. The CONT command will not work, either because the program was never RUN, there has been an error, or a line has been edited.

DEVICE NOT PRESENT. The required I/O device was not available for an OPEN, CLOSE, CMD, PRINT#, INPUT#, or GET#.

DIVISION BY ZERO. Division by zero is a mathematical oddity and not allowed.

EXTRA IGNORED. Too many items of data were typed in response to an INPUT statement. Only the first few items were accepted.

FILE NOT FOUND. If you were looking for a file on tape, and END-OF-TAPE marker was found. If you were looking on disk, no file with that name exists.

FILE NOT OPEN. The file specified in a CLOSE, CMD, PRINT#, INPUT#, or GET#, must first be OPENed.

FILE OPEN. An attempt was made to open a file using the number of an already open file.

FORMULA TOO COMPLEX. The string expression being evaluated should be split into at least two parts.

ILLEGAL DIRECT. The INPUT statement can only be used within a program, and not in direct mode.

ILLEGAL QUANTITY. A number used as the argument of a function or statement is out of the allowable range.

LOAD. There is a problem with the program on tape.

NEXT WITHOUT FOR. This is caused by either incorrectly nesting loops or having a variable name in a NEXT statement that doesn't correspond with one in a FOR statement.

NOT INPUT FILE. An attempt was made to INPUT or GET data from a file which was specified to be for output only.

NOT OUTPUT FILE. An attempt was made to PRINT data to a file which was specified as input only.

OUT OF DATA. A READ statement was executed but there is no data left unread in a DATA statement

OUT OF MEMORY. There is no more RAM available for program or variables. This may also occur when too many FOR loops have been nested, or when there are too many GOSUBs in effect.

OVERFLOW. The result of a computation is larger than the largest number allowed, which is $1.701418848 + 38$.

REDIM'D ARRAY. An array may only be dimensioned once. If an array variable is used before that array is DIM'd, an automatic DIM operation is performed on that array setting the number of elements to ten, and any subsequent DIMs will cause this error.

REDO FROM START. Character data was typed in during an INPUT statement when numeric data was expected. Reype the entry so that it is correct, and the program will continue by itself.

RETURN WITHOUT GOSUB. A RETURN statement was encountered, and no GOSUB command has been issued.

STRING TOO LONG. A string can contain up to 255 characters.

?SYNTAX ERROR. A statement is unrecognizable by the Commodore 64. This could mean missing or extra parenthesis, misspelled keywords, etc.

TYPE MISMATCH. This error occurs when a number is used in place of a string, or vice-versa.

UNDEF'D FUNCTION. A user defined function was referenced, but it has never been defined using the DEF FN statement.

UNDEF'D STATEMENT. An attempt was made to GOTO or GOSUB or RUN a line number that doesn't exist.

VERIFY. The program on tape or disk does not match the program currently in memory.

Index

- 3G Company 115
- 64 **FORTH** 104
- 64 **GRAFIX SAMPLER** 89
- 64 **PANORAMA** 88

- Abacus Software 88
- abbreviations 64
- Academy Software 88, 106
- Access Software 88
- accessories 109
- ACCOUNT PAC** 100
- accounting software 99
- assembly language 5
- attack/delay pattern 89
- Audio/Video Connector 9

- Balis 115
- BANNER/HEADLINER** 89
- BASIC** 5, 25
- BAUD** rate 119
- Bibliographic Retrieval Service 8
- BILLPAYER** 104
- BILL COLLECTOR** 104
- bit 17, 86
- blocks 42
- BREAK** 59
- business software 97
- byte 17, 86

- C64FORTH** 104
- Cab-Tek 123
- CALC RESULT** 102
- calculator mode 57
- Cardco 113, 118
- cartridge port 16
- cassettes 117
- cassette connection 31
- cassette I/O 16
- cassette storage 31
- CBM 64 Rabbit**
 - Cartridge 118
- central processing unit (CPU) 18
- Channel 47
- character codes 84
- CLOSE** 47
- CLR** 81
- CLR/HOME** 24, 58
- CMD** 51
- color 79
- color plotter 121
- comma 62
- Command 57
- COMMANDER** 126
- COMMODORE** 126
- Commodore Information Network 8
- Commodore International 2

- COMMODORE**
- COMPUTING** 127
 - Commodore Logo key 29
 - Commodore 64 Users Group**
 - Newsletter** 127
 - CompuServe 8
 - Compute!** 90, 125
 - COMPUTE!'s Commodore**
 - Gazette** 125
 - Computer Case Company 122
 - computer literacy 4
 - Computer Marketing
 - Services 102, 104, 121
 - Computers for**
 - Everybody** 110
 - conditions 73
 - CONT 61
 - CONTROL 82
 - correcting typing errors 23
 - CrossTech Graphics 88
 - CP/M 113
- DATA MANAGER** 104
- Data 20 113
 - Datawasher 115
 - DELETE key 34
 - device 39, 47
 - device number 39, 47
 - DIALOG 8
 - dilithium Press 110
 - DIN connector 10
 - directory 41
 - Directory of Online**
 - Databases** 8
 - Directory of Online**
 - Information Resources** 8
 - Discwasher 115
 - disk drives 117
 - disk drive connection 37
 - disk drive I/O 16
 - Disk Identifier (DI) 47
 - disk storage 37
 - Eastern House Software 118
 - EASY FINANCE** 101
 - EASY LESSON** 107
 - EASY MAIL** 99
 - EASY QUIZ** 107
 - EASY SCRIPT** 98
 - EASY SPELL** 64 99
 - edit 60
 - educational software 106
 - Education Circuit** 94
 - END 28, 60
 - Exatron 117
 - expansion boards 109
 - expansion buss 16
 - expansion port 17
 - execute 25
- FACEMAKER** 88
- FAST EDDIE** 96
- File Not Found 40
- file number 50
- FILE PAC 104
- FOR NEXT loops 73
- FOR NEXT TO 73
- format 47
- Fox Soft 88
- game paddles 114
- GET 68
- Gloucester Computer Bus
- Company 109
- GOSUB RETURN 75
- GOTO 27, 68
- graphics 29, 83
- GRIDRUNNER** 95
- hard copy 43, 49
- HES 95
- Hexkit 1.0** 109
- H.D. Manufacturing 97
- IF THEN 27, 73
- information utilities 8

- immediate mode 57
- INPUT 27, 66
- INQUIRE PAC** 104
- I/O (input/output) 16
- INST/DEL key 23, 60
- installation 9
- integrated circuits (IC) 2
- Integrated Controls 119

- Jupiter Lander** 96

- keyboard 21
- key words 56
- KNOCKOUT** 95

- left arrow key 23
- LET 65
- light pens 114
- line number 57
- LIST 42, 50, 60
- literal string 62, 66
- load 31
- logical file 50
- LOGO 105
- loop 69, 75

- machine language 5
- math expressions 70
- memory 17
- MICRO 126
- Micro Powerbench** 123
- Micro Software
 - International 101
- Micro Systems
 - Development 119
- Micro World Electronix 113
- Microcomputer
 - Applications 94
- MicroSpec 104
- MicroWare
 - Distributing 112, 118
- Midwest Micro Associates 88,
89, 107, 111

- Miller, Merl 110
- MISSING FILE NAME 48
- modem 116
- MOS Technology 2
- Music Composer** 90
- Music Keyboard** 121
- Music Machine** 90
- Musicmaster** 90

- networking 120
- networks 119
- NEW 41, 58
- null string 70
- numeric variable 65

- OEM 113
- OPEN 47, 50

- Pacific Coast Software 95,
100, 104, 106
- PAYMASTER** 104
- Performance Micro
 - Products 104
- PETSPEED** 105
- PILOT** 105
- POKE 79
- ports 16
- power connection 13
- power supply 14
- Powerbyte Software 99
- POWERPLAY** 126
- PRACTICALC** 106
- PRG 42
- PRINT 26, 53, 61
- printers 109
- PromQueen** 109
- public domain software 93
- publishers 127

- Quality Computer 90
- QUICK BROWN FOX** 97
- ? 64
- ?? 68

- Random Access memory
 (RAM) 18
RAPIDWRITER 97
 RCA phone plug 10
 Read Only Memory (ROM) 17
READY 13
 recreational software 94
 RF modulator 10
RESEARCH
 ASSISTANT 103
RESTORE 68
RETURN 22, 58
 Richvale Engineering 120
 right arrow key 23
RUN/STOP 33, 59
RTC C64-LINK 120
RUN 26, 57, 58
RVS ON 82
- SAVE** 43
SCREEN GRAPHICS 88
 screen memory 83
 semicolon 62
 serial interface 110
 setup 9
SEQ 42
SHIFT key 29
 Sirius Software 95, 96
SKI MAN 95
 Small Systems
 Engineering 105, 119
Smart ASCII 111
 Sound 89
 sound interface device 19
Sound Shaper 90
 Southern Case Company
 Southern Solutions 100, 104
 Southwest Micro
 Systems 119
 Spinnaker Software 88
 spreadsheet software 101
 Sprite-64 88
- Spectravision 115
Sprite Designer 88
Spritegraphics 87
SpriteMaster 64, 88
SpryteByter 88
SQUISH'EM 95
 Statements 56, 57
STOP 61
STRICTLY
 COMMODORE 126
 String variable 65
 Stringy Floppy 117
 subroutine 76
Suncom 115
SUPER EXPANDER 105
Super Term-64 107
 Suppliers 128
 sustain/release 90
- Tamarack Software 105
 tape recorder connection 12
 telecommunication
 software 107
TERMINAL 107
 TG Products 115
THE BUSINESSMAN 100
THE HAPPY TUTOR 106
TIME MANAGEMENT 103
 The Source 8
THE TRANSACTOR 126
 tone 89
TORPET 94
TOTL.TEXT 99
 TOTL Software 99, 103
 trackballs 114
Tymac Parallel Printer
 Interface 112
TYPING TUTOR 106
 user port 10, 16
 values 65
VANILLA PILOT 105

variable 65
variable name 27
VERIFY 44
VERIFY ERROR 49
VIC/64 Switch 120
video enhancement 113
video display chip 19
Video Pak 113
VISIBLE SOLAR
SYSTEMS 107
VICmodem 117
volatile memory 18

waveform 90
WORD INVADERS 106
WICO 115

Z80 VideoPak 113

MORE HELPFUL WORDS FOR YOU from dilithium Press



Bits, Bytes and Buzzwords

Mark Garetz

This book translates all the computer jargon you've been hearing into words you can understand. It explains microcomputers, software and peripherals in a way that makes sense, so your buying decisions are easier and smarter.

ISBN 0-88056-111-4

160 pages

\$7.95



Computers For Everybody, 3rd Edition

Jerry Willis and Merl Miller

In a clear, understandable way, this new edition explains how a computer can be used in your home, office or at school. If you're anxious to buy a computer, use one, or just want to find out about them, read this book first!

ISBN 0-88056-131-9

200 pages

\$7.95



Instant (Freeze-Dried Computer Programming in) BASIC—2nd Astounding! Edition

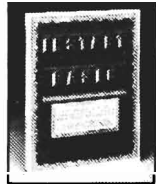
Jerald R. Brown

Here is an active, easy, painless way to learn BASIC. This primer and workbook gives you a fast, working familiarity with the real basics of BASIC. It is one of the smoothest and best-tested instructional sequences going!

ISBN 0-918398-57-6

200 pages

\$12.95



Are You Computer Literate?

Karen Billings and David Moursund

This is a fun introduction to computers for the real novice. Written by educators, it is a book that teaches the capabilities, limitations, applications and implications of computers.

ISBN 0-918398-29-0

160 pages

\$9.95



BRAINFOOD — Our catalog listing of over 130 microcomputer books covering software, hardware, business applications, general computer literacy and programming languages.

dilithium Press books are available at your local bookstore or computer store. If there is not a bookstore or computer store in your area, charge your order on VISA or MC by calling our toll-free number, (800) 547-1842.

Send to: dilithium Press, P.O. Box E, Beaverton, OR 97075

Please send me the book(s) I have checked. I understand that if I'm not fully satisfied I can return the book(s) within 10 days for full and prompt refund.

- Bits, Bytes and Buzzwords
 Computers For Everybody, 3rd Edition

- Instant (Freeze-Dried Computer Programming in) BASIC—2nd Astounding! Edition
 Are You Computer Literate?

Check enclosed \$ _____
 Payable to dilithium Press

Please charge my
 VISA MASTERCHARGE
 # _____ Exp. Date _____

Name _____

Signature _____

Address _____

City, State, Zip _____

Send me your catalog, Brainfood.

How To Use The **COMMODORE**[®] **64**[™] COMPUTER

Here are all the secrets you need to know to successfully and happily operate your Commodore 64. *How to Use Your Commodore 64:*

- Introduces you to the computer and its basic components.
- Tells you what the components do and how they work together.
- Gives you step-by-step instructions on how to set up or install your Commodore 64 and how to get it up and running.
- Shows you how to load and save your programs on diskette or standard audio cassettes.
- Tells you how to type in, use, and modify programs published in books and magazines.
- Presents you with other sources of information about the computer such as magazines, books, and users groups.

With the emphasis on practical information, this guide will be your constant companion as you learn to effectively use your Commodore 64 computer.



ISBN 0-88056-133-5

>>\$3.95

 **dilithium Press**
GENERAL INTEREST