
THE MICROCOMPUTER MAGAZINE
commodore
OCTOBER/NOVEMBER, 1982
$2.50
Halloween Hot Toddy Recipe
1 Fishbone
2 Rat Tails
1 Spider
2 Skunk Tails
1 Lizard
Orange Juice
Pumpkin
4 Frogs
2 Bat Wings
1 tsp Salt & Pepper

Software is the solution... for making your business more profitable!

And, the most comprehensive business and professional software is Info-Solutions from Info-Designs, Inc.

Info-Designs, Inc. is a rapidly growing company with a "commitment to provide the highest quality microcomputer software anywhere!"

Our Professional Practice Software Solutions provide a comprehensive management system for attorneys, engineers, architects, consultants, accountants, and other professionals who derive income from billing their professional services and reimbursable expenses.

Our management Accounting Software Solutions provide a totally integrated accounting system for the small to medium size business—retailer, wholesaler, distributor, manufacturer, and service company.

OFFICIALLY APPROVED BY COMMODORE

Now, on one compact Commodore CBM microcomputer system, you can achieve better management control of unbilled time, billings, accounts receivable, accounts payable, checkwriting, payroll, general ledger, inventory management and on-line order entry/point-of-sale.

See your local Commodore dealer today for the **Information-Solution** to your business automation needs.

InfoDesigns

INFORMATION SOLUTIONS USING MICROCOMPUTER TECHNOLOGY

6905 Telegraph Rd. ● Birmingham, MI 48010 ● (313) 540-4010

Circle #1 on Reader Service Card

THE MICROCOMPUTER MAGAZINE

commodore

FEATURES

Commodore Information Network is Growing Rapidly 20

Ontario Schools and Commodore Cooperate to Create Huge Collection of Educational Software 30

Boy Scouts' Pinewood Derby Uses Commodore Equipment to Time Races 43

Learn the Fundamentals of Writing Games for the VIC 20 61

What Happens When You Add the 64K Memory Expansion Board to Your 8032? 73

DEPARTMENTS

Q & A Hotline 2
Editor's Notes 4
From Where I Sit 5
Commodore News
Commodore Offers New Version of PETSPEED™ . . . 8
$5 Million Dollar Ad Campaign Kicks off Commodore 64 8
Broad Software Offerings for Commodore 64 8
VIC 20 Ad Campaign Challenges Video Game Industry 9
Commodore Approaches Industry Record 9
New Software for the VIC 20 9
Software Developers and the Commodore Software Encyclopedia 10
SuperPET Update 12
Commodore 64 User Guide Update 14
Frequently Asked Questions 16
Some BASIC Answers for the Commodore 64 19
Telecommunications: Present and Future 20
Captain Scuttlebutt's Unbelievable Rumors 23
User Clubs 24
Commodore User Group Flourishes in the Land of Lincoln 26
Education
Ontario Schools Create Phenomenal Educational Software Package 30
Kids Star at Connecticut Computer Schools 32
New Software Exchange for Educators 34
COMAL: Programming Language of the Future? . . 35
Business
Small Businessman No Longer Tied Up With Paperwork 40
The Other Office of the Future 41
A Timing System for the Pinewood Derby 43
VIC 20
The VIC Magician 57
Writing Games in BASIC 61
Meet the VIC 20, Part 2: What's in the Box 64
Programmer's Tips
Using a Joystick on the PET 68
Program Review
The Commodore BASIC Integer Compiler 71
Excerpts from a Technical Notebook
The Commodore 64K Memory Expansion Board . . . 73
New Products 77
Butterfield on Commodore 81
Projections and Reflections 95

commodore

Editor
Diane LeBold

Staff Writers
Bernard Falkoff
Paul Goheen
Jeff Hand
Neil Harris
Dan Kunz
Walter Kutz
Pat McAllister
Dave Moyssiadas
Ira Neal
John O'Brien
Mike Richter
Michael Tomczyk

Contributing Writers
Jim Butterfield
Mario Caprio
Elizabeth Deal
Miguel Feyjoo
Ron Straley

Technical Staff
Paul Goheen
Pat McAllister
Dave Middleton
Tom Rizol, Jr.
Michael Tomczyk

Circulation Manager
John O'Brien

Circulation and Advertising Assistant
Sharon Steinhofer

Advertising Manager
Diane LeBold

Cover art by Don Dyen

Q&A HOTLINE

Q. I recently purchased a CBM 8032 computer and 8050 disk drive and would like to know if I can use my audio cassette recorder to load and store programs. I would like to be able to send copies of my own programs to my friends who do not have disk drives. If I have to buy the Commodore datassette can I use it to record music?

A. *Commodore computers use what is called pulse width modulation to record programs and data on a cassette. Other microcomputers that you may be familiar with such as Radio Shack and Texas Instruments use what is known as FSK or frequency shift keying. In this type of recording the information being stored is recorded as two alternating high and low tones which represent the binary zeros and ones. Since these are audio tones that are being recorded and played back, it is possible to use a standard audio tape recorder with this technique. Pulse width modulation, on the other hand, is a technique that is less susceptible to noise and interference and is the type of recording used on large main frame computers for this reason. Pulse width modulation uses square waves to record information where the binary zeros and ones are represented by varying the widths of these waves or pulses. This is similar to the way that information is transmitted in RS-232 serial communications. Unfortunately, because pulse modulation requires special wave shaping circuits to generate these precise square waves it is not possible to use an ordinary audio tape recorder. Likewise, it would not be possible to use the Commodore datassette to record music because it has these special circuits.*

Q. I have a Commodore VIC 20 and want to know how to create bit mapped graphics on the screen. Is this possible with the VIC?

A. *The way to draw images on the screen, other than by using the built-in graphics characters, is to use programmable characters. The VIC allows the programmer to create up to 64 separately defined characters that can have any shape required. By carefully placing these special characters on the screen, it is possible to draw items such as pie charts, waveforms, etc.*

Q. I have a Commodore 8032. I recently called Customer Support with a problem. If the following program is entered, the results are very strange!

```
10 A=9000+43.51
20 K=9043.51
30 IF A<KTHEN PRINT"HELP"
```

The computer responds by saying A *is* less than K. Customer Support suggested I try using the rounding function, but because similar coding is to be used in an accounting program it is very important that I compare exact figures including cents. I was able to get around the problem by entering the program in the following manner:

```
10 A=9000+43.51
20 K=9043.51
30 A$=STR$(A):A=VAL(A$)
40 K$=STR$(K):K=VAL(K$)
50 IF A<KTHENPRINT"HELP"
```

Because in the first program A will never compare equal to K, I am concerned how this may unknowingly affect previous programs already online. Can you tell me what causes this

Continued on page 4.

Commodore—The Microcomputer Magazine *is published bi-monthly by Commodore Business Machines, Inc., 487 Devon Park Drive, Wayne, PA 19087. Phone: (215)687-9750. Domestic Subscriptions: six issues, $15.00. Application To Mail At Second-Class Postage Rates Is Pending At Wayne, PA 19087 and at additional mailing offices. POSTMASTER: Send address changes to* **Commodore—The Microcomputer Magazine,** *487 Devon Park Drive, Wayne, PA 19087.*

IF YOU'RE WAITING FOR THE PRICE OF WORD PROCESSORS TO FALL WITHIN REASON,

Everyone expected it would happen sooner or later...with WordPro PLUS™ it already has! Now all the marvelous benefits of expensive and advanced word processing systems are available on Commodore computers, America's largest selling computer line. WordPro PLUS, when combined with the new 80 column CBM 8032, creates a word processing system comparable to virtually any other top quality word processor available—but at savings of thousands of dollars!

New, low cost computer technology is now available at a fraction of what you would expect to pay. This technology allowed Commodore to introduce the new and revolutionary CBM 8032 Computer.

WordPro PLUS turns this new CBM 8032 Computer into a sophisticated, time saving word processing tool. With WordPro PLUS, documents are displayed on the computer's screen. Editing and last minute revisions are simple and easy. No more lengthy re-typing sessions. Letters and documents are easily re-called from memory storage for editing or printing with final drafts printed perfectly at over five hundred words per minute!

Our nationwide team of professional dealers will show you how your office will benefit by using WordPro PLUS. At a price far less than you realize.

Invest in your office's future...
Invest in WordPro PLUS...
Call us today for the name of the WordPro PLUS dealer nearest you.

Professional Software Inc.
166 Crescent Road
Needham, MA 02194
(617) 444-5224
TELEX: 95 1579

Circle #2 on Reader Service Card

Q&A HOTLINE

problem and if there is any other way to get around it? Thank you.

Maureen Collins
Bradford Woods, PA

A. *The apparent problem you are having is caused by the rounding inaccuracies of the floating point binary arithmetic conversion used on all microcomputers. In your first example, one number is entered and converted into floating point directly and the other value is derived by floating point addition. The results of this addition are not precisely equal to the number that was only converted to floating point, in this particular case. If you were to try different values, you are likely to get results that are equal when compared. This is similar to the problem encountered when multiplying or dividing on a hand calculator and receiving an answer of 9.999999 when the expected answer is 10. Your workaround is the correct way to get exact comparisons when working with accounting figures, etc. This is because you are comparing the literal numerals rather than the binary values. Your answers will always be correct if you use this comparison technique.* C=

Editor's Notes

In an industry like this one, where change is the norm, it shouldn't surprise you to learn that Commodore's user publications are continuing to change. Our rapid evolution has led us to consolidate the editorial staffs of COMMODORE and POWER/PLAY magazines. As a result of that consolidation, I will now be editing both our user publications. I'm looking forward to maintaining the level of excellence that Paul Fleming established as editor of COMMODORE—and, perhaps, even improving on it.

As you'll be able to tell by glancing through this issue, the Commodore 64 is the focus of everyone's attention right now. And with good reason. Shearson/American Express was right on target when they called the 64 "the microcomputer industry's outstanding new product introduction since the birth of the industry." Who else can do what the 64 can do for even close to the price? We all know the answer to that—nobody. And the most exciting part is that its full capabilities have only begun to be tapped.

Nevertheless, because of its versatility, the 64 presents an occasional editorial problem for us here in publications, because it crosses over what used to be a well defined line between our business-educational-scientific users—traditionally readers of COMMODORE—and our home users, who generally lean toward POWER/PLAY. We ran into that problem head-on when we tried to decide which audience should get the update on the Commodore 64 user manual (see page 14).

For the sake of the many readers who subscribe to both magazines, we'll try our best to keep from overlapping, but sometimes it will be inevitable when it comes to the 64.

You'll also notice that our user groups are multiplying faster than rabbits in a sunny meadow. So, if you've never had a user group in your area before, check the list again. You may find there's suddenly one nearby. As for

those of you who already participate in a user group—how about more input? We'll be featuring one group in every issue, and will publish the best from your newsletters, as well. And don't forget you can also communicate with each other via the Commodore Information Network bulletin board. The activity there has been astonishing, lately. A modem and a subscription to CompuServe will get you there in the time it takes to dial a local phone number.

While we're on the subject of the Commodore Information Network, we might mention that since we established our Network, subscriptions to CompuServe have increased significantly. We do seem to have quite a few active, enthusiastic users out there. As Jeff Hand points out on page 20, we'll continue to try to meet your needs with continual improvements in the Network, so you can get the most out of your Commodore computer via our electronic "paperless magazine."

Meanwhile, enjoy this issue of our regular paper-and-ink magazine. C=

Diane LeBold

Diane LeBold
Editor

From Where I Sit

by Dr. Dan Kunz, Project Manager, Educational Systems

I am pleased when I see teachers using microcomputers effectively. These tools can do many of the things that used to take up so much of my teaching time. They can balance a register at the end of the year when, in fact, it would often take me days trying to find that one absence that didn't show up in the cross balance. They can keep accurate track of student records, determine what objective a particular student is on, monitor who brought in their photograph money and generally aid in all the time-consuming information collecting activities teachers are responsible for.

More important, however, is that microcomputers are an aid to student learning. These amazing machines can present concepts and then support those concepts with practice exercises. They can eliminate much of the tedium involved in many learning activities, thereby letting the students focus on the real concepts. I love to see very young children performing rigorous computer-generated activities that were previously thought to be beyond their abilities.

A whole new world has been opened up to students because of the computer. I see students improving their decision-making skills, which I personally feel is one of the most significant contributions of microcomputers in the classroom. Virtually all microcomputer use aids a child's intellectual development, but it particularly improves decision-making skills via games, simulations and drill-and-practice exercises. The teaching of decision making, the teaching of logical thinking and even the improvement of psychomotor activities, such as eye-hand coordination, are all things for which computers are particularly well suited.

It's an exciting time in the field of microcomputers and in education. The opportunity exists to move students well beyond what we thought were their capabilities, and to do it in a logical and consistent fashion so we build upon the best learning theory, the best brain research and some of the best curriculum development efforts ever available.

It is a pleasure to be a part of the effort to help teachers by providing the best hardware on the market, and to be working very closely with many major publishers, so that in the near future even more of the major publishing houses will have curriculum materials and software available for Commodore microcomputers.

I find Commodore's general approach to be right on target, but specifically what do I see from where I sit? I see a modem that is so inexpensive that through telecommunications, it brings a complete resource center into every classroom. I see new support methods for the Education Resources Centers that are being identified across the nation. More immediately, I see a computerized telephone network available through the Commodore Information Network that would allow our Education Resource Centers to have their own electronic newsletter bulletin boards and wallboxes, so they can communicate with their colleagues around the country.

We also hope to ask the Education Resource Center teachers and administrators who are familiar with Commodore equipment and use it on a day-to-day basis to help us with state and regional shows for organizations such as teacher and administrator associations. These Centers will also help us evaluate curriculum software. Commodore believes it is vitally important that we have classroom teachers critique these materials, evaluate them and tell us what needs to be changed before we make them widely available.

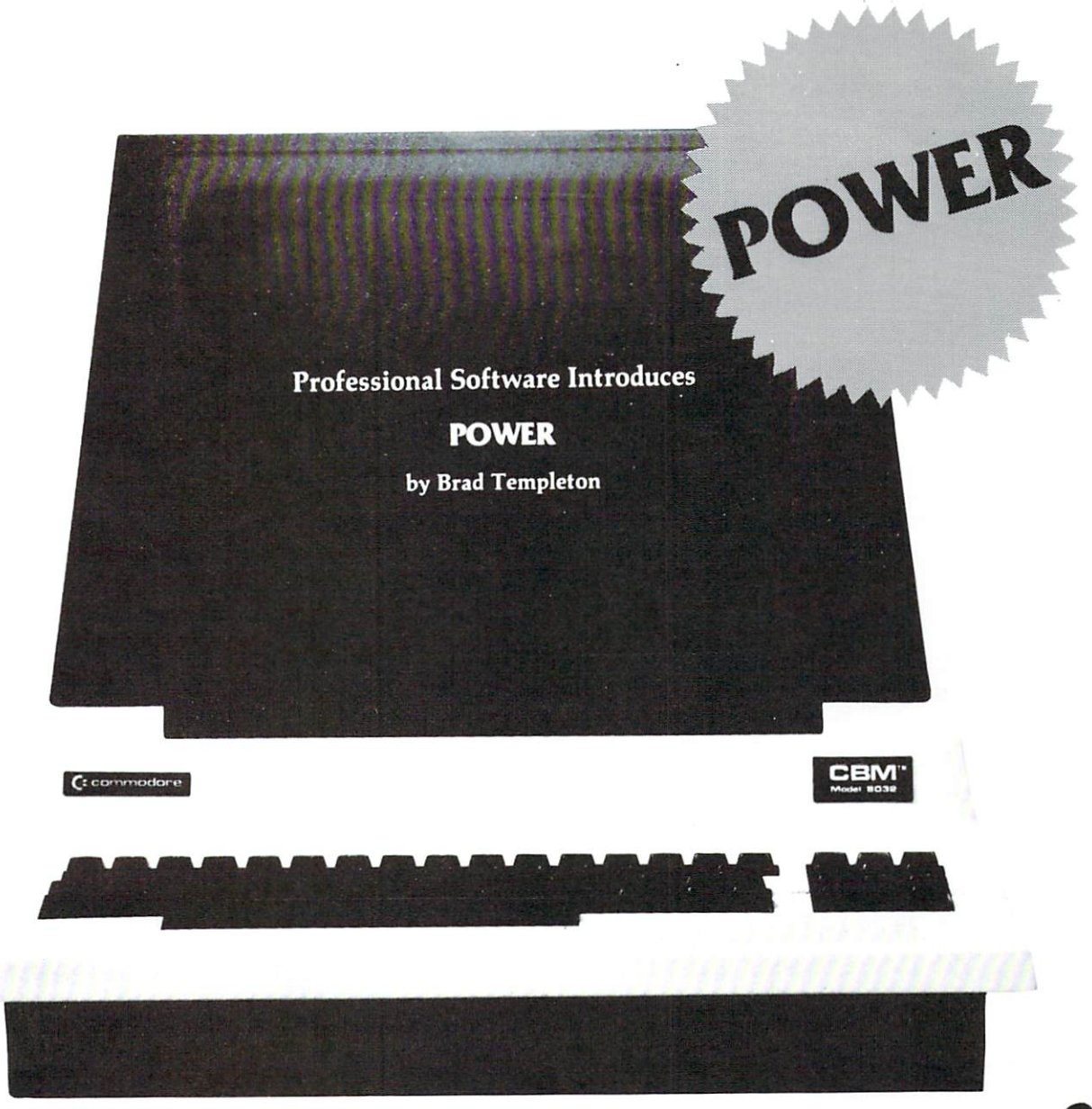
I also see a soon-to-be-advertised networking system that will connect up to 64 students in a classroom, and will allow the teacher to observe any student's screen from his or her desk and comment on that student's work by typing in a line or two that will be transmitted back to the student. In addition, if a teacher sees that a student has not quite gained mastery of the concept being presented, the teacher can freeze that student's screen and walk to them and deal with the problem immediately. The teacher does not have to wait for the student to ask for assistance or worry that the student will proceed incorrectly.

I see a vision of many things that can be accomplished because Commodore has committed itself to helping teachers in ways that teachers have requested and is, as a corporation, com-

mitted to making necessary changes to provide continually improved service to educators and the students they serve. The Commodore 64 will be the focus of our next education initiative. The 64's capabilities, extensive software and ease of use make it a perfect home and school learning tool. As technology moves ahead we will always be on the cutting edge but we will also support those people who have supported Commodore.

I anticipate a very hectic but stimulating time for everyone involved in microcomputers and education. I realize we all have a lot to do, but I will be periodically asking for your assistance as educational experts. I would also appreciate hearing from you. Your ideas and suggestions are important to us. You may write to me at Commodore, Computer Systems Division, 487 Devon Park Drive, Wayne, PA 19087. C=

Dan Kunz with students from the West Chester, Pennsylvania school district.



ADD POWER TO YOUR COMMODORE COMPUTER

$89.95

POWER produces a dramatic improvement in the ease of editing BASIC on Commodore's computers. POWER is a programmer's utility package (in a 4K ROM) that contains a series of new commands and utilities which are added to the Screen Editor and the BASIC Interpreter. Designed for the CBM BASIC user, POWER contains special editing, programming, and software debugging tools not found in any other microcomputer BASIC. POWER is easy to use and is sold complete with a full operator's manual written by Jim Butterfield.

POWER's special keyboard 'instant action' features and additional commands make up for, and go beyond the limitations of CBM BASIC. The added features include auto line numbering, tracing, single stepping through programs, line renumbering, and definition of keys as BASIC keywords. POWER even includes new "stick-on" keycap labels. The cursor movement keys are enhanced by the addition of auto-repeat and text searching functions are added to help ease program modification. Cursor UP and cursor DOWN produce **previous** and next lines of source code. COMPLETE BASIC program listings in memory can be displayed on the screen and scrolled in either direction. POWER is a must for every serious CBM user.

Call us today, for the name of the Professional Software dealer nearest you.

Professional Software Inc.
166 Crescent Road
Needham, MA 02194
Tel: (617) 444-5224 Telex #951579

Circle #5 on the Reader Service Card

'*POWER is a registered trademark of Professional Software, Inc.

Commodore Offers New Version of PETSPEED™

Relying on optimization to reduce program running times, the PETSPEED Optimizing BASIC Compiler runs Commodore BASIC significantly faster. After being reduced to small components, programs are reassembled to run in a more efficient form.

A four-pass compiler, PETSPEED is available for any combination of 4000 and 8000 series Commodore computers, using 8050 or 4040 disk formats. Unlike other compilers, it achieves major improvements in processing speeds by compiling BASIC applications instead of translating them from one language to another. The newest version, PETSPEED 2.6, uses three-dimensional arrays.

PETSPEED's easy initialization process makes it unnecessary to add compiler directives when compiling a new program. The user simply types in the program name. In less than two minutes, the program will be ready to run significantly faster than before.

PETSPEED automatically uses faster integer arithmetic whenever possible. Frequently occurring variables and arrays are handled automatically. All subroutines are automatically called at maximum speed, eliminating the need to locate subroutines at the beginning of a program.

A security device provided to run PETSPEED does not require a run-time key for a compiled program. Since PETSPEED code cannot be listed by others, users can control access to their programs and prevent tampering.

PETSPEED 2.6 is available from your Commodore dealer for $300.00. In California, it is also available from Small Systems Engineering, 222-B View Street, Mountain View, CA 94041.

Commodore Launches $5 Million Ad Campaign for 64K Micro

Commodore is introducing its Commodore 64 personal computer with a $5 million advertising campaign—the largest budget ever allotted by the company for a new product introduction.

The initial campaign, which began in August, will run through the end of 1982. It features both print advertisements in newspapers and magazines and television spots.

The television campaign will be the largest media launch ever experienced in Commodore's history. It will easily surpass the original television campaign for the VIC 20, and will include 30-second spots during prime time shows, major sporting events, family programming and late night entertainment.

Broad Software Offerings for Commodore 64

Addressing the needs of home, business and educational users, Commodore has developed a variety of software products for the Commodore 64, and will utilize a CP/M* option, thus providing access to over 2000 additional programs. Drawing on Commodore's international strength, many of the programs will come from the United Kingdom and Canada, as well as the United States.

Planned for release over the next sixty days: Computer Tutor and Assembler Tutor teach-yourself-programming packages; Easy Script word processor; Easy File data filing system; Name Machine and Word Machine (one package) for generating letters and other documents that require name-address headings; and an Assembler System that allows the experienced user to write programs in assembly language.

Also in the works are: Easy Calc electronic spreadsheet; Easy Finance for financial analyses; Easy Mail mailing list; and a BASIC compiler. For educators, Easy Lesson and Easy Quiz will help generate questions for classroom lessons and tests.

In addition to these Commodore-contracted software developments, Commodore has also encouraged outside vendors to develop software for the 64. According to Kit Spencer, vice president-marketing, over 200 software projects are already in progress worldwide.

"Because we are one of the few companies that design and manufacture our own chips, we've been able to get some of the world's best programmers to develop software for the 64 at a very early stage," Spencer said.

* CP/M is a registered trademark of Digital Research, Inc.

"Explosive" Commodore Computer Sales Put Industry Record in Sight

Recent "explosive" sales of its VIC 20 personal computer have put Commodore right on target with its prediction to sell more home computers in 1982 than all other companies sold the previous year.

According to Kit Spencer, vice president of marketing, the VIC 20 has the number one market share in the home computer industry and is one of the fastest selling products in consumer electronics.

"In January we predicted we would sell more computers this year than the entire industry sold in 1981 (approximately 800,000 computers). We're now well on our way to making this bold prediction a reality," he said. "The way we've done this is to make the VIC 20 the most accessible, most affordable and friendliest true computer on the market."

"We don't believe the 'home computer revolution' is coming," added Ron Glatz, vice president of consumer sales. "We believe it's here, and the VIC 20 proves it."

Commodore Challenges Video Game Industry with $20 Million Ad Campaign

Commodore is taking on the video game industry with a tough new $20 million advertising campaign to promote its VIC 20 home computer. Most of the budget—the biggest ever for any Commodore product—is allocated to prime-time television spots presenting the VIC as a superior alternative to video game machines, which cost about the same amount.

Ally & Gargano, Inc., Commodore's New York ad agency, has developed a variety of messages that emphasize the "true computing power" of the VIC compared to the limited capabilities of the game machines.

Kit Spencer, vice president of marketing, noted that some video game companies have started offering rebates. "That signals the end of the video game machine as we know it," he said. "From now on, people will buy real computers and settle for nothing less."

Newly Announced Releases Make VIC 20 the Best in the Computer Industry

The popular VIC 20 now has one of the biggest software libraries of any personal computer. Along with Commodore's newly released cartridges that include the smash hit Bally/Midway arcade games, Gorf and Omega Race, Commodore has announced 12 new cartridges to be available this fall.

These new cartridges will make a total of 30 cartridges available by Christmas, and this does not even include Commodore's cassette six-packs or the hundreds of programs available from independents.

Some of the new cartridges go beyond games and actually teach while still being fun. One example is the Home Babysitter, which will help preschoolers learn counting and alphabet skills. Another is Bingo/Speed Math, designed to make math fun. A third is the Visible Solar System, a game-science cartridge that provides a tour of the solar system while it teaches. And a personal finance cartridge will help organize and arrange your home finances.

Leading off the new game cartridges is Cosmic Cruncher. This game combines the popular space theme with a maze game. Players maneuver through the Milky Way to try to crunch all the pulsars in the galaxy. It features 11 levels of play and over 300 color/maze combinations.

Also included among the new game cartridges are two more Bally/Midway arcade sensations, Clowns and Sea Wolf. Clowns is a true arcade classic, with a circus theme. In Sea Wolf, the player is a submarine commander whose mission is to destroy enemy ships.

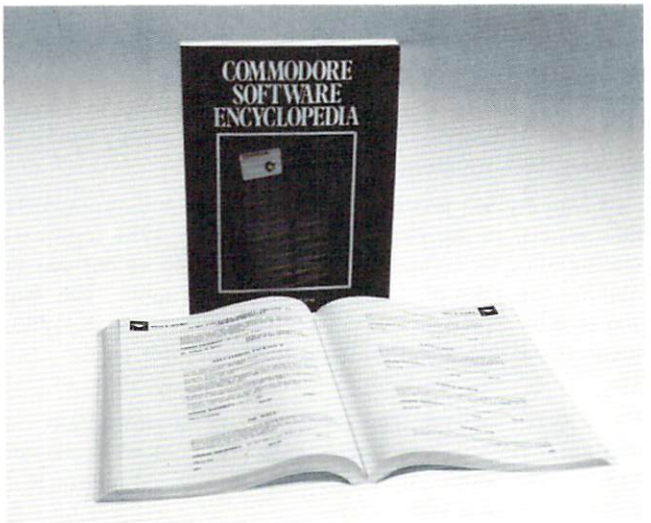
Other new games include Super Smash, Money Wars, Menagerie and Cosmic Jailbreak. All these new games feature Commodore's exciting graphic action that will challenge even the best arcade game players. C=

Software Developers and The Commodore Software Encyclopedia

by
Dave Moyssiadas

Many people may not realize it, but there are thousands of programs for Commodore computers. Hundreds (perhaps thousands) of people are independently writing software for the Commodore line. A few of these software developers ask us to market their products for them, but a great many more have written applications they have kept to themselves, not realizing there are hundreds of people out there begging for just that program to run their house or conduct a physics experiment or simulate a chemical analysis.

Most of the people who have written software are hidden away in colleges and universities, or have developed a program they think is only for them. Many hobbyists have written great programs that do unique things around the home that other computer owners would love to buy. Few of these people have any desire to make money by selling their programs, and most don't even consider the possibility



that what they wrote for their own use to solve a specific problem may have widespread appeal.

We are constantly getting requests for specialized applications that apparently no one ever thought of doing before, but, nevertheless, for which there is substantial need. These take in the whole gamut, from business applications to something as simple as a histogram program to show the fluctuations in your electric bill. Some of these programs are quite complex, and not everyone has the ability or time to develop them. Some are so simple—like a program for a bar graph—that no one takes the time to sit down and knock it out, yet many people would use it a good deal if they had one.

Free Advertising

What is all this leading up to? Just that if you have developed a program, no matter how specialized it may seem to you, why not think about selling it in the marketplace? How do you go about doing that? You advertise. You don't have the inclination to drop a few hundred dollars on ads? Not necessary! There is a place you can advertise directly to Commodore computer owners for FREE. It is the *Commodore Software Encyclopedia*.

There is no charge for listing product descriptions in this publication. The only requirement is that you send us a sample of your product. If it runs, it will be listed in the next edition on a permanent basis (as long as you keep the product available). The people who own Commodore computers range from the casual hobbyist to professional people to universities to industrial people like Lockheed, Rockwell

and NASA. That's quite a potential sales field. Think about it.

But what is the *Commodore Software Encyclopedia*, anyhow? It is a 400 page softcover book listing over a thousand products. Products cover software, hardware, firmware, educational courseware, publications and all sorts of applications. It carries a description of the product, the price, who sells the product, where they are located and their phone number.

You will find a hardware overview describing most Commodore products and the differences among the various models. There is a handy guide to using the book effectively, and also—of all things—an index. The book continues to grow with each edition, as more and more programmers and ordinary folk learn to take advantage of this free advertising. Who said there ain't no such thing as a free lunch?

Why does Commodore offer this "free lunch"? The truth is that the more software available for our computers, the more people are likely to buy a Commodore product. But it's a two-way street. The more Commodore computers that are out there, the more customers you software people will have to sell to. Get the picture? We've got the computers out there; now it's your turn. The more you put out there, the more will be out there for you to put out to (or something like that!). So much for you casual programmers.

The Pros

There are also professional developers who seek out markets for their products. These people have, generally, expensive and powerful programs of exceptional quality. There is a place for their products, also, in the *Encyclopedia*. Commodore is first and foremost a hardware manufacturer, which is why we make the best microcomputer available. We do some software development ourselves, but are always looking for outstanding products our end users would find useful. If you are a professional developer, the *Encyclopedia* is an obvious method for reaching your potential customers.

Approved Products

Obviously, we can't fill everyone's needs with a dozen or so packages, so we also endorse a larger number of equally high quality products to cover a more varied marketplace. These are called Commodore Approved Products. They are products Commodore has tested extensively, field researched, and found to be exceptional. Approved Products must meet or exceed the same stringent requirements that Commodore's own products meet.

Again, if you have a high quality product, you are welcome to bid for an endorsement by Commodore. Among the advantages are that you are allowed to use Commodore's name and the Approved Products logo in all your advertising, your products are given special consideration in the *Software Encyclopedia*, and the products hold more credibility in the marketplace.

So, if you have a product you wouldn't mind making some money on, bring it to our attention. Write to the Approved Products Manager at Commodore, 487 Devon Park Drive, Wayne, PA 19087, for details on submitting software for listing in the *Encyclopedia*, acquiring Approved status, or marketing your product through Commodore. C=

SuperPET Update

I would like to sincerely 'Thank You' for the positive responses to the SuperPET Update section in the August/September issue. There are many questions that need to be responded to concerning the SuperPET. I will try to clear up some misinformation printed in past issues of the Commodore Magazine, then answer this month's most frequently asked questions.

Clearing up misinformation:

Q. How can the screen be cleared from within the program on the SuperPET?

A. *Printing the chr$(12) character in 'Waterloo MicroBASIC' will clear the screen and home the cursor.*

Q. What pins are required to connect a modem to the SuperPET?

A. *The SuperPET has pins 1,2,3,4, 5,6,7,8, and 20 for use. The SuperPET can be used to interface with serial devices other than modems. However, depending upon which particular device is being used, as few as three (1,2,and 3) or a combination up to the nine pins may be needed. The pin configuration will be dictated by the peripheral being connected to the SuperPET. For connecting to a modem, a direct connect 25 pin cable is recommended.*

Walt Kutz

Most asked questions

Q. How can I get a listing of my program on the printer?

A. *To get a listing of a program (APL was covered in the August/September issue) do the following: depress the 'shift key' and '5 key' to put the language editor in command mode, then for;*

(1) BASIC — save 'printer'.
(2) COBOL — p'printer'.
(3) FORTRAN — p'printer'.
(4) PASCAL — p'printer'.

If you are using the 8300P and getting reversed case printing, check the ADA-1450 and 'close' switch #1, this should correct the problem.

Q. How can I use the printer within my program to list data?

A. *Each language interpreter accesses the printer in a different way. Rather than give program listings to show how the printer is accessed, I will indicate modifications that can be made to the 'tutorials' to use the printer.*

(1) BASIC — Page 53, Example-29. Change line 40, " 'namefile' " to " 'printer' ". To get a clean run when executing this program DELETE lines 140-220. The printer will list names entered via the console.

(2) COBOL — Page 64-68, Example-18. This program shows exactly how the printer is used to "Print a report on the printer".

(3) FORTRAN — Page 61, Example-39. Change line 5, (be sure to count the blank line) " "namefile" " to " 'printer' ". Again for a clean run eliminate the lines of code from the second 'open. . .' statement through the 'close. . .' statement.

(4) PASCAL — Page 7, Example 5. This program requires some additional defining in the VAR section. Between the "var" and "begin" lines enter "p:text; ". After the first "begin" line, enter " rewrite (p,'printer'); ". You might want to refer to page 93 for additional information regarding "var" and "rewrite" usage within a program. The "p" becomes the first parameter within parenthese of the "writeln" statements: " writeln(p,'A table of squares and cubes:'); " and " writeln(p,x, xsquared, xcubed); ".

On the move with SuperPET

The SuperPET is being used at Commodore Corporate by the National Credit Manager, Gene Weichman, to do credit checks. He is tying into the Dun & Bradstreet credit services division data base. This is done optionally through either Tymenet or Telenet. Using the SuperPET has reduced reporting costs by nearly 30%, not to mention that the reports are available immediately as opposed to having to wait for their delivery. The SuperPET was demonstrated to, and has become a recommended 'terminal interface' by the Dun & Bradstreet Credit Services Division.

I have had the good fortune of working with the Philadelphia branch of I. P. Sharp, helping to interface the SuperPET to their data base. It has been a coordinated effort, Commodore supplying the hardware, Keysoft International Limited providing the interfacing software, and I. P. Sharp Associates, Inc., the technical expertise. Sharp provides worldwide timesharing "public online data bases" that contain historical, numeric data in time series form. The data bases are concentrated in five subjects: aviation, economics, energy, finance, and actuarial. Future articles will include more information on these data bases as well as the electronic mail service. Sharp's comment, "It is successful, not only as a computer, but also as a terminal."

One of the more pressing areas concerning the SuperPET is the area of communications, especially with respect to interfacing with mainframes. In the next issue I will be covering available communications software for the SuperPET. For now there is a 'public domain' program, sent to us from the folk in Canada (always a good source of information and help) that you can have by sending me a disk (specify 4040 or 8050 format) and self addressed (please include return postage) envelope.

The next issue will also contain a very interesting article by Tama Traberman, Social Studies teacher and Middle Schools Social Studies Department Chairperson. It is about how the SuperPET is being used to teach social studies concepts to eighth grade middle school students using, would you believe, APL. In later articles she will be discussing APL as a tool for teaching biology and physics. C=

— *Walt Kutz*
SuperPET Product Manager, U.S.

WORDCRAFT 20 ...A First! Word Processing for VIC 20®

Never be at a loss for words again! Expressing yourself is easier, faster, and less expensive than ever before . . . thanks to Wordcraft 20© from United Microware. Wordcraft 20© has all the features found on bigger systems costing a lot more — all the features you'll need to prepare "picture-perfect" documents, letters, personalized mailers, lists, and many other projects that have you juggling words.

Wordcraft 20© is easy to use; it won't confuse you. Your words will appear on the screen just as you've written them. This fully featured system lets you do what you need to do . . . change a character, a word, or an entire block of copy . . . arrange and rearrange the text . . . set the tabs and margins . . . adjust the line lengths . . . move the copy up and down, left or right, so you see every word on the screen . . . before you print. Wordcraft 20© gives you personal word processing with professional results. Plug Wordcraft 20© into your VIC® today . . . see for yourself just what words will do. Available exclusively from your local UMI dealer.

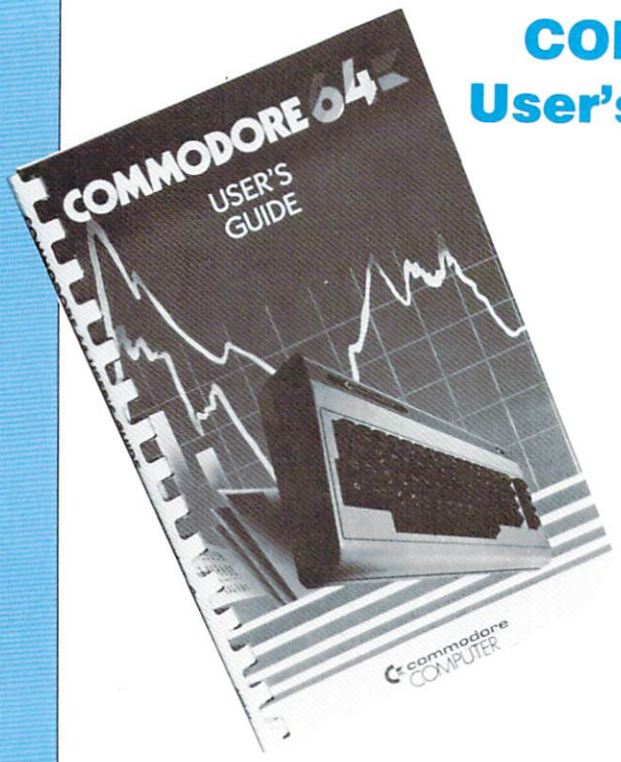
umi

United Microware Industries, Inc.
3503-C Temple Avenue
Pomona, CA 91768
(714) 594-1351

Commodore and VIC 20 are registered trademarks of Commodore Business Machines, Inc. Wordcraft 20© copyright held by P.L. Dowson.

Circle #9 on the Reader Service Card

COMMODORE 64
User's Guide Update



Those of you who read both COMMODORE and POWER/PLAY may be perturbed with us for publishing this article in both magazines. But we're caught in a terrible dilemma. The 64 is an incredibly versatile machine, lending itself to both business-educational-scientific use AND home use. Although we can generally sort out which kinds of information go to which audience (a new game for the 64 obviously goes into POWER/PLAY, while a review of an electronic spreadsheet goes into COMMODORE, for instance), we sometimes have to get the same information to BOTH audiences. This is just such a case. Please forgive us, loyal fans, but that seems to be the nature of this computer.

Introduction

Commodore is constantly trying to bring you the most efficient and reliable computer in the world today. Along with the hardware improvements that come from practical applications of the Commodore 64 in the marketplace, the documentation should also reflect any changes and/or improvements that occur. This is the most up-to-date information available for your Commodore 64. The changes listed here should be used to replace the comparable information in your User's Guide. Future updates will normally be available through the Commodore User Magazines (COMMODORE and POWER/PLAY) as well as the COMMODORE INFORMATION NETWORK on CompuServe.

The format of this update is as follows:

A 1. Page and Paragraph or Section of the Commodore 64 User's Guide.
2. Old information.
3. New information.

EXAMPLE:

A 1. P.2, SIDE PANEL CONNECTIONS, 3. Game Ports
2. Each game connector can accept a joystick, game controller, or lightpen.
3. Each game connector can accept a joystick or game controller paddle, while the lightpen can only be plugged into the game port closest to the front of your computer.

The following list is in numerical order by page.

A 1. P.vii, INTRODUCTION, paragraph 3
2. . . . the SPRITE EDITOR lets you animate as many as 8 different picture levels at one time.
3. . . . at one time. The SPRITE EDITOR will soon be available as a software program that you can load directly into your Commodore 64.

B 1. P.vii, INTRODUCTION, paragraph 4
2. . . . a programmable ADSR . . . generator, an envelope generator, . . . filters for each voice
3. . . . a programmable ADSR . . . envelope generator and a programmable high, low, and bandpass filter for the voices

C 1. P.2, SIDE PANEL CONNECTIONS, 3. Game Ports
2. Each game connector can accept a joystick, game controller, or lightpen.
3. Each game connector can accept a joystick or game controller paddle, while the lightpen can only be plugged into the game port closest to the front of your computer.

D 1. P.3, figure 1
2. Control port at (3)
3. CONTROL PORT 1 CONTROL PORT 2 at (3)

E 1. P.3, figure 2
2. (5) CHANNEL SELECTOR
3. above (5) <−Ch.3 Ch.4−>

F 1. P.3, CONNECTIONS TO YOUR TV, 1.
2. . . . push it in. The cable will only go in one way.
3. . . . push it in. Either end of the cable can be used.

G 1. P.4, 6.
2. . . . channel selector switch (channel 3 or 4)
3. . . . channel selector switch (channel 3 move the switch to the left, channel 4 move the switch to the right)

H 1. P.8, USING THE COMMODORE 64, 1.
2. . . . the rocker switch on the left-side panel
3. . . . the rocker switch on the right-side panel when you're looking at the computer from the front.

I 1. P.10, TROUBLESHOOTING CHART
2. Picture with excess background noise
3. Sound with excess background noise

J 1. P.10, CURSOR
2. The flashing square next to READY . . .
3. The flashing square under READY . . .

K 1. P.14, KEYBOARD, paragraph 4
2. . . . the graphic character on the right side of the key.
3. . . . the graphic character on the right hand side of the front part of the key.

L
1. P.14, KEYBOARD, last sentence
2. . . . special function keys . . . marked on the upper part of the key.
3. . . . special YELLOW function keys . . . marked on the front of the key.

M
1. P.18, TIP, paragraph 1
2. . . . normal display. Simultaneously depress:
3. . . . normal display. First press the RUN/STOP key and then press the RESTORE key. RUN/
   STOP must always be held down in order to use the RESTORE key function.

N
1. P.18, TIP, Last paragraph
2. . . . type, SYS64738 and press RETURN.
3. . . . type, SYS64759 and press RETURN.

O
1. P.18, 1. CARTRIDGES
2. . . . REMOVING CARTRIDGES OR YOU WILL DESTROY THE CARTRIDGE!
3. . . . REMOVING CARTRIDGES OR YOU MAY DAMAGE THE CARTRIDGE AND/OR YOUR COMMODORE 64!

P
1. P.19, 3. DISK, sentence 4
2. . . . protective gate by pushing in on the lever.
3. . . . protective gate by pushing down on the lever.

Q
1. P.20, paragraph 1
2. . . . blank, turning the background color of the screen . . .
3. . . . blank, turning the border color of the screen . . .

R
1. P.20, LOADING PROGRAMS FROM DISK
2. After LOAD "PROGRAM NAME,8"
3. The 8 is the code for the disk, so you're just letting the computer know that you want the program loaded from the disk.

S
1. P.29. Paragraph 2
2. You must always separate the parts of a mixed print statement with some punctuation for it to work properly.
3. You can separate the parts of a mixed print statement with punctuation for various formats.

T
1. P.34, EDITING TIPS, last paragraph
2. Try editing our sample . . .
   10 PRINT "COMMODORE";
3. Try editing our sample program on page 33 by changing line 10 and adding a comma to the end of the line. Then RUN the program again.
   10 PRINT "COMMODORE",

U
1. P.45, First paragraph
2. Pretty neat!
3. Pretty neat! To stop the program hold down RESTORE and hit RUN/STOP.

V
1. P.45, Last line
2. PROMPT MUST BE LESS THAN 40 CHARACTERS.
3. PROMPT MUST BE 38 CHARACTERS OR LESS.

W
1. P.46, Program
2. 20 IF A$ = "" THEN 10
   30 IF A$ = "F" THEN 100
   40 IF A$ = "C" THEN 50
3. 20 IF A$ = "" THEN 20
   30 IF A$ = "F" THEN 100
   40 IF A$ <> "C" THEN 10

X
1. P.48, Program Line 20
2. 20 GET A$: IF A$ = "" THEN 20
3. 20 GET A$: IF A$ = "" THEN 20 ⟵No space between quotes

Y
1. P.49, Sample program
2. IF YOU . . . APPEAR AS 1 ROW
3. IF YOU . . . APPEAR AS 1 COLUMN

Z
1. P.50, General formula for random numbers
2. NUMBER=INT(UPPER LIMIT*RND(1))+LOWER LIMIT
3. NUMBER=INT(LOWER LIMIT+(UPPER−LOWER+1)*RND(1))

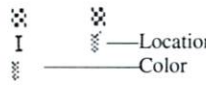
AA
1. P.51, Sample program
2. 60 IF GU = NM THEN PRINT "GREAT! YOU GOT MY NUMBER"
3. 60 PRINT "GREAT! YOU GOT MY NUMBER"

BB
1. P.51, Sample program
2. 100 IF AN$ <> "N" THEN 80
3. 100 IF AN$ <> "N" THEN 70

CC
1. P.57, KEYBOARD COLOR DISPLAY Chart
2. CTRL 1 BLACK +1
3. CTRL 1 BLACK +1

DD
1. P.58, COLOR CHR$ CODES, screen display
2. 10 PRINT CHR$(30);"CHR$(30) CHANGES ME TO?"
3. 20 PRINT CHR$(30);"CHR$(30) CHANGES ME TO?"

EE
1. P.64, Top of the page
2. POKE 1524,81
3. POKE 1524,81
   POKE 55796,1



FF
1. P.69, SPRITE CREATION, paragraph 5
2. Sprites are displayed in a special high-resolution mode which turns the screen into a 320 dot wide by 200 dot high area.
3. Sprites are displayed in a special independent 320 dot wide by 200 dot high area. However, you can use your Sprite with any mode, highresolution, low-resolution, text, etc.

GG
1. P.71, Line 230 in program
2. 230 DATA 1,62,64,0,15,128,0,156,128,0,73,0,0,73,0
3. 230 DATA 1,62,64,0,156,128,0,156,128,0,73,0,0,73,0

HH
1. P.72, first sentence
2. . . . you need to know from which 64 block section each series of 8 memory blocks will get sprites in their data . . .
3. . . . you need to know from which 64 byte section (1 whole sprite) each series of 8 memory blocks (1 row of sprite making data) will give each sprite its data . . .

II
1. P.72, Step 1
2. . . . location 21 which turns on the sprite.
3. . . . location 21 a 1 for the bit which turns on the sprite.

JJ
1. P.76, Paragraph 2
2. Now start to move in the usual X direction register for sprite 2 (which is in location 4 of the map) starting from 1 again.
3. Now start from 0 again in the usual X direction register for sprite 2 (which is in location 4 of the map).

KK
1. P.76, Additional notes on sprites, additional paragraph
2. . . .
3. To define multiple sprites, you may need additional blocks for the sprite data. You can use some of BASIC's RAM by moving BASIC. Before typing or loading your program type:
   POKE44,16:POKE16*256,0:NEW
   Now, you can use blocks 32 through 41 (locations 2048 through 4095) to store sprite data.

Continued on page 96.

Everything You Always Wanted To Know About Commodore Computers *

*And Asked!

Commodore 64

1. Q: How do you find out the amount of BYTES FREE on the Commodore 64?

A: The following formula computes the number of bytes available to BASIC between the end of the array storage and the start of strings.

```
FR=FRE(0):IF FR<0 THEN
FR=FR+65536:PRINT FR
```

2. Q: How is the RUN/STOP key disabled on the Commodore 64?

A: Disable— POKE788,52
Enable— POKE788,49

3. Q: How can the Commodore 64 be used with the VICMODEM?

A: By eliminating line 400 of the program in the VICMODEM manual this can be achieved.

4. Q: Will VIC 20 cartridges work with the Commodore 64 or the Max Machine?

A: VIC 20 cartridges will not run on the Commodore 64. The Max Machine cartridges will run on the Commodore 64, however.

5. Q: What is LOGO?

A: LOGO is an instructive graphic language that will be usable on the Commodore 64 with the CP/M option.

VIC 20

1. Q: What telecommunications-networks are accessible with the VIC 20?

A: The VICMODEM allows the VIC 20 to access such networks as The Source, CompuServe, Micronet, Dow Jones, and the New York Times, to name a few.

2. Q: How many digits does the VIC 20 floating point have?

A: The floating point variable routines in the VIC have nine significant digits for the mantissa, and the exponent the range of −38 to +37.

3. Q: What is the difference in the voltage levels of RS232 that the VIC 20 uses and the standard RS232?

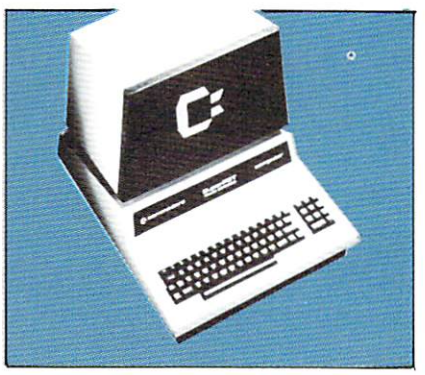
A: The VIC 20 voltages are at TTL level (0 to 5 volts), while the RS232 standard is −12 to +12 volts. In addition, the signal levels from the VIC are inverted from the standard RS232.

4. Q: How is the RUN/STOP key disabled on the VIC 20?

A: Disable— POKE788,194
Enable— POKE788,191

5. Q: What does EMAIL stand for when accessing a telecommunications network?

A: EMAIL stands for Electronic Mail. This is a means of sending and receiving messages through telecommunication lines.



SuperPET

1. Q. How do you access the COBOL interpreter on the new Update disk?

A. The menu displayed on the screen does not show a selection for COBOL. Replacement ROM's to change the screen menu will not be available through Commodore. The Cobol interpreter is loaded by typing, using the shift key, COBOL.

2. Q. Are there any SuperPET User Groups?

A. Yes, one such group is:
The SuperPET Gazette
c/o Richard W. Barnes, Editor
P.O. Box 411
Hatteras, NC

Another is:
SuperPET User Group
4782 Boston Post Rd.

Pelham, NY 10803
Paul Skipski

3. Q. Is there a networking system for the SuperPET?

A. There is, at this time, no true networking system for the SuperPET. Until one becomes available, the MUPET and DOUBLE MUPET from CMD should satisfy most users.

4. Q. What is the cost of the 'ROM Selection Modification Kit'?

A. The 'ROM' Kit (P/N 9000029) is FREE. The dealer may charge for installation but the kit is FREE. Dealers will receive a credit of $25.00 from Commodore upon receipt of the 'Bingo Card'.

5. Q. What baud rates can be used with the SuperPET?

A. The SuperPET is capable of transmitting from 50 to 9600 baud. Refer to the SuperPET overview Manual, pages 60 and 61 for 'Setup' options.

CBM

1. Q. What is the difference between ULSL ASCII code and Commodore PET ASCII?

A: U.S. ASCII code uses a seven bit code which can represent 128 characters (2=128). PET ASCII has an eight bit code which can represent 256 characters (2=256); this accounts for the additional characters on Commodore computers. This is why an ASCII converter is necessary to interface with third party printers.

2. Q: Can a CBM computer run a VIC 20 Program?

A: Our CBM line of computers will run a 3K expanded VIC 20 program with no difficulties. To run a program from an unexpanded VIC 20 the following steps are necessary:

POKE41,16 (moves BASIC pointer to start of BASIC on CBM)
POKE4096,0 (initializes the new BASIC area with zero)

To run an 8K or 16K program the following steps are necessary:

POKE41,18
POKE4608,0
CLR

Your VIC 20 programs will now run on the CBM computer!

3. Q: Are spaces necessary in the BASIC text of a line?

A: The Commodore computer will disregard unnecessary spaces in a program. Spaces are mainly used for clarity and easy reading of programs.

4. Q: How can one program be loaded from another program?

A: The first program must incorporate a load statement in the last executed statement of the program.

Disk Drives

1. Q: How can an 8050 diskette be downgraded to a 4040 diskette?

A: The following procedure lists the proper steps: 1) Turn on the computer and the 4040 drive only; 2) Run the "Change 4040" program (included on the Test/Demo diskette); 3) Turn on the 8050 drive; 4) Run the "Unit to Unit" program (found on the Test/Demo diskette). The Unit to Unit program will copy all of your files from the 8050 diskette to the 4040 diskette.

2. Q: Will the 8250 Disk Drive accept 8050 Disk Drive diskettes and vice versa?

A: The 8050 diskettes are read/write compatible when used on the 8250 Disk Drive. When using an 8250 diskette in the 8050 Disk Drive only the underside of the 8250 diskette will be read/write compatible.

3. Q: What is the BAM?

A: BAM stands for Block Availability Map. This is a disk memory representation of available and allocated space on disk. It is referenced by the DOS (Disk Operating System) to determine what space is available and how many blocks can be allocated.

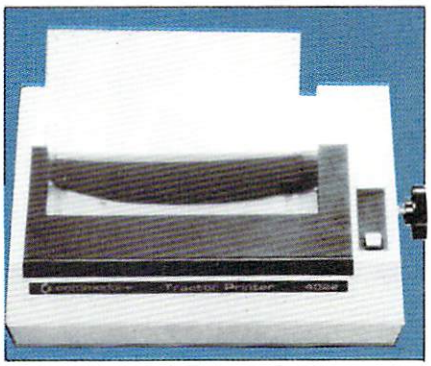
4. Q: In disk operating commands, why does a "d" precede certain ▶

commands?

A: Machines using BASIC 4.0 need this preceding the command.

5. Q: What is a Hard Disk?

A: A Hard disk is one that has a rigid platter on which the magnetic media iron oxide is coated.



Printers

1. Q: What Commodore printer supports the A.P.L. character set?

A: The Commodore 8300P printer supports the A.P.L. character set.

2. Q: What are the DIP switch settings on the PET Printer Interface when Interfacing an 8300P printer?

A: Switches 1, 2, 3 are off and switch 4 is on.

3. Q: What print wheels can be used with the 83000P printer?

A: Over 150 print wheels are listed in the Diablo book #90044-01.

4. Q: Can a tractor feed mechanism be obtained for the 8300P printer?

A: Yes, this device can be obtained through any authorized Commodore dealer.

5. Q: How can a program be listed in upper/lower case on the 8023 printer?

A: Set the printer to upper/lower case by issuing this sequence of commands:

```
OPEN 1,4,7
PRINT #1
CLOSE1
```

The printer now prints in upper/lower case.

Some BASIC Answers for the 64

by Mike Richter

One of the first questions to come to mind about the Commodore 64 is: will it outdate my old PET/CBM software? The answer is a conditional "no." The condition is that those programs were in BASIC and that you can make a few minor changes where necessary. Many programs will run as they are; most will need only a few lines changed; a few may be more trouble than they're worth. Let's go through them one at a time to see what's changed—and why.

Cassette Input

Your old datassette programs will use the 64's tape interface just as the PET did, with an exception in the display only. Since the screen is relocatable in the 64, memory must be accessed for it periodically. When that happens, the tape interface could lose data. So, the 64 blanks the display while the tape is running. Everything else functions just as the first generations did. One other difference is that when the 64 finds the header, it pauses for 30 seconds before loading a program; you can eliminate that delay by hitting the Commodore key when the display appears.

Literals mode

In PET and CBM, 59468 served a number of functions, including switching between literals and graphics modes. To get lower case, we POKEd 59468,14; for graphics, POKE 59468,12. On the 64, we still switch the 2's bit, but now on 53272. The simple way to do it is with POKE 53272,23 for literals, POKE 53272,21 for graphics (the early manuals have the wrong values). It would be smarter to allow for the other uses of 53272 by using

POKE53272,PEEK(53272)OR2 for literals,
POKE53272,PEEK(53272)AND253 for graphics.

Bytes Free

There's a lot more memory available on the 64 than on a 32K PET or CBM; in BASIC, you have 38911 bytes at the start, and even adding the DOS doesn't use any of it. Unfortunately, the logic for FRE(0) translates the answer as an integer using integer logic; the largest number it knows is 32767. If you have a small program (less than 7K), FRE(0) gives a negative answer. How much memory is available? 64K plus FRE(0). A typical way to get it if you need it is

X=FRE(0):IFX<0THENX=X+65536.

Screen access

The screen in the 64 is laid out exactly as it was in the PET, but in a different place. Where the older machines started at 32768 (32K), the 64 begins at 1024 (1K). Change the addressing, and your old program can safely PEEK and POKE the screen.

There are two other changes that usually won't matter. One is, simply, that there is only one cassette port on the 64. If you have a two-cassette program, well, it won't convert. Second, the longest input prompt that the 64 will take is 38 characters (the manual is wrong). However, very few programs need that many; if yours does, just convert "input . . .'>38 characters'"to "print'. . .>38 characters':input"

The only other changes are those for disk operations, arising from the 64's use of BASIC 2.0 instead of 4.0. They were covered in a previous issue. If you are not using BASIC 4.0, your programs should convert in a few minutes—if they need conversion at all.

One example of the interoperability of software among Commodore computers is seen in our new line of Personal Software. By writing entirely in BASIC 2.0, and using just one extra BASIC line, the programs run in 64, 2001-16, 2001-32, 4032, 8032, and 8096. The line is

print"SNssqqqqqqqqqqqqqqqqqqqqqqq"spc(39)"Oe

where everything in quotes is in reverse video. It isn't as confusing as it looks. First, we clear the screen, set the graphics mode (to compress the screen), cancel any preexisting window, and mark the top left corner of a 4032, 8032, or 8096. Then we use cursor controls to move to the bottom right corner and set it for the 80-column machines. Finally, we set the print color to white for the 64. All that's left to do is to POKE the case we want; doing it for *both* 59468 and 53272 is safe on all machines.

Commodore has done a remarkable job on the 64, if we do say so ourselves. We've added sprites and sound and color; upgraded everything and dropped the price. And with all that, no more than 100 bytes are needed to run your tried and true programs in the new machines. C=

Telecommunications: PRESENT AND FUTURE

Jeff Hand

Originally, this article was intended to discuss the amazing potentials for telecommunications in the future. However, many of the ideas are happening now! So, I figured we'd touch on some interesting applications that are happening today and explore some of the ways you can use your home or business computer and a modem, now!

Telecommunications, along with the rapid growth and power of personal computers like the VIC 20 and Commodore 64, is quickly creating a worldwide computer community. Telecommunications networks are great and powerful tools, (they can even be a lot of fun!). They are being used successfully around the world in such diverse disciplines as medicine, business, education, and communications. Hobbyists and the lay person are also getting into networking, evidenced by the rapid growth of the information networks; i.e., CompuServe, Dow Jones News/Retrieval and the Source.

Technology is advancing so rapidly that computer development and telecommunications are "growing up" side by side. Ten years ago there were no personal computers in the U.S. By '87 it's predicted that one in five homes will have a home computer and all of them will be able to communicate over standard phone lines. A recent International Data Corp. (IDC) study found that more than eleven billion dollars have already been invested in networks, and 108,000 interlinked data terminals are in use. By '85 these figures are expected to double, with computer networks growing at 20% to 40% a year for the next ten years.

Fundamental changes in the way computers are used began around 1968 with the U.S. Department of Defense's ARPANET (Advanced Research Projects Agency Network) project. This project allowed computers around the country to communicate with each other, work at the same time on the same problems and to tap into each other's memories for possible answers to difficult questions. From these initial attempts at networking, the International Standards Organization has defined seven levels of communication protocols for networks. Each level has a set of rules and definitions for communicating. In each level a separate type of information or level of organization is implemented. The lower levels define electrical and mechanical signaling standards.

The British, Canadian, and French governments are actively supporting the development of home and business computer networks. The systems they have set up are used to disperse weather, news, home banking, agricultural, financial and other information services. The Canadians and British have developed their own national system of protocols for telecommunications; Telidon and Prestel respectively. The French phone system has demonstrated commitment to computer telecommunications by ordering 300,000 terminals to give to phone users. The terminals will be used to replace the phone book.

How would you and your VIC 20 like to have access to a network of captive satellites, microwave and private wire transmission systems, as well as electronic mail distribution centers, in hundreds of thousands of locations around the world? Just to show you how sophisticated networks are becoming, you can use your VIC 20 and VICMODEM to send a message from your living room to anywhere in the world.

Two sophisticated electronic communication networks, Access[SM] and EasyLink[SM], have been developed by Western Union. They eliminate the time and cost constraints of the traditional means of documentation preparation, handling, and delivery. In addition, Access and Easylink afford the user a variety of important benefits whether the need is to communicate with just one or one hundred thousand people.

For instance, this new system allows the sender to decide when the communication can be delivered:

1) Now — by Telex or TWX
2) Today — by telegram or cablegram
3) Tomorrow — by Mailgram®
4) 2 days — E-COM (US Postal Service's version of electronic mail)
5) 3 days — Computer Letter

The best part of the Access and EasyLink system is that they require no special equipment by the sender or receiver, beyond the VIC 20 and the VICMODEM. There is also a small fee for the initial hookup. If you're interested give your local Western Union representative a call.

Another way for Commodore Computer owners to keep up-to-date is through the use of "bulletin boards." An "electronic bulletin board" is similar to the cork bulletin board at work or school. However, electronic bulletin boards are more flexible and can disseminate the information you want in a simple, effective manner. For example, on a normal bulletin board you'd have to sort through scraps of papers and advertisements to find something of interest to you. With brief commands, the electronic bulletin board allows you to scan messages, read only the newest messages, read messages on a specific topic, or read only the messages directed specifically to you. In many cases electronic bulletin boards allow users to download free public domain software. Here is a listing of privately operated Commodore bulletin boards in the U.S.:

Board Name: Commodore Chicago
Phone: (312) 397-0871
System Operator: Keith Peterson
Time: 24 hrs./7 days
Address: 2246 North Palmer Drive
Schaumburg, IL 60195

Board Name: Kansas City Pet User Group
Phone: (816) 356-2382
System Operator: Rick West
Address: 5415 Blue Ridge Boulevard
Kansas City, MO 64133

Board Name: Commodore Communications
Phone: (314) 625-4576
System Operator: Tony Ott
Address: 633 Bent Oak Drive
Lake St. Louis, MO 63367

Board Name: South East Wisconsin Pet User Group (SEWPUG)
Phone: (414) 554-9520
System Operator: Tim Tremmel
Address: 3614 Sovereign Drive
Racine, WI 53406

Board Name: South East Wyoming Computer Users
Phone: (307) 637-6045
System Operator: Roger Kelsar
Address: 1104 Logan Avenue
Cheyenne, WY 82001

And for Canadian users:

Name: PSI-Wordpro BBS
Phone: (416) 624-5431
System Operator: Steve Punter
Address: Mississauga, Ontario

Name: Thunder Bay BBS
Phone: (807) 345-7199
Address: Thunder Bay, Ontario

Name: NORTEC
Phone: (416) 782-7320
Address: 147 Roe Avenue
Toronto, Ontario

Name: Toronto PET User Group (TPUG)
Phone: (416) 223-2325
Address: 5529 Yonge Street
Willowdale, Ontario

If there is anyone out there who for some extenuating circumstance would like to look at some other bulletin boards besides Commodore's, a complete listing can be found on The Source telecommunications network, by accessing PAMS on the system. There is even a bulletin board to reserve room for experiments on NASA's space shuttle: (301) 344-9156.

To tie into any of these bulletin boards you will need a modem that operates at 300 baud (some work at an optional 1200 baud) and in the originate mode. The other terminal settings are: full duplex, one stop bit, 8 bit word, no parity. The software must also translate all signals into ASCII code. These are also the default settings for VICTERM-1 and the settings for the CompuServe network.

If you're interested in starting your own bulletin board, it's not a job for those needing a lot of free time. And you should be a true computer fanatic. The system operators I've talked to are a hardy, dedicated breed of computer enthusiast, and make no money for all their efforts. (It's the hardest job you'll ever love; it's an adventure.) Stop in and say howdy to them sometime (Tony Ott of St. Louis was particularly helpful directing me to most of the information in this part of the article).

You can order the Bulletin Board software package from Commodore Dealers in Canada, or you can contact Commodore Canada, 3370 Pharmacy Avenue, Agincourt, Ontario, Canada M1W 2K4. The cost is about $230 Canadian, which is slightly cheaper in American dollars. The price includes the BBS software, documentation, and a never-ending supply of updates from the author, Steve Punter (of WordPro fame).

Schematics are also included for building a simple automatic-answer device for the 8010 modem and telephone, that will cost about $35 to build. You will also need to purchase a special 'user port connector' that will allow the BBS program to monitor the automatic-answering device so that any disconnection by a user during operation can be detected.

Unfortunately, at this time the software will only work with the 8010 or Livermore IEEE modem and a CBM or upgraded PET computer. Commodore no longer markets the 8010 modem, but ECX Computer Company (2678 North Main St., Walnut Creek, CA 94596) will sell you one for about $350 (U.S.). I've heard that two gents are working on a version of the BBS software that will operate with a Hayes Smartmodem. When I'm sure there is a finished product I'll let you know about it.

I saved the major (well, not really, it's more the level of a lieutenant but who's ranking?) Commodore bulletin board till last: **The Commodore Information Network** on CompuServe. The Commodore Information Network can be thought of as a paperless magazine. But the Commodore Information Network is better because it has capabilities that can't be found in ordinary magazines. A paper magazine does not have daily update capabilities, or the search capabilities of a main frame computer, or the immediacy of conferencing. And, most important, an ordinary magazine can't give you personalized answers to your Commodore questions within two days. Eat your heart out Power/Play and Commodore Magazines.

Commodore's Network can be separated into five distinct parts: videotex, bulletin board, HOTLINE, database, and conferencing. The videotex area was described in the last Commodore Magazine (July/August). Don't tell me you forgot already?

The bulletin board section has been very active since the beginning of the Network in June. The bulletin board is primarily for user-to-user interaction and discussion. I read the bulletin board daily. If there are any questions or problems I think I can answer, I'll throw my two cents worth in.

The HOTLINE can be accessed by entering GO CBM-200. On the HOTLINE users can direct questions to Commodore customer support. To save yourself and us some aggravation write the question out as clearly and concisely as possible. Despite this minor problem our success rate has been very high for answering questions within a day or two.

The database portion of Commodore's Information Network is just starting to get off the ground. Plans are underway to implement the following sections of information:

0. HOTLINE/Psst — This section will contain questions on Commodore equipment. You'll be able to use the keyword search capabilities of the computer to find the answer to your question . . . before you even ask. How's that for service! Psst stands for Problem Solvers Support Team. (Don't laugh at me, I didn't think of it.) Each week there will be a new write up on some interesting topic such as: Using the 1540 disk drive with the Commodore 64, backward scrolling on the VIC, and more.

1. Manual Updates — This will be a listing of revisions to manuals. This is the quickest way we could think of getting the information to you.

2. Vendors — A listing of third party vendors that provide you with Commodore related equipment, information and peripherals that would be of interest to you.

3. Software — Commodore has several thousand public domain programs. We will begin distributing this software to the public through this section of our Network, for FREE!

4. Bulletin Boards — This section contains much of the information that is contained in this article.

5. User Group Newsletters — Computer groups will inform you of their meeting dates, important annoncements, and newsletters.

6. Chip Specs — Outline of specifications for some of Commodore's more popular semiconductors.

The conferencing capability of the Network can be used to hold nationwide user group meetings. Eventually we'll have lectures and guest speakers on the Network where questions can be fielded from across the country.

To access all of these fun things on CompuServe and the bulletin boards you must have terminal software, so I guess this is a good place to fill you in on the status of: 64-TERM, VICTERM-40, and Executive Software for the PET and CBM. The 64-TERM software is identical to VICTERM-1 tape software except it has been modified to work on the 64. By the time you read this article 64-TERM will be available at your dealers. For those of you who bought the VIC-MODEM for the 64 and didn't receive 64-TERM, Commodore is providing your dealer with the appropriate software so that you can make an exchange.

CompuServe is in the process of creating Executive Software for the PET/CBM computers. This doesn't mean that only executives can use this software but rather it's the software that includes all of the protocols of operation on the CompuServe system. You'll have uploading/downloading, transmission error correction, cursor positioning, graphics, and color capabilities on CompuServe. The folks at CompuServe have targeted this software for late December and it can be purchased through the Softex section of CompuServe.

The VICTERM-40 cartridge will provide a 40-column screen, and downloading capabilities to disk, printer, or disk drive. I know some folks are so excited to get a hold of the VICTERM-40 cartridge that they are jumping up and down in anticipation. But, I'm afraid those people are going to have to jump a little while longer.

A major decision was made that in the long run will save users money and aggravation. We're upgrading the VICTERM-40 cartridge so that it is the only terminal program you'll need for the VIC. Our original plan was to have three levels of terminal software sophistication: VICTERM-1, VICTERM-40, and the Executive Terminal package. We have opted to combine VICTERM-40 and the Executive Software package, resulting in a savings to you the user.

The new VICTERM-40 will still have the 40-column screen, and downloading capabilities, but in addition we will add another menu for all file transfers, including uploading, and CompuServe's protocols for: transmission error checking, cursor positioning, color and semi-graphics mode. Unfortunately, this extra capabilitiy requires additional time. We are making every effort to get this software to you quickly.

When you get on to CompuServe stop in and check out Commodore's Information Network (G CBM1), and say hi. I'm always glad to see new people making use of the system. If you have any ideas or suggestions feel free to leave a message on the bulletin board (CBM310) or HOTLINE (CBM200).

We know our readers love to get the inside story on the latest developments at Commodore before people on the outside find out. But we "official" company spokespeople usually aren't willing to put ourselves at risk by talking about what's going on behind the scenes. What a dilemma.

Imagine our delight, then, when about a month ago we noticed a little fellow with very large ears scuttling among the offices, standing in dark corners, lurking behind partitions. When we finally collared him he admitted his name was Captain Scuttlebutt and his occupation was "professional rumorist."

"You're just the one we want," we said, and put him to work immediately writing this column. We cannot be held accountable for Captain Scuttlebutt's scribblings, but you may enjoy them.

Captain Scuttlebutt's Unbelievable Rumors

Attention teachers! Wouldn't you love to be able to sit at your computer, tune in on what each of your students is doing at his or her terminal, and interact with each of them individually without ever leaving your seat or distracting other students? ("OK, Mary, let's wake up," you might type onto Mary's screen at station six. "You've made the same mistake five times now." "I don't understand these problems," Mary might answer on your screen from her own keyboard.)

Well, your pal Scuttlebutt heard Commodore is working on just such a classroom network that allows computer interaction between a teacher terminal and individual student terminals. In fact he saw one in action! Right now the idea is to set it up for both PETs and Commodore 64s.

OK, so you've heard of classroom networks before—at costs of $400-700 per student station. That's what you call prohibitive for most schools. So you'll be glad to find out that Commodore's network will cost only about the price of a game cartridge—or so they say. (Please don't ask who "they" are. It's top secret.)

If you'll suspend your disbelief for a moment, we'll give you the whole story, as we understand it from our usual reliable sources. (No names, please.)

Evidently, the hardware consists of two cartridges per terminal and a lot of 6-strand wire. (The big debate is whether or not educators will have the savvy to get the right kind of wire when they set up the network.) One cartridge contains the software and the other is the interface—basically two buffers and a dipswitch.

The only limitation, if you can call it that, is that, when these two cartridges are in place, you can use only disk-based, unprotected software for instruction. Don't panic. "Unprotected," in this case, means requiring no external hardware (i.e., dongle) for protection. That means you still can use just about every existing education program, including those we've just put together in Ontario (see page 30).

Now, if you'd like to get really outrageous, consider this. What if the teacher hooked the control computer to a modem, dialed up CompuServe, and everybody in the class could participate in the exciting world of telecommunications? Don't say we told you so, but that may be yet another possibility for this project.

Think of it. Thirty-two happy faces at thirty-two happy Commodore 64s (or PETs). The teacher loads the day's lesson into everybody's computer with one simple command. As the lesson proceeds, the teacher is continually scanning each screen. Ooops, there's that Mary, again, asleep at the keyboard. Wake up, kiddo.

So much for that rumor. On to other juicy tidbits. Guess what someone saw passing through the office in Wayne, reputedly on its way to the publisher? Could it be the *Commodore 64 Programmer's Reference Guide*? All 300 pages of it? Don't quiver in fear—it's supposed to be very friendly and eminently readable. We heard whispers about a late October delivery date, but keep that under your hat. We don't want to give Commodore a reputation for announcing things prematurely. But we'll bet you can look forward to a review in the December issue of this magazine.

OK, let's get it all out. Now that the "Introduction to Basic, Part II" package is available for people who want to teach themselves programming on the VIC 20, could you entertain the possiblity of a similar package for the 64? Around here the walls are thin, and rumors do have a way of sometimes turning out to be true. Might it be called the Computer Tutor?? And while we're on the subject of the 64, could you use a word processor supposedly more powerful than WordPro? We won't say to look for it near the end of the year or anything like that, but . . .

We'll keep you tuned in to the latest hearsay as it occurs. Meanwhile, as far as we're concerned, you never heard any of this. C=

USER CLUBS
Sound Off!

We're continuing to compile a list of all Commodore Users clubs throughout the country. If you'd like to add your name to the rolls, please send your club's name, address, and other pertinent information to:

Commodore Users Clubs
c/o Editor
Commodore Magazine
487 Devon Park Drive
Wayne, PA 19087

And remember, once our list is comprehensive enough, we will begin forwarding valuable information to clubs on a regular basis, including hardware and software updates, technical bulletins, new product announcements, and troubleshooting tips.

ALABAMA

Huntsville PET Users Club
9002 Berclair Road
Huntsville, AL 35802
Contact: Hal Carey
Meetings: every 2nd Thursday

ARIZONA

VIC Users Group
1206 N. Fraser Drive
Mesa, AZ 85203
Contact: Paul V. Muffuletto

ARKANSAS

Commodore/PET Users Club
Conway Middle School
Davis Street
Conway, AR 72032
Contact: Geneva Bowlin

CALIFORNIA

Lawrence Hall of Science
UC Berkeley
Computer Project,Room 254
Berkeley, CA 94720
(415) 642-3598

PALS (PETS Around Livermore Society)
886 South K
Livermore, CA 94550
Contact: John Rambo

SCPUG Southern California PET Users Group
c/o Data Equipment Supply Corp.
8315 Firestone Blvd.
Downey, CA 90241
(213) 923-9361
Meetings: First Tuesday of each month

California VIC Users Group "VIC-VILLE"
c/o Data Equipment Supply Corp.
8315 Firestone Blvd.
Downey, CA 90241
(213) 923-9361
Meetings: Second Tues. of each month

Commodore Users Club
1041 Foxenwoods Drive
Santa Maria, CA 93455
(805) 937-4106
Contact: Greg Johnson

Valley Computer Club
2006 Magnolia Blvd.
Burbank, CA
(213) 849-4094
1st Wed. 6 p.m.

Valley Computer Club
1913 Booth Road
Ceres, CA 95307

PUG of Silicon Valley
22355 Rancho Ventura Road
Cupertino, CA 95014

BAMBUG
1450 53rd Street
Emeryville, CA
(415) 523-7396

North Orange County Computer Club
3030 Topaz, Apt. A
Fullerton, CA 92361
Dave Smith

Lincoln Computer Club
750 E. Yosemite
Manteca, CA 95336
John Fung, Advisor

PET on the Air
525 Crestlake Drive
San Francisco, CA 94132
Max J. Babin, secretary

PALS (PETs Around Livermore Society)
886 South K
Livermore, CA 94550
(415) 449-1084
Every third Wednesday
7:30 p.m.
Contact: J. Johnson

SPHINX
314 10th Avenue
Oakland, CA
(415) 451-6364
Every 2nd & 4th Thurs.

San Diego PUG
c/o D. Costarakis
3562 Union Street
(714) 235-7626
7 a.m.-4 p.m.

Walnut Creek PET Users Club
1815 Ygnacio Valley Road
Walnut Creek, CA 94596

Jurupa Wizards
4526 Kingsbury Pl.
Riverside, CA 92503
Contact: Walter J. Scott

PET/CBM/VIC User Club
c/o General Computer Store
22323 Sherman Way #9
Canoga Park, CA 91303
Contact: Tom Lynch

Vincent Yanniello's VIC 20 Software Exchange Club
2130 Colby Avenue
West Los Angeles, CA 90025
(213) 479-3000

The Commodore Connection
2301 Mission St.
Santa Cruz, CA 95060
408-425-8054
Bud Massey

COLORADO

VICKIMPET Users' Group
4 Waring Lane,
Greenwood Village
Littleton, CO 80121
Contact: Louis Roehrs

CONNECTICUT

John F. Garbarino
Skiff Lane Masons Island
Mystic, CT 06355
(203) 536-9789

Commodore User Club
Wethersfield High School
411 Wolcott Hill Road
Wethersfield, CT 06109
Contact: Daniel G. Spaneas

VIC Users Club
c/o Edward Barszczewski
22 Tunxis Road
West Hartford, CT 06107

New London County Commodore Club
Doolittle Road
Preston, CT 06360
Contact: Dr. Walter Doolittle

FLORIDA

Jacksonville Area PET Society
401 Monument Road, #177
Jacksonville, FL 32211

Richard Prestien
6278 SW 14th Street
Miami, FL 33144

South Florida PET Users Group
Dave Young
7170 S.W. 11th
West Hollywood, FL 33023
(305) 987-6982

VIC Users Club
c/o Ray Thigpen
4071 Edgewater Drive
Orlando, FL 32804

ILLINOIS

Shelly Wernikoff
2731 N. Milwaukee Avenue
Chicago, IL 60647

VIC 20/64 Users Support Group
c/o David R. Tarvin
114 S. Clark Street
Pana, IL 62557
(217) 562-4568

Central Illinois PET User Group
635 Maple
Mt. Zion, IL 62549
(217) 864-5320
Contact: Jim Oldfield

ASM/TED User Group
200 S. Century
Rantoul, IL 61866
(217) 893-4577
Contact: Brant Anderson

PET VIC Club (PVC)
40 S. Lincoln
Mundelein, IL 60060
Contact: Paul Schmidt, president

Rockford Area PET User's Group
1608 Benton Street
Rockford, IL 61107

Commodore Users Club
1707 East Main St.
Olney, IL 62450
Contact: David E. Lawless

VIC Chicago Club
3822 N. Bell Ave.
Chicago, IL 60618
John L. Rosengarten

INDIANA

PET Users
Jerry Brinson
PO Box 36014
Indianapolis, IN 46236
(317) 898-3604

GHS Computer Club
c/o Grangeville High School
910 S. D St.
Grangeville, ID 83530
Contact: Don Kissinger

Cardinal Sales
6225 Coffman Road
Indianapolis, IN 46268
(317) 298-9650
Contact: Carol Wheeler

CHUG (Commodore Hardware Users Group)
12104 Meadow Lane
Oaklandon, IN 46236
Contact: Ted Powell

VIC Indy Club
P.O. Box 11543
Indianapolis, IN 46201
(317) 898-8023
Ken Ralston

IOWA

PET Users Group
c/o Don Vorhies
1321 42 St. SE
Cedar Rapids, IA 52403

Commodore User Group
114 8th St.
Ames, IA 50010

KANSAS

Wichita Area PET Users Group
2231 Bullinger
Wichita, Kansas 67204
(316) 838-0518
Contact: Mel Zandler

Kansas Commodore Computer Club
101 S. Burch
Olathe, KS 66061
Contact: Paul B. Howard

VIC 20 Users Club
739 Litchfield
Wichita, KS 67203
(316) 262-4861
Contact: Walter Lounsbery

LOUISIANA

Franklin Parish Computer Club
#3 Fair Ave.
Winnsboro, LA 71295
James D. Mays, Sr.

NOVA
917 Gordon St.
New Orleans, LA 70117
(504) 948-7643
Kenneth McGruder, Sr.

MARYLAND

Assoc. of Personal Computer Users
5014 Rodman Road
Bethesda, MD 20016

Blue TUSK
700 East Joppa Road
Baltimore, MD 21204
Contact: Jim Hauff

House of Commodore
8835 Satyr Hill Road
Baltimore, MD 21234
Contact: Ernest J. Fischer

MASSACHUSETTS

Eastern Massachusetts VIC Users Group
c/o Frank Ordway
7 Flagg Road
Marlboro, MA 02173

VIC Users Group
c/o Ilene Hoffman-Sholar
193 Garden St.
Needham, MA 02192

Commodore Users Club
Stoughton High School
Stoughton, MA 02072
Contact: Mike Lennon

Berkshire PET Lovers
CBM Users Group
Taconic High
Pittsfield, MA 01201

The Boston Computer Society
Three Center Plaza
Boston, MA 02108
(617) 367-8080
Mary E. McCann

MICHIGAN

David Liem
14361 Warwick Street
Detroit, MI 48223

VIC Users Club
University of Michigan
School of Public Health
Ann Arbor, MI 48109
Contact: John Gannon

Toledo PETS
734 Donna Drive
Temperance, MI 48182
Contact: Gerald Carter, president

Commodore User Club
32303 Columbus Drive
Warren, MI 48093
Contact: Robert Steinbrecher

Commodore Users Group
c/o Family Computer
3947 W. 12 Mile Rd.
Berkley, MI 48072

VIC Commodore Club
2765 Bristol Rd.
Dowling, MI 49050
Bill Kelley

MINNESOTA

MUPET (Minnesota Users of PET)
P.O. Box 179
Annandale, MN 55302
c/o Jon T. Minerich

Twin Cities Commodore Computer Club
6623 Ives Lane
Maple Grove, MN 55369
(612) 424-2425
Contact: Rollie Schmidt

MISSOURI

Clearwater Club
Clearwater School
Star Route
Piedmont, MO 63957
Contact: Carolyn Polk

KCPUG
5214 Blue Ridge Boulevard
Kansas City, MO 64133
Contact: Rick West
(816) 356-2382

PET SET Club of St. Louis
1501 Carman Road
Manchester, MO 63011
(314) 527-3236
Contact: Ed or Til Keil

MONTANA

Powder River Computer Club
Powder River County High School
Broadus, MT 59317
Contact: Jim Sampson

Commodore User Club
1109 West Broadway
Butte, MT 59701
Contact: Mike McCarthy

NEVADA

Las Vegas PET Users
4884 Iron Avenue
Las Vegas, NV 89110

NEW JERSEY

Amateur Computer Group
18 Alpine Drive
Wayne, NJ 07470

Somerset Users Club
49 Marcy Street
Somerset, NJ 08873
Contact: Robert Holzer

Educators Advisory
P.O. Box 186
Medford, NJ 08055
(609) 953-1200
John Handfield

NEW HAMPSHIRE

Northern New England Computer Society
PO Box 69
Berlin, NH 03570

NEW YORK

Capital District PET Users
Ben Green
Albany Area, NY
(518) 370-1820

Long Island PET Society
Ralph Bressler
Harborfields HS
Taylor Avenue
Greenlawn, NY 11740

PET User Club of Westchester
Box 1280
White Plains, NY 10602
Contact: Ben Meyer

LIVE (Long Island VIC Enthusiasts)
17 Picadilly Road
Great Neck, NY 11023
Contact: Arnold Friedman

Commodore Masters
25 Croton Ave.
Staten Island, NY 10301
Contact: Stephen Farkouh

VIC Users Club
76 Radford St.
Staten Island, NY 10314
Contact: Michael Frantz

VIC Users Club
c/o Christopher Kwasnicki
44 Harvey Ave.
Staten Island, NY 10314

PUG of Rockland County
c/o Ross Garber
14 Hillside Court
Suffern, NY 10901
(914) 354-7439

West Chester County VIC Users Group
P.O. Box 146
Pelham, NY 10552
Joe Brown

SPUG
4782 Boston Post Rd.
Pelham, NY 10803
Paul Skipski

VIC 20 User Club
151-28 22nd Ave.
Whitestone, NY 11357
Jean F. Coppola

VIC 20 User Club
339 Park Ave.
Babylon, NY 11702
(516)669-9126
Gary Overman

NORTH CAROLINA

Amateur Radio PET User's Group
PO Box 30694
Raleigh, NC 27622
Contact: Hank Roth

Commodore Users Club
4241 Castleton Road
Charlotte, NC 28211
Contact: Ed Harris

VIC Users Club
c/o David C. Fortenberry
Route 3, Box 351
Lincolnton, NC 28092

Microcomputer Users Club
Box 17142 Bethabara Sta.
Winston-Salem, NC 27116
Joel D. Brown

OHIO

Dayton Area PET User Group
933 Livingston Drive
Xenia, OH 45385
B. Worby, president
(513) 848-2065
J. Watson, secretary
(513) 372-2052

Central Ohio PET Users Group
107 S. Westmoor Avenue
Columbus, Ohio 43204
(614) 274-6451
Contact: Philip H. Lynch

OKLAHOMA

Southwest Oklahoma Computer Club
4311 Floyd Ave. NW
Lawton, OK 73505
Joe Teeples

OREGON

NW PET Users Group
John F. Jones
2134 N.E. 45th Avenue
Portland, OR 97213

PENNSYLVANIA

PET User Group
Gene Beals
PO Box 371
Montgomeryville, PA 18936

Penn Conference Computer Club
c/o Penn Conference of SDA
720 Museum Road
Reading, PA 19611
Contact: Dan R. Knepp

PACS PET Users Group
20th & Olney Streets
Philadelphia, PA

Glen Schwartz
807 Avon
Philadelphia, PA 19116

Gene Planchak
4820 Anne Lane
Sharpsville, PA 15150
(412) 962-9682

PPG (Pittsburgh PET Group)
c/o Joel A. Casar, DMD
2015 Garrick Drive
Pittsburgh, PA 15235
(412) 371-2882

Bucks County Commodore User Club
72 E. Rambler Dr.
Holland, PA 18966
(215) 322-0394
Jim Dubrouillet

PUERTO RICO

CUG of Puerto Rico
1607 Calle Colon
Santurce, PR 00911
Ken Burch

SOUTH DAKOTA

PET User Group
515 South Duff
Mitchell, SD 57301
(605) 996-8277
Contact: Jim Dallas

TENNESSEE

River City Computer Hobbyists
Memphis, TN
1st Mon. at Main Library

Nashville VIC Users Group
American Computer Store
1004 8th Ave., S.
Nashville, TN 37203
(615) 242-8592
Jane Maggard
1st Thurs. of month, 6 pm

TEXAS

SCOPE
1020 Summit Circle
Carrolton, TX 75006

PET Users
2001 Bryan Tower
Suite 3800
Dallas, TX 75201

Larry Williams
PO Box 652
San Antonio, TX 78293

PET User Group
John Bowen
Texas A & M Microcomputer Club
Texas A & M, TX

CHUG (Commodore Houston Users Group)
8738 Wildforest
Houston, TX 77088
(713) 999-3650
Contact: John Walker

UTAH

Utah PUG
Jack Fleck
2236 Washington Blvd.
Ogden, UT 84401

The Commodore User's Club
742 Taylor Avenue
Ogden, Utah 84404
Contact: Todd Woods
Kap, president;
David J. Shreeve, vice president

The VIClic
799 Ponderosa Drive
Sandy, UT 84070
Contact: Steve Graham

VIRGINIA

Northern VA PET Users
Bob Karpen
2045 Eakins Court
Reston, VA 22091
(803) 860-9116

VIC Users Group
Rt. 2, Box 180
Lynchburg, VA 24501
Contact: Dick Rossignol

VIC Users Group
c/o Donnie L. Thompson
1502 Harvard Rd.
Richmond, VA 23226

Dale City Commodore User Group
14752 Danville Rd.
Dale City, VA 22193
(703) 680-2270
James Hogler

WASHINGTON

NW PET Users Group
2565 Dexter N. 3203
Seattle, WA 98109
Contact: Richard Ball

PET Users Group
c/o Kenneth Tong
1800 Taylor Ave. N102
Seattle, WA 98102

VIC 20 Computer Club
947 N. Burroughs Ave.
Oak Harbor, WA 98277
Michael D. Clark

WISCONSIN

Sewpus
c/o Theodore J. Polozynski
PO Box 21851
Milwaukee, WI 53221

Waukesha Area Commodore User Group (WACUG)
256½ W. Broadway
Waukesha, WI 53186
Contact: Walter Sadler
(414) 547-9391

Commodore User Group
1130 Elm Grove St.
Elm Grove, WI 53122
Tony Hunter

CANADA

Toronto PET Users Group
381 Lawrence Ave. West
Toronto, Ontario, Canada
M5M 1B9
(416) 782-9252
Contact: Chris Bennett

PET Users Club
c/o Daniel Cayer
R.R. 6
Simcoe, Ontario
Canada N3Y 4K5

Vancouver PET Users Group
Box 91164
West Vancouver, British Columbia
Canada V7V 3N6

CCCC (Canadian Commodore Computer Club)
c/o Strictly Commodore
47 Coachwood Place
Calgary, Alberta, Canada
T3H 1E1
Contact: Roger Olanson

KOREA

Commodore Users Club
K.P.O. Box 1437
Seoul, Korea
Contact: S. K. Cha

MEXICO

Asociacion De Usuarios Commodore
c/o Alejandro Lopez Arechiga
Holbein 174-6° Piso
Mexico 18, D.F.

Club de Usarios Commodore
Sigma del Norte
Mol del Valle, Local 44
Garza Garcia, N.L. 66220

NORWAY

VIC Club of Norway
Nedre Bankegt 10,
1750 Halden
Norway

COMMODORE NEWS

Commodore User Group Flourishes in the Land of Lincoln

by John O'Brien

As promised last issue, here is the second of our User Group profiles. In this issue we are featuring the Central Illinois PET User Group (CIPUG). The most striking thing about this group is the incredible amount of written material on Commodore that they are responsible for. This is mainly due to the fact that one of the founders is Jim Strasma. Strasma, along with his wife Ellen, have co-written the third edition of the PET/CBM users guide. Strasma and co-founder Jim Oldfield started the club and the club newsletter, the Midnite Software Gazette. The group is also selling *The Whole PET Catalogue*, that is a "best of" the first two years of the Midnite Software Gazette, combined with information from the Toronto PET User Group.

CIPUG began as two people exchanging information. Both Oldfield and Strasma bought their PETs back in 1978, and according to Oldfield "I think we're one of the older user groups. We started out just trying to get people together and start a newsletter. We found that there was a void—nobody was doing software reviews."

The first issue was a four page handout. Oldfield said, "It's just about doubled in size every issue since."

Oldfield describes the publication this way: "We are very opinionated and we print mostly software reviews and some gossip."

The Gazette (the name may be changed soon) was well received, but as it grew problems arose. Strasma and others including his wife found more and more of their time being spent producing the Gazette.

"We just couldn't give it away anymore. My wife, Ellen, threatened to quit. We had to start paying some of the people who were working so hard on it. Plus the cost of producing got so high. We also found out that Jim Bressler's The Paper, one of the oldest of PET resources, was no longer going to be published."

The result is that The Gazette will now be a paid subscription magazine incorporated with The Paper. Anyone subscribing to The Paper will have their subscription finished off with the Gazette.

The group now meets the third Friday of every month at Computer Country in Springfield, Illinois. The group discusses general topics in the first half of the meeting but then splits into different interest groups, such as VIC, or machine language.

"Right now most of the users are in the Springfield area," Strasma said. "Eventually we will have different sub-groups meeting in different towns."

Strasma is also the founder of ASM/TED, the worldwide group of assembly language programmers. The ASM/TED software library also acts as CIPUG's library. The librarian is Brent Anderson and, as Oldfield said, "Boy, is he busy."

Strasma said, "Brent Anderson takes a lot of trouble to organize and catalogue the library. It may seem as though the library is a little smaller than some, but that's because Anderson takes the time to weed out duplicates and uses only the best available program. We also have the Help disk that will go along with my book and is sort of a 'best of' public domain software."

For more information on CIPUG and its publications, contact Jim Oldfield, 635 Maple, Mt. Zion, IL 62549. Subscriptions for the Gazette are $20 for six issues.

User Bulletin Board

User Groups Forming

Florida
VIC 20 User Group now forming.
Contact Robert D. Campbell
c/o Newberry Fire Station
P.O. Box 747
Newberry, FL 32669
904-472-2137

Illinois
VIC club now forming in Chicago.
Contact John Leon at 312-267-4935
between 9 and 5, Monday-Friday.

GLITCH FIX

"Two-handed Sketching"
by Preston Marshall
April/May COMMODORE,
pages 70-72

A few typos in the program listing may have stymied your efforts to run this program. Please note the following corrections:

Line 21530 on page 71 should read:
REM SEARCH ARRAY FOR VALUE OF PK
Delete line 61405, immediately following the above.
In its place add:
21540 FOR I=0 TO 15
Halfway down the page, change line 35060 to read:
IF PEEK (KB)=0 THEN GO TO 35020

Thanks to Preston Marshall for these corrections. Apologies to those who were frustrated by the misprints.

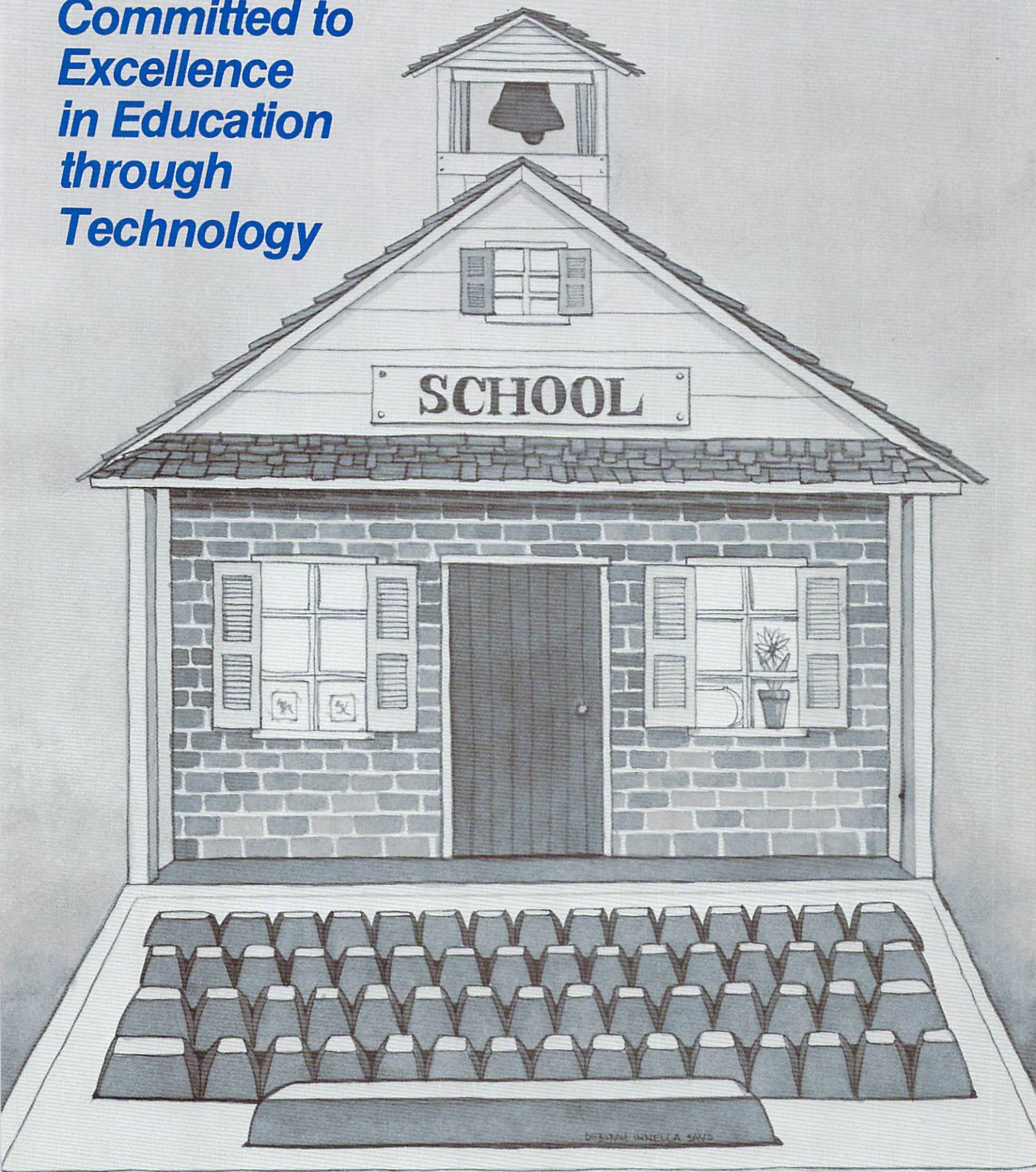
"Mailbox"
August/September
COMMODORE, page 6

At the top of the second column, the sys locations reference should read "Load $d8", NOT "Load$dB". Thanks to Joe Rotello of Tucson for that one.

"Conversion Tutor"
August/September
COMMODORE, pages 67-68

Please correct line 570 to read as follows:
570 If LEFT $(A$,1)="Y" THEN RUN

Commodore: Committed to Excellence in Education through Technology



Ontario Schools Create Phenomenal Educational Software Package

Imagine. Six hundred-forty six (that's 646) genuinely workable educational programs on 50 disks, neatly packaged in two large volumes, clearly labeled and accompanied by a catalogue that lists programs by subject category, grade level and quality. Programs developed by teachers working in the classroom. Programs created to really teach, not just show off some flashy programming techniques. All of them in the public domain.

This may sound like some misty-eyed educator's hopelessly optimistic daydream. But, thanks to unprecedented cooperation among Commodore companies, the Ontario Department of Education and the government of Ontario, about 100 kids helped make this "daydream" a reality last summer.

The kids, from thirty school districts in the province of Ontario, were selected by their teachers to spend the summer revamping a veritable ocean of public domain programs, mainly for the PET, that had been collected in the Ontario schools, the Toronto PET Users Group, several colleges, and Commodore companies in Europe and North America.

The young programmers, aged 16-18, were paid mainly through a student summer work grant from Ontario 's provincial government, along with a grant from the Ontario Department of Education. In addition, Commodore's Canadian office hired ten programmers and a full-time supervisor to oversee the project. The students worked every day from 8:30 to 4:30 standardizing the raw programs, then sent them to Commodore in Toronto, where the programs received a final cleanup, and were also converted to run on the Commodore 64.

Then, late in August—the frantic final week before school began—ten students ran forty dual disk drives eight hours a day to copy about 800 disks for distribution to the school boards in September. Astonishingly, they made the deadline.

One of the ten Commodore programmers works on conversions.

You may suspect it wasn't quite as easy as it sounds. Gaining a concensus among thirty school boards in a province the size of Texas is a phenomenon in itself, whatever the issue may be. You're right. It wasn't easy, but the enthusiasm of the participants carried the project through to completion.

The project actually began in 1981, when several educators affiliated with three large school boards in Ontario received government funding to hire a number of students for four-week periods during the summer. The intention was to begin cataloguing the programs, but, as these things go sometimes, the group found they first had to lay down some ground rules before they could plunge into work on the programs themselves. So they spent that summer learning where the problems were and putting down a foundation for the next summer's effort.

Meanwhile, Commodore's Toronto office, without knowing this specific project was already underway in the schools, decided it was time to start stepping up development in educational software. It wasn't too long, however, before Commodore's people got wind of what the schools were doing and decided to combine their efforts.

With the cooperation of the school boards, the project was extended to include conversions for the Commodore 64. Some pre-released prototypes were allocated by Commodore for this part of the project. A central clearing house was set up in Toronto, the students were hired, and by June, 1982, the all-out effort was underway.

Everything wasn't a bed of roses, however. The cataloguing committee, for instance, set a July cutoff date for completed programs, because they needed that much time to convert and catalogue the 600-odd programs that had been submitted in order to have them finished by Sep-

tember. However, the programmers in the schools went on working right through August, so by the time the school boards met, they received a total of 1031 fully catalogued, converted programs. Of these, the Commodore programmers had had time to convert 646 to run on the 64. (These 646 are the programs now available in the public domain, worldwide.)

Or, take the week of frantic disk copying, when the students ran out of disks in mid-week. And then ran out of labels the next day. As Commodore's Frank Winter, who became involved in the project early in 1982, put it, "It's the logistics that kill you."

"Nevertheless, this is a project that could be carried out by anyone anywhere in the world," Winter went on. "It's a great example of how cooperation among government, school boards and private industry can get a project finished that no one member could have accomplished individually."

Not only that, but, as Winter pointed out, Commodore and the schools are ready to do it again next summer — for another 600 programs.

As a result of the students' and teachers' efforts, each of the thirty school boards involved in the project received their complete set of disks at

Finally, the disks are inserted into plastic jackets and assembled in binders.

a special presentation on September 13. According to Winter, more than one board member expressed surprise that the project had actually been completed.

"It's easy to start out on a project of this size," Winter commented, "but it's a lot harder to keep the ball rolling, especially in the summer, when many people go on holiday."

In addition to the disks received by the school boards, every authorized Commodore dealer in Canada also received a set of the 646 programs that had been converted to run on the 64. Each of these sets contains a total of 52 disks packaged in two volumes and a printed catalogue. Fifty disks contain the actual programs, and two contain demo programs, including character and sprite editors for the 64.

These 646 programs cover most subject categories, and also include administrative programs for grading, attendance and statistical analysis, a number of games and several utilities. They will run on any Commodore computer, including the 8032, PET (thin and fat 40) and Commodore 64, and will be available in the public domain all over the world.

Frank Winter (right) presents the first copies of the new public domain software to Ed Kellow, president of Commodore, Canada.

For those educators interested in exactly what subjects are covered in these programs, a brief breakdown: 24 business, 21 computer science, 108 English, 13 Francais, 10 history, 203 math, 101 science, 17 technology and 18 miscellaneous (includes things like "Man and Society", physical education and music).

Commodore's Public Domain Educational Policy

For some time Commodore companies around the world have been encouraging the development and exchange of public domain educational programs with schools. An example of this in the U.S. has been the Educational Marketing Resource Book, which contains 185 educational programs. The 646 programs referred to in this article will be released here in the U.S. and to the public domain throughout the world in conjunction with Commodore dealers, user clubs and Educational Resource Centers. Watch for details in future issues.

Ten students hired by Commodore ran 40 dual disk drives to copy 8,000 disks in one week.

Kids Star at Connecticut Computer Schools

Futureschool's philosophy is that people who can't control a computer are at the mercy of those who can

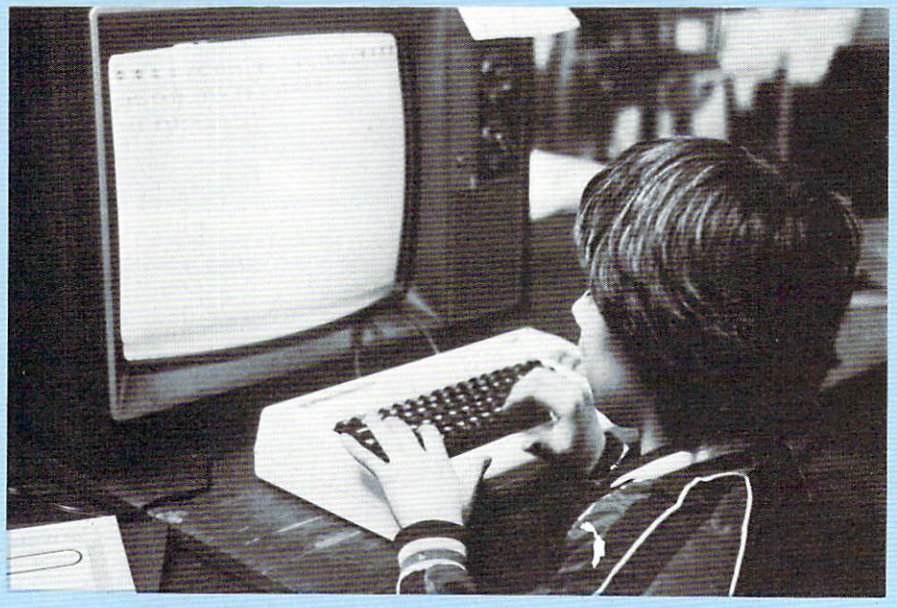
Fewer than half of the 15,000 school systems in the United States now offer computer training, and in the elementary grades most of those teach only "computer literacy" — the ability to operate a personal computer — rather than programming. Yet, according to the people at Futureschool, based in Madison, Connecticut, it is quickly becoming necessary for even second- and third-graders to learn how to program.

To fill that need, Futureschool has been setting up "itinerant" computer schools for children and teenagers throughout Connecticut and Rhode Island, using VIC 20 computers as the main teaching aid. According to Louis Esposito, Futureschool's president, the VIC's price/performance ratio made it the logical choice for use at the Futureschool sites.

"We couldn't find another combination of ingredients to beat it," Esposito said of the school's decision. "In other words, we couldn't do better for the money."

Futureschool's approach to computer education is unique for several reasons. First, it has no classrooms of its own. In fact, Futureschool is actually an educational package, rather than a physical school — a package the founders hope to eventually market all over the country as a carefully controlled franchise.

Right now the school is operating out of 15 locations, many of them YMCAs. When they get the use of a location, the school moves in its specially designed physical package — lightweight walls and a large central column, around which computer stations are arranged (more or less "in the round"). The idea, according to Esposito, is to create a very specific kind of environment, appropriate for a class of 12 to 20 students.



Even second graders can learn to program the VIC 20 at Futureschool.

"The secret of providing optimum comfort, involvement and exposure for each student is partly in how we arrange the physical space — how we arrange walls and seating, and how we separate the kids from the instructors. Wherever we get a space, we create a distinct Futureschool environment," he explained.

Futureschool is also unique in its teaching methods. Using what they call the ZuseMethod (developed by Futureschool founder DeWitt Zuse), Futureschool tries to remove the tedium from learning, minimizing early frustrations through encouragement, yet challenging advanced students. As Jim Dore, one of Futureschool's officers, put it, "Kids soon find out that learning computer programming can be as much fun as playing Pac Man or soccer."

Esposito emphasizes the "problem-solving" approach of the Zuse-Method. The computer itself is presented as a project or problem. Then, at each stage, both the instructors and the students generate problems that the students must then solve. As students become more advanced, the problems become more "real". So, although they might start off programming a simple graphic element, they eventually may be designing an inventory control system.

"Kids generally think up problems that have meaning to THEM — so a third-grader might give you a problem that has to do with ducks and cookies, and a teenager might generate a similar problem, but centered around sports," Esposito elaborated. "We can tell by the style and quality of the problems they pose whether or not they really understand what they're doing."

Children are placed in the program

according to their skill level, Esposito explained, and proceed through the course at their own pace. After a four-week general introductory course, the students — some of them as young as eight years old — begin to learn programming on a VIC 20, which they can also take home with them to practice. They progress in four-week "modules," taking as many modules as necessary to reach whatever skill level they wish. The average student takes about six months to get completely through this part of the course, Esposito said, although some finish in a shorter time.

Those students who complete the second part and show a high level of computer understanding may then have the option of continuing as paid assistants, employed by Futureschool as aides in their classrooms.

Futureschool has been in operation only since January of this year, but has already sparked an enthusiastic response from parents, students and schools in the Connecticut-Rhode Island area. Last summer, for instance, Futureschool attracted over 200 students to a summer program at a Connecticut private school. The side-effect, Esposito said, was that many of these students also signed up for other courses, since they were going to be at the school, anyway.

The school's philosophy is that a person who cannot control a computer is at the mercy of anyone who can. As a result, Futureschool sees to it that its students have the learning and problem-solving power of a computer at their fingertips.

Futureschool, P.O. Box 1129, Madison, CT 06443/203-453-2577

FOCUS:

A New Software Exchange for Educators

To help meet the need for low-cost educational software a new software exchange is being developed for—and by—professional educators. For more information about the exchange, contact Mario W. Caprio, Box 180, Stony Brook, NY 11790.

Focus is dedicated to making pedagogically sound educational software available at reasonable prices. If you have written a program that facilitates learning and if you would like to share it with other teachers, or if you would simply like to be on the mailing list, contact Focus. The exchange is interested in programs for all K-12 curriculum topics. This winter a list of programs will be available for teachers. For a modest production fee, copies of programs will be made for any teacher sending a blank tape to Focus. Royalties will be paid for programs accepted for publication. The object is to bring down the cost of effective educational software.

All programs will be written by teachers to run on a forty column, 8K PET. The philosophy of Focus is that learning is satisfying in itself and that making it into an arcade game is ultimately destructive to the learning process. Furthermore, Focus takes the position that truly effective educational software can only be written by educators.

Currently available software includes:

1. **Writing Numbers**
Numbers from one to 999,999,999,999 are written in word form. The student answers in standard numerical form; with commas.

2. **Place Value**
A number is given with one digit highlighted. The student types in the place value of the indicated digit. Ones to hundred billions; decimals to thousandths. Can be used with or without decimals. Selectable level; scores work.

3. **Rounding Off**
Can be run with or without decimals. Provides instructions on errors. Selectable level; scores work.

4. **Number Cruncher**
Play against the computer to guess an integer. Clues are given, score is kept. Selectable range of integers.

5. **Making Change**
In this simulation students use play money to give change. Instructions are given on all errors. After two errors interim assistance is given *during* counting.

6. **Concentration**
In this well-known memory-building card game, students can play against each other, themselves, or the computer. The computer matches its play to student performances.

7. **Metric**
Practice converting from one metric unit to another. Instructions (two explanations) when requested; assistance on errors. Scores work.

COMAL: Programming Language of the Future?

COMAL, devised two years ago by Borge Christensen of the College of Higher Education in Tonder, Denmark, is the official programming language in the schools of Denmark and Ireland. Commodore funded these early efforts and placed the programs in the public domain in Europe.

The new language is now developing a following in the United States, where versions are available for PET and CBM computers. This article provides some background on the new language. It is reprinted from issue 3 of the COMAL Catalyst, *5501 Groveland Terrace, Madison, WI 53716.*

For more detailed information on COMAL, try the COMAL Handbook *by Len Lindsay (editor of the* COMAL Catalyst*), available from Reston Publishing in Reston, Virginia.*

You don't need me to tell you what others have been saying for years. Computers are becoming more advanced every year. After all the advances and innovations in the computer hardware over the past 20 years, why are so many people still using BASIC, which is a rather primitive language compared with some of the new ones available now? Well, until last year I could answer that easily. BASIC came with the computer, it was FREE. It was easy to learn and use. The other better languages may be better for professional programmers, but not for the ordinary non-professional home programmer. They were too rigid, required you to understand complex principles and usually were not interactive.

Languages were paraded around us, each claiming superiority. PILOT claimed to be easier to use (but lacked some of the power). FORTH claimed to be fast, and indeed it is. If you need application software for industrial control, FORTH may be for you. Then APL appeared for the microcomputer. You would see a letter from an APL programmer showing a one line program (with triangles and squiggles) that would do some mathematical calculation that BASIC would need 20 lines to do. But who could read it?

And how about PASCAL and C, both somewhat similar? C is good for people who don't like to type, since it uses brackets a lot to save typing in BEGIN, etc. But it is not too readable. PASCAL is readable but imposes too many rules and regulations on you. EVERY variable must be declared. Besides just declaring it, you must specify what type of variable it is, REAL, INTEGER, etc. And each line must end with a semi-colon (with some exceptions). The program must have the word END. at the end (including the period). For some reason, PASCAL can't figure out where the end of the program is.

PASCAL is also very bad in string handling and Input/Output. Possibly the worst part is that PASCAL has no line numbers and requires the use of an EDITOR that is separate from PASCAL. So you must learn to use the EDITOR as well as learn to use PASCAL.

I tried PILOT, FORTH, and PASCAL. I found that PILOT and FORTH were not suited to my situation. PASCAL would have been nice, but it got me too frustrated with its silly rules.

"If you have been used to great freedom (with BASIC), you will come to regard your PASCAL compiler as a strict and fussy schoolmaster. You still have the chance to be creative, certainly, and you can still have fun, but you must play strictly by the rules." — page 18, *PASCAL FROM BASIC* by Peter Brown.

It has been said that COMAL has the ease of BASIC with the power of PASCAL. The corollary to this is that COMAL doesn't have the problems of BASIC nor the problems of PASCAL. (I use CBM COMAL-80 version 1.02 for reference). If you are considering leaving BASIC behind, read the book *PASCAL FROM BASIC* by Peter Brown. It is very good. It shows how BASIC can be improved with structure, but also points out PASCAL's shortcomings. After reading it you will not want to use either language (BASIC or PASCAL). Then remember that COMAL doesn't have the problems of PASCAL; some of which are indicated by two quotes below from the book:

"Modern education is often held to emphasize flamboyant and trendy subjects at the expense of basic skills. PASCAL suffers in the same way. It is strong on data structures and the like but comparatively weak on the three R's: reading, riting, and rithmetic. Its rithmetic lacks an exponentiation operator, and, an omission felt by commercial programmers, decimal operation. Its reading and riting, i.e. its input and output, seem to be designed to help sell Bill's *BASIC FROM PASCAL* book. It is not so much that PASCAL's input/output is short on facilities; it is just that some fundamental things are difficult or impossible to do."

"You have probably got the impression after reading this chapter, that getting anything in or out of your PASCAL program is like going round the Royal Saint George's golf course equipped only with a putter."

PASCAL FROM BASIC is available from Addison-Wesley Publishing Company for $12.95. Copyright 1982.

So there I was (as many of you may be right now), knowing full well that BASIC was outdated, and not being able to find a suitable replacement. PASCAL is not meant to be a replacement for BASIC for the home computer. But in May of 1981 there was a major breakthrough for the PET/CBM. COMAL arrived on the scene like a ray from heaven. COMAL is an advanced programming language for ▶

the non-professional programmer which includes the advantages of PASCAl without losing the friendliness of BASIC. Anyone who uses the FULL ENHANCED COMAL (CBM COMAL version 1.02) for a week, will not want to go back to BASIC. (Version 1.02 is available on disk for the CBM 8096, or as a plug-in board from Instrutek for any PET/CBM except the original PET with BASIC 1.0 — introductory version 0.12 is available on disk for any PET/CBM with BASIC 4.0. All of this is available through the COMAL USERS GROUP).

You may have heard about COMAL before. You heard that it allows structured programming and even indents the structures automatically so you can SEE them (PASCAL does not provide this 'pretty printing' automatically. You must run a program whose listing is 7 pages long to list your program nicely). You may already know that COMAL allows a multiple line IF . . . THEN . . . ELSE. You may know that it has both a REPEAT . . . UNTIL and a WHILE . . . ENDWHILE loop along with the standard FOR . . . NEXT. You have been told that it has the wonderful CASE structure (replacing the ON . . . GOTO). Then finally you probably have been told about COMAL's multi-line FUNCTIONS and PROCEDURES, both allowing parameter passing and local or global variables. So you already know that COMAL is far superior to BASIC. But many probably still have not used COMAL. Why? Is it fear of change?

Well, fear no more. The COMAL system is on your side, helping you, not fighting you with a bunch of silly rules. COMAL uses line numbers simply for your use in editing the program. The line numbers are not used by the program itself. You can delete blocks of lines with one command (DEL). The system will prompt line numbers for you automatically with the AUTO command. A renumber command (RENUM) is available if you need it. You can list all or part of your program. The listing automatically indents the structures (pretty printing). To LIST a specific procedure, INTRO for example, simply type, LIST INTRO. A program can be listed without the indenting if you wish. Simply add a file name to the LIST command, and the program, or program segment, will be listed to disk or tape. These segments can later be merged into another program via the ENTER command. The disk directory can be printed in full, selectively using pattern matching, or even just listing the SEQ files (or just PRG or USR files).

If you have a printer, it can be turned on or off with the SELECT command. Everything works the same on the printer as on the screen, including TAB and ZONE. So once you have your program output looking nice on the screen, simply add the line SELECT "LP" (for Line Printer) and the program will print the same thing, formatted nicely, just as it was on the screen. (You can't easily do this with BASIC).

COMAL provides you with two different types of sequential files, as well as direct access (random) files. Standard PASCAL has only a sequential file capability. COMAL has the GET$ and KEY$ statements, greatly improved over BASIC's simple GET. COMAL lets you easily 'tack on' your own enhancement set of 'keywords'. Just define them as procedures or functions. Once a program is RUN all procedure and function names are remembered by the system, and can be called from direct mode at any time (the sky is the limit here).

For example, let's say you have defined a function in your program called GCD (greatest common divisor) that had two parameters (the two numbers to test for the greatest common divisor). After you are done running the program, COMAL still remembers that function. You can now find the greatest common divisor of any two numbers in direct mode. For example: PRINT GCD(35,21) — COMAL will reply with 7.

COMAL has PRINT USING, allowing the formatting of numbers into neat columns, as well as PRINT AT, allowing you to specify the row and column to begin printing on the screen. It reads DATA statements with a READ statement (Did you know that standard PASCAL does not have this capability!). It also has the RESTORE statement like BASIC, but with a major improvement. You can restore the next DATA pointer to any line you wish. But the line is not specified by its line number (remember, line numbers are not significant to a running COMAL program). Instead you include a label on the line before the DATA line to be restored to, and use RESTORE NAME (using the name of the label of course).

BASIC has a 'tab' point every 10 columns on the screen and when you print items separated by a comma, BASIC skips to the next column when it comes to the comma. COMAL calls these tab positions ZONES and lets you assign them to be whatever interval you wish. They will work on the printer as well as the screen. So if you wish to have columns 6 characters wide simply say ZONE 6.

COMAL also has the CURSOR statement that will put the cursor in whatever ROW and COLUMN you wish. Text can be read right off the screen under program controls too. Plus division with an integer answer is provided for with the DIV statement. A MOD statement is also available, giving the remainder to the division. You can disable the STOP key if you want. The words TRUE and FALSE are part of COMAL (just like in PASCAL). PEEK and POKE are permitted, as well as SYS (so you still can directly play with the system).

Machine code can be loaded by a program with the command OBJ-LOAD. Programs can be CHAINed together without any problems (BASIC has a few problems in this area). String arrays are available, and can have up to 33 dimensions each with whatever top and bottom indexes you wish to use (more than 256 is no problem). Of course numeric arrays also have this capability, but allow up to 36 dimensions.

Variable names in BASIC are a problem since only the first two characters are significant. COMAL allows variable names to be up to 78 characters long, with all the characters significant. These characters can include the apostrophe ('), square brackets ([]), backslash (×), and underline (represented by the left arrow key), along with the 26 letters and 10 digits. The backslash and square brackets are included so that languages with some extra 'letters' can use these. Danish uses one of these for the AE combination letter. These long variable names apply to both strings (end with a $) and

integer (end with a #).

Substrings are very easy with COMAL. No need to use special words like MID$ to manipulate a string. You can print part of a string by specifying its start character position and its end character position (BASIC would use RIGHT$, LEFT$, and MID$ to do this). But COMAL also allows you to change a substring to something else without affecting the rest of the string (BASIC won't let you do this).

COMAL also includes end of data (EOD) and end of file (EOF) pointers, making reading in data from data statements or files very easy.

Those were just some of the advantages of the language. Now I will mention how COMAL is a FRIENDLY language. It often will know what you want but not require you to type it (i.e., PASCAL requires an END. statement in every program). BASIC is friendly in this respect, but COMAL is even more friendly. Both allow you the option of leaving off the closing quote mark of a string constant if it is a at the end of a line. But COMAL does more than allow you to skip it, it puts it in for you so that your line lists with the quote mark at the end.

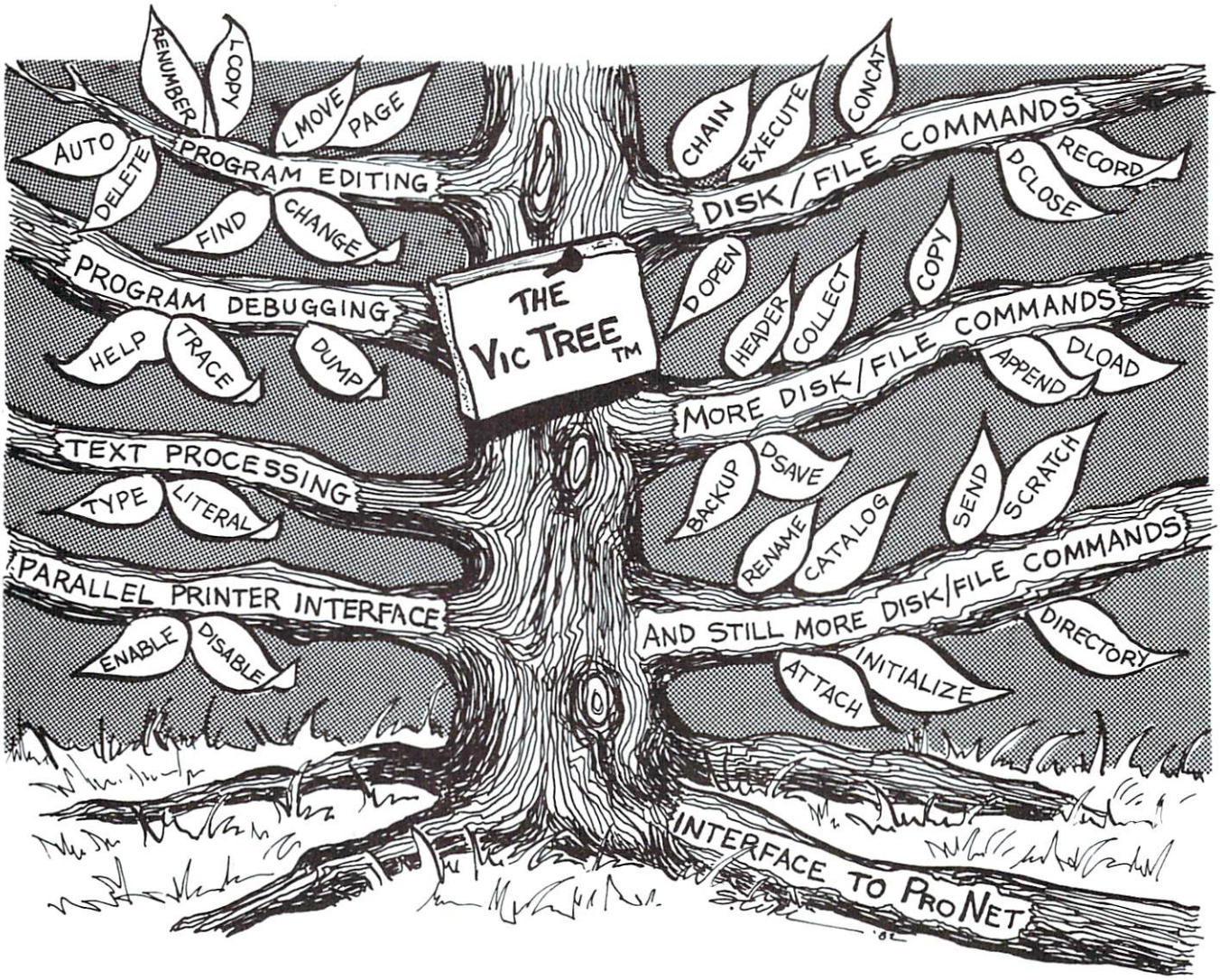
COMAL distinguishes between the comparison equals and the assignment equals (as does PASCAL) by use of a colon in front of the equal sign ':=' to mean assignment. But you don't have to type the colon, for COMAL will put it in for you. And incrementing a variable is easier than with BASIC (borrowing from ALGOL). In BASIC, to add a number to the total you would say: T=T+N while in COMAL you would say: T:+N or more appropriately, TOTAL:+NUMBER. This feature is available with strings as well as numbers.

CBM COMAL provides the same full screen editor that all PET/CBM users are fond of (and that most other micros don't have). In addition to that, COMAL checks each line as it is entered for correct syntax. If it finds a mistake, it prints a very helpful error message under the line, and puts the cursor at the spot in the line of the suspected error. Simply correct the error and hit RETURN. The error message is erased from the screen and the line that it overwrote is restored (the error message is therefore nondestructive to the information on the screen). This is a fantastic system, especially the first few times you use COMAL.

COMAL often will provide keywords you leave out if it knows that they belong in the statement.For instance, to select the printer for your output you can type SELECT "LP", but COMAL will insert the word OUTPUT and list the line as SELECT OUTPUT "LP". The word THEN (in the multi-line IF . . . THEN statements) can be omitted and will be supplied by COMAL.

So, what language should you now be using? That depends upon your situation. Both the Danish and Irish school systems are now using and teaching COMAL as their official language (with Sweden and England soon to follow). I haven't used BASIC for over a year now and haven't suffered from any withdrawal symptoms. I enjoy using COMAL. It makes you feel like a great programmer.

Skyles Electric Works Presents



The VicTree™

...Leaves your new Vic (or CBM 64) with 35 additional commands.
...Branches out to most BASIC 4.0 programs.
...Roots into most printers.

New from Skyles: the VicTree, a coordinated hardware and software package that allows your Vic to branch out in unbelievable directions and makes it easier than ever to do BASIC programming, debugging and to access your disk. And the new VicTree provides routines to interface the Vic to the powerful ProNet local network. 8kb of ROM — 4kb for the BASIC commands, 4kb for disk commands and interfacing to ProNet — plus 4kb of RAM for miscellaneous storage. Perfect not only for the new Vic but also for the Commodore 64. Unbelievably simple to use and to install, the VicTree gives you all the additional BASIC 4.0 commands to allow most BASIC 4.0 programs to work on your new Vic or CBM 64.

Now only $89.95...or **$99.95** complete with Centronics standard printer cable. (Cable alone **$19.95.)** Available now from your local dealer or order through your Visa or MasterCard toll free: (800) 227-9998 (California, Canada, Alaska, Hawaii: (415) 965-1735) or send check or money order directly to:

Skyles Electric Works

231E South Whisman Road
Mountain View, CA 94041
(415) 965-1735

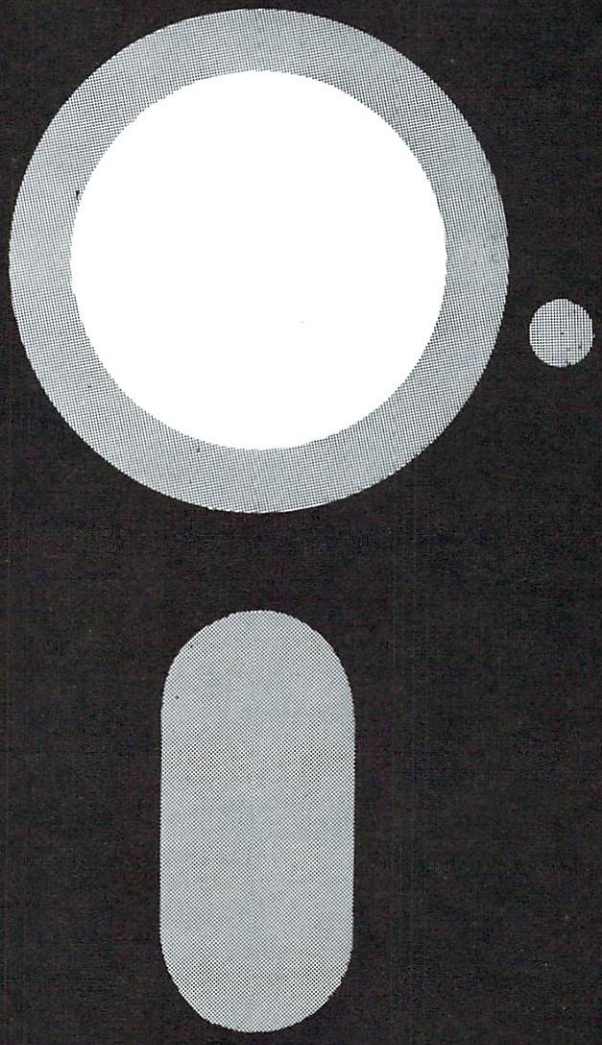
Circle #20 on the Reader Service Card

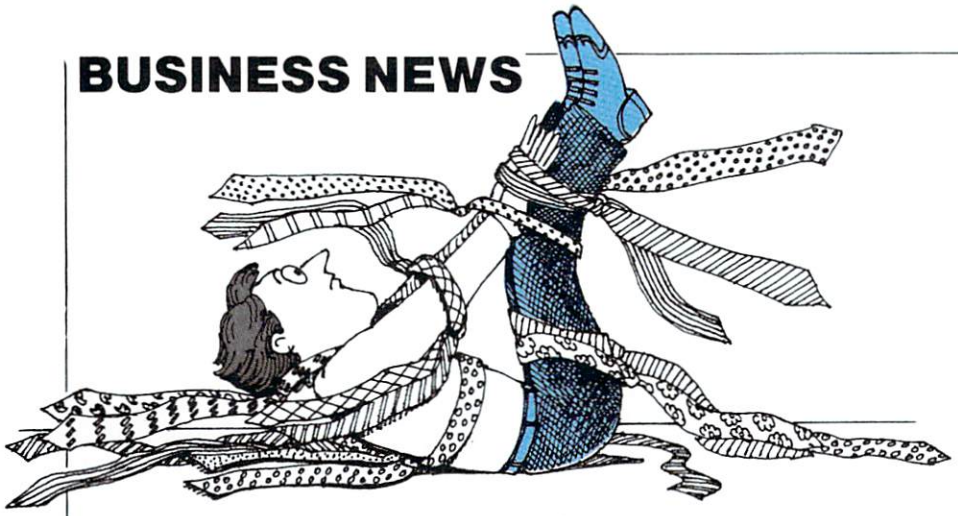
BUSINESS

commodore

A Timing System for the Pinewood Derby

The Other Office of the Future





Small Businessman No Longer Tied Up With Paperwork

by
John O'Brien

Alan R. Cadan runs his own necktie business from his garage in Stamford, Connecticut, but he is not your ordinary person running a small business from his home. Cadan's Alynn Neckwear is a $1 million business, staffed only by Cadan, his wife Lynn, and his four children.

Cadan does two types of business. Because he can do small custom orders that large companies can't handle profitably, he often supplies specialty shops, gift shops and mail order houses. The other type is custom designing ties for large companies, schools and clubs. Some of the ties are for promotional activities, advertising campaigns, and commemorative events, such as the 50th anniversary of a company. He has provided ties for such well-known companies as Ford, RCA and Saavin, and has done work for companies from France to Chile.

The business has grown enormously since he first began back in 1979. Until recently he found he could handle the work on his own. However, one aspect of the accounting paperwork started consuming too much of his time.

"Up to a certain point I could track the business manually. Last year the paperwork began interfering with my ability to create new business," he explains. "I didn't have a problem with checks, purchase journals, or payables, but too much time was spent writing invoices and entering them in a sales journal and then when payment came in to enter and cross reference them with the invoice. The number of accounts increased so dramatically that I couldn't manually send out statements."

Cadan saw two alternatives to his dilemma: either hire a bookkeeper or buy a computer. Because Cadan is a man who enjoys his privacy, he says he didn't want a stranger around the house. Also he figured that a bookkeeper would have to be paid at least $200 a week with vacation, sick days, and weekends off.

"I decided a computer was more practical, but knew nothing about computers and felt intimidated by them. I thought you had to be a mathematician to use one," he said.

His solution: "When you don't know anything about a subject, go to an expert, someone whom you can trust and rely on for help with anticipated problems."

So Cadan went to a computer chain store, but was not impressed. "I just didn't have any confidence that the sales people would back me up after the sale," he went on.

He then saw a Commodore ad placed by Haase Equipment Company in Stamford. Cadan said, "The price sounded right, so I took the ad with me to Haase. I liked the fact that they've been in business a long time, and they impressed me as people who really knew their business."

Cadan said he never had a particular brand in mind, but Haase had a predominance of Commodore products, were using Commodores themselves for their bookkeeping and were obviously familiar with Commodore as far as repairs and software were concerned.

"I told them what I needed, and they made a recommendation," he elaborated. So Cadan bought an 8032, 8050 disk drive, and a letter-quality printer. The software he is using is CMS General Accounting System. Now he is only using the Accounts Receivable part of it but feels he will eventually use it for payables and general ledger.

He said his investment has paid off. "Haase has given me great support with any software or hardware problems and even made housecalls."

While Cadan considered the accounting function a necessity, he found he also had access to the luxury of word processing. "I was getting along without it but what it has enabled me to do is look more professional. I make it look like 50 people work for the company," he laughed. "Plus it helps me gain confidence through the mail. People often are buying a product they have never seen."

One way Cadan gets business is to contact companies who are changing their name or logo about possibly getting ties to match. Now with the word processer he uses a form letter, but can input a name to make it appear to be a "personal" letter.

Cadan is not a man who's afraid of work. The business is a seven day a week job and he even carries a cordless telephone when he's doing household chores so he won't miss any calls. Still, everyone needs some free time.

"The result of my investment is that the computer allows me the luxury to enjoy my free time. I now know exactly where my business stands at any time and my monthly statements are up to date and accurate, enabling me to keep abreast of delinquent accounts. This summer I actually went to the beach and didn't feel guilty, he said of his new approach to managing his business.

The Other Office of the Future

by
Mike Richter

What will the office of the future be like? "Everyone" knows the answer to that. A terminal at every desk, wired to a central computing system that provides word processing, inventory accounting, and all of those other services that digital hardware has made possible. In fact, for a few large companies, that future is already here. But what "everyone" doesn't know is that this is not the only future—and it may not be the future for *your* office.

Don't misunderstand—the future office will use digital computers for all of those wonderful services. Or, at least, for as many as that business needs. But "everyone's" version of the office may be unnecessary, uneconomical, or just plain wrong for many applications.

The basic question is how fast you need to operate—or, more accurately, how quickly the different users of the system have to coordinate their activities. For example, travel agents booking space on a flight must have (effectively) instantaneous communications. You can't have two different systems booking on the same flight, or you may find passengers stacked up two to a seat. But most businesses, from factories to hospitals, operate on a more leisurely schedule for most of their activities. If an inventory item runs low, its reorder will be filled in a matter of days (or more). If the hospital finds out the limit has been passed at the moment that the drug is ordered, it might save a few hours to order it via computer, but probably not enough to matter. That is, the time saved by an instantaneous network is negligible compared with the delay at the supplier's end. In the same way, tying five attorneys to one another with a computer network doesn't necessarily save time or money compared with having each attorney maintain his own resources at his command, and carry information to the place it's needed by hand. That system may not be elegant, but it sure is less expensive—and often more reliable, easier to use, and otherwise better suited to the real needs of the office.

To support a computer network, a central system of substantial power is needed. Typically, there are data bases in the system (inventories, documents, whatever) that are shared and updated by all users. The software manager of such a shared system is very complex and demands quantities of memory and rapid access to large, on-line disk storage. So, "everybody's" office of the future starts out in the tens of thousands of dollars for hardware and software, then grows for terminals. Perhaps most important, the cost takes a giant leap upward when the wiring is considered—especially where the physical plant already exists and has to be made to fit.

Fitting the office into the software is another problem. The software for the central computer system is very costly and cannot be changed easily. So, the office of the future will have to adapt itself to the machine. If your accounting system doesn't fit, change the way you do your business. Unless you're prepared to spend the time and money to develop your own customized system for a mini or a mainframe, you take the least offensive choice. Generally, that means that the word processor you use to write an internal memo is the same one that can prepare a manuscript for publication. That means that you will have to learn the rules for formatting a report in order to send a note to the next office!

Okay, you want a system that suits *your* needs and can be run by people, not programmers. How can that be achieved today? The answer is simple and economical. Not surprisingly, the example we use here exploits Commodore equipment. It hinges on the compatibility of Commodore's hardware—from the VIC to the 8096. The alternative office of the future uses one or more central computers, just like "everyone's" version. But instead of wiring terminals to a mainframe, we use satellite computers. Users communicate with the central machine by carrying a disk or tape from one to the other—or over telephone lines with modems. How many satellites will one central computer support? As many as you like. What happens when the load becomes too big? Split it, at no cost in software or operations. How can you work at home? Take a satellite with you. Let's look at these points one at a time.

We can assume the central computer to be an 8032 or its successor. Let's give it a 4040 disk drive, maybe an 8050, 8250, or (when it arrives) a hard disk. It's run by a professional, who can use WordPro or Wordcraft, general ledger, and all the other sophisticated programs and data bases that exploit the capabilities of the system.

Each satellite is a Commodore 64 with a 1541 disk. Some might use a datasette instead, others might have modems, but disk is efficient and economical. The software for a satellite might be the full-blown version in the central ▶

machine, but most users will be happier with simpler, less powerful programs. For example, Commodore will soon be releasing a BASIC-language word processor that is very simple to use and even easy to modify to your personal needs. It produces the expected paper copy if you want it, but will also write out files to disk or tape. The originater can use it easily to draft any input.

The full-time data processor inputs that file to the full-blown word processor, then prints it using all the special commands and capabilities the simple Word Machine can't offer. The satellite user has software tailored to the problem at hand; the central system is the only one that demands a computer professional. The attorney doesn't have to learn computers, the data processor needn't learn law.

When the office first goes computer, both the central system and the satellites will have a lot of idle time. In this phase, people will learn what the machines can do for them, and will start asking them to do more. At first, whenever a document is to be printed the central system will be available to load and run the word processor; the document will get out almost immediately. But as the work load increases, the demands for the different programs will pile up. When conflicting demands for, say, inventory control and word processing stack up, a second central machine may be installed. One does only inventory control, the other picks up all the remaining load.

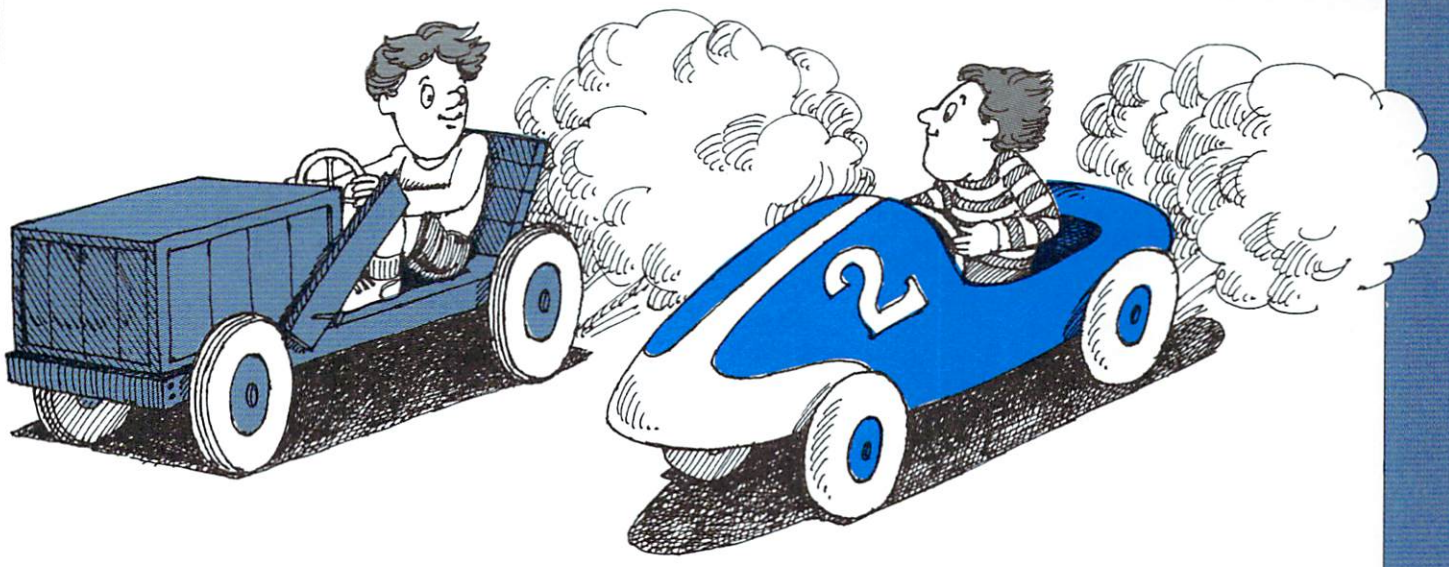
If the bottleneck turns out to be the printer, a Commodore 64 can run a dot-matrix printer for drafts, and the 8300 can stay on the central machine for letter-quality printing. The variations are limitless, but in the typical office it is unlikely that any business that could exhaust the capabilities could ever be considered "small." A starting system for three or four satellites and a single central system will cost less than $10,000. Considering that sum won't even buy a minicomputer's disk drive, you can see that "everybody's" office may be far into your future, but this one may be possible for you now.

The possibilities of the satellite concept are endless. For example, suppose an otherwise capable employee is confined at home. The satellite system is self-contained and has all the power he needs to do much of his work—given only occasional access to the central computer. So, someone drops by every day or so to pick up and drop off disks or tapes, and the invalid can continue to produce.

Or the travelling salesman ties in to the office on the telephone at 7 AM (when the rates are low), and dumps his contact reports, trouble memos, and sales to home base, then picks up the new price list and the latest corporate reorganization. Even communicating at 300 baud (30 characters per second), that's about 300 words per minute, and that spells efficiency.

If you tried to use a terminal to do the job, the delay would be prohibitive (response time at telephone speeds will drive an efficient user up the wall), and the cost of tying up the telephones while the user is thinking and typing will help AT&T a lot more than it will your company. But use a microcomputer like the 64 instead of a terminal, and all of the interaction is local—just you and your machine; all communication is rapid—just your machine and theirs. Compared with the terminal, the 64 is cheaper, smaller, more durable, and much more portable. And the central machine or machines will be big enough for the job—and need be no bigger.

One more idea before closing this overlong note—maybe there's a whole new business out there for someone to exploit the satellite system. More accurately, it's a new approach to the old Service Bureau idea. Picture a shop, or a Commodore dealer, whose job is to handle some of the central functions of a number of businesses. Maybe a business doesn't want to start with a letter-quality printer for its low volume of formal output. The service shop provides the printout to your specifications using hardware shared among many applications (including their own). The salesman just get a reply to his customer's problem? Take it to the service shop and deliver hard copy where and when it does the most good. So now we have an office of the future that looks a little different from the one "everyone" knows. Among other things, it may not even be in the office that "everyone" knows today!



A Timing System for the Pinewood Derby

by
Miguel A. Feyjoo

The Pinewood Derby is a race between model cars built by Cub Scouts performed on an inclined track. All of the cars are started at the top of the track. A starting gate is activated which releases the cars to roll down the track under their own power. The races are run in heats, and the winning car from each heat is pitted against other winners in later heats until the field is narrowed down to one car, which is eventually declared the winner.

Due to the closeness of some of the races, it was felt that an automatic timing system was desirable to eliminate doubt concerning which car finished first in a race. At the 1982 Hato Rey, Puerto Rico, BSA Pinewood Derby, a PET microcomputer was used as the timing device to measure the elapsed time of the individual cars as they completed their run down the track. (A similar system was provided at the 1979 Fort Wayne Pinewood Derby by C. W. Lilie using an Ohio Scientific Challenger 1P Computer with a Motorola 6821PIA module).

A micro-switch was installed at the starting gate to automatically signal the start of each race. At the end of the track, a set of photo cells were located so that the order of the cars crossing the finish line could be detected. The system used a PET 40-column microcomputer, although the program was originally developed for a Commodore CBM 8032 microcomputer. The timing data was read through the microcomputer parallel port providing a timing resolution of 4 milliseconds and counting up to 999.996 seconds. This provided very good resolution in timing of the cars on the track since the typical times for the cars to run the track were 4 to 10 seconds. Ties in the first three places were the only ones of concern, because the heat would have to be rerun, but there were no ties detected. Only on very few occasions a car defeated the next car by the minimum amount of 4 milliseconds.

The hardware interface between the racetrack and the microcomputer system was made with seven C-MOS integrated circuits consisting of four dual type D-flip/flops, a dual quad input not-or gate, a quad dual input nor gate, and a hex inverter. The above chips were mounted in a wire wrapped card inside a 4 × 5 × 3 inch aluminum minibox. The cost of the parts including cables and connectors was $48.26. The minibox was wired and assembled in one day by Wilfredo Rosado, electronics technician at the Arecibo Observatory.

Eight photo-transistors were used at the finish line of the race track to act as the photo cells. The operation was started by the operator pressing the "RESET" push button at the minibox and then the space bar at the microcomputer prior to the start of each race. The activating of the starting gate sets the D-flip/flops which provide a signal indicating the start of the race and a signal on each of the eight channels to the microcomputer parallel port (see Figure 1).

As the cars finish the race, the photo cells are triggered in the order of the finish of the cars. When the photo cell is activated, it resets the latch. Therefore, the time during which the latch is set is the running time of the race. If less than eight cars were running in a race, or any car stopped before reaching the finish line, the operator pressed the "RESET" push button after the competing cars passed the finish line and all other cars stopped, to obtain the results.

The timing pulses for the time measurements are obtained from the microcomputer clock, which are derived from a quartz crystal that provides very accurate timing. The pulses are programmed to a rate of one every four milliseconds after the start of the race and, they are turned off at the end of the race. The time measurements obtained at each race may be used to compare with the time obtained by other cars on other races to determine ▸

the possibility of winning a race against the other car.

The microcomputer software uses a routine that waits for a high signal on the CA1 line when the starting gate is activated. Then it will turn on the 4 millisecond timing pulses in the CB2 line. While there are cars running, a logic gate will pass the 4 millisecond timing pulses back to the microcomputer through the CA1 line. For each track input that is high, or a 1, indicating that the car has not reached the finish line, the microcomputer increments or adds 4 to the timer location for each of the tracks.

Thus, each time the microcomputer gets a time pulse, which is once every 4/1000 of a second, it increases the count by four, or 4 milliseconds, on the time storage location. Summing the number of pulses during which the D-flip/flop was set then indicates the time in milliseconds which is the time required for the car to run the length of the track. Since all eight of the channels are sampled at the same time, there is no ambiguity in the timing, and the accuracy is the same as the clock, plus or minus 4 milliseconds.

Figure 1 is a schematic of the interface box used for the race. At the top is a single inverter connected as a positive edge differentiator which is activated by the start switch, when the starting gate is released. This circuit is connected to all eight D-flip/flops and sets all of them at the same time. Each of the eight channels is identical. Each channel is fed from a photo-transistor (type VTT-1113 with an 18,000 ohm resistor from its collector to the five volt supply) at the finish line of the track. The photo-transistor output is routed to the edge clock input of the D-flip/flop to reset it when the car crosses the finish line.

The state of the flip/flops is monitored by the microcomputer via its parallel port. An inverter gate is connected to all the reset inputs of the D-flip/flops to allow the operator to reset the flip/flops prior to the start of a race or at the end of a race if less then eight cars are running. At the inverted output of the flip/flops there is an eight input OR gate to detect when any cars are running and enable the CB2 pulses into the CA1 line.

The program was written to run on the Commodore CBM 8032 and a second version was written to run on the PET 40-column microcomputer. The software consists of an assembly program imbedded into a BASIC program. The assembly program was needed to provide the required speed to measure the running time on all eight tracks. The assembly program code was changed to DATA statements to allow its loading by the BASIC program into the microcomputer upper memory.

The system could be used with most any type of microcomputer having the 8-bit wide data bus. The software provided is written in 6502 assembly language, but the same principles could be applied to any other system.

The table for race number 52 shows the results of one of the semifinals. This printout is arranged in order by finish position with the track number shown in the second column and finally the elapsed time. This data is printed out at the end of each race, at the option of the operator, and it provides a permanent record of the finishing order. On this race it can be seen that the car that arrived in first place defeated the car arriving in second place by only 4 milliseconds. Also, the car that arrived third defeated the next car by 4 milliseconds. The cars on tracks 1 and 2 did not make it to the finish line. They show identical times because the operator pressed the reset button to finish the race and get the results for the printer. Race 71 was the last heat from which the winner and the second and third places were determined. The closeness of this race, in which the seventh place followed the first place by only 172 milliseconds, shows the need of an automatic timing system.

The program could be expanded to include names or car numbers for each race so that this information could be printed out. The last option was not included to keep the operation of this program as simple as possible. During the running of the race, the monitor of the microcomputer provides a real time display for each track showing the time elapsed. As soon as any car finishes the course, the timer for that car stops, showing the time and the finish position of that car on the monitor.

The program listing is shown in BASIC language. Line 330 sets the program memory limit so that the BASIC program does not overlap the memory space required for the assembly language program. Lines 350–370 load the assembly language program, which is stored as DATA statement on lines 1030–1270 into high memory between hexadecimal 7800 and 7988, at the top of the microcomputer's 32K user memory. Lines 380–430 request the input of the date and time to save this data for printing on race table. Lines 450–470 and the subroutine at lines 960–1000 wait for the beginning of the race. The operator must press the space bar at the microcomputer's keyboard before the starting gate is released.

Line 480 transfers control to the assembly program by calling the sys 30720. The microcomputer will then display the running time for each track on real time and at the end of the race it will return to the BASIC program at line 490. Lines 510–590 will collect the time data from the storage locations in which the time was stored during the race and computed by the assembly language program. Lines 600–660 are a sorting routine which sorts the time information in the order that the cars finished. Lines 680 through 770 print the result of the race on the CRT screen and lines 820 through 940 print the results on the printer for a hard copy.

The program then returns to the wait routine at lines 450–470 until the space bar is pressed again prior to the start of a new race. The final list is the assembly language program used. This was written in standard 6502 assembly code using the MAE Macroassembler. The program listing is full of comments so that any person with a knowledge of assembly programming can easily understand it.

Race Track Logic

Figure 1

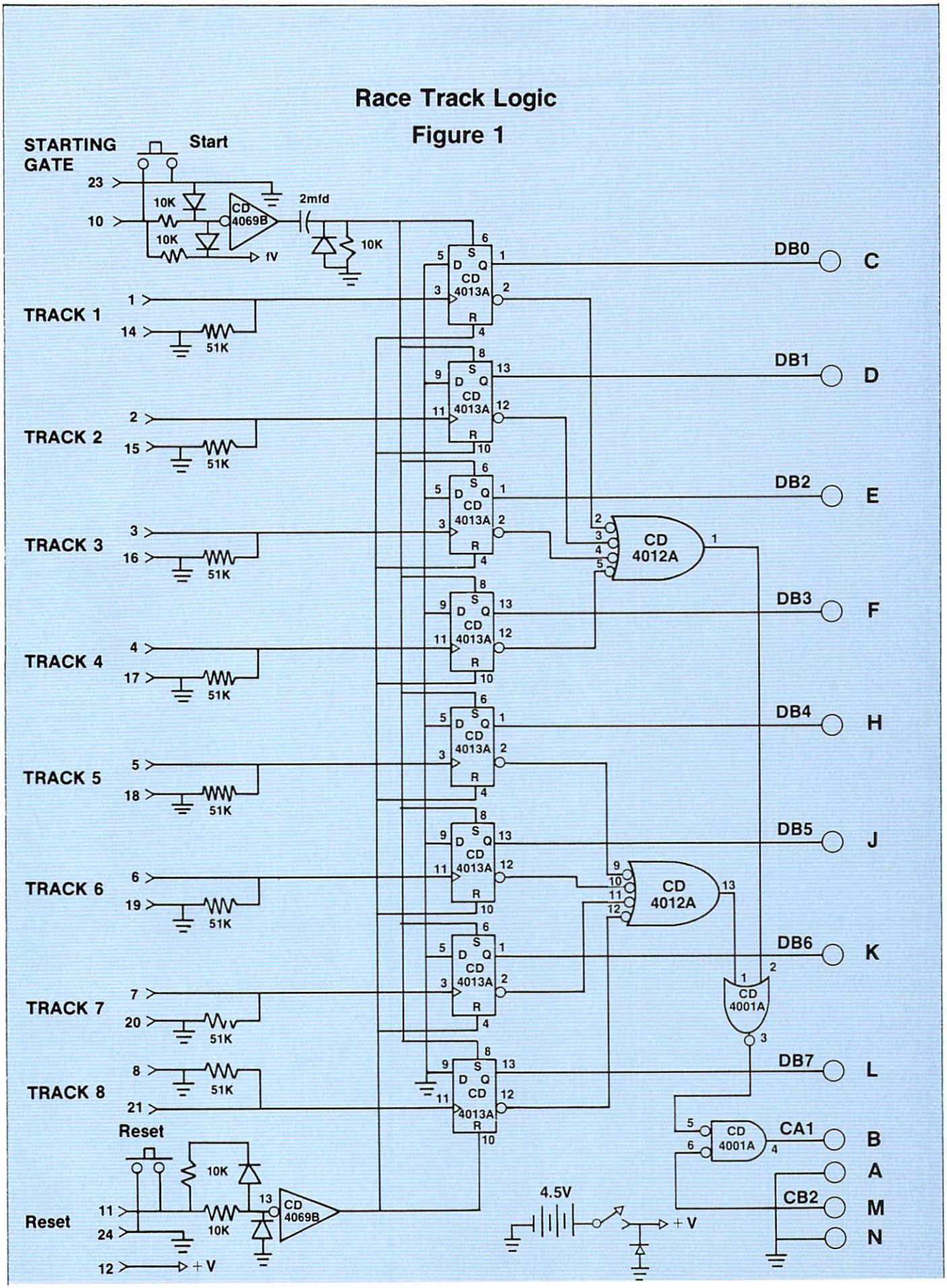
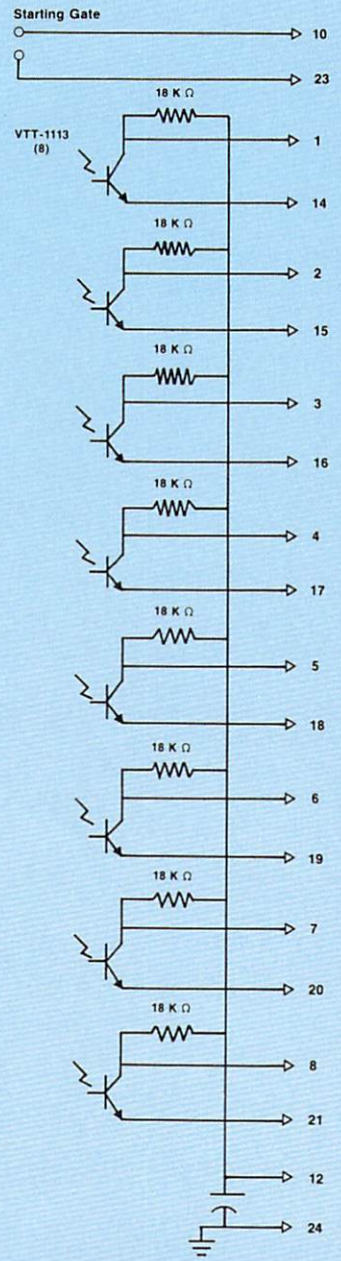


Figure 2



Finish Line Photo Cells and Interface Box Front View

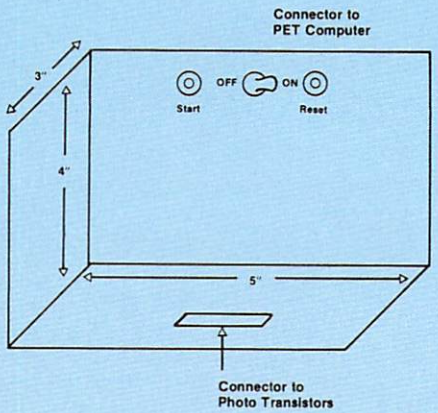


Table I
Pinewood Derby Race Number 52
08/29/82 11:52:31

Place	Track No.	Time
1	6	4.308
2	8	4.312
3	3	4.392
4	7	4.396
5	5	4.424
6	4	4.516
7	1	5.556
8	2	5.556

For the Boy Scouts of America
In Hato Rey, Puerto Rico

Table II
Pinewood Derby Race Number 71
08/29/82 12:47:27

Place	Track No.	Time
1	2	4.212
2	7	4.272
3	1	4.28
4	3	4.352
5	8	4.36
6	4	4.364
7	6	4.384
8	5	4.484

For the Boy Scouts of America
In Hato Rey, Puerto Rico

```
100 REM ************************************************
110 REM *   BOY SCOUTS PINEWOOD DERBY RACE PROGRAM     *
120 REM *                    BY                        *
130 REM *                MIGUEL A FEYJOO               *
140 REM *     P.O. BOX 1441, ARECIBO, P. R. 00613      *
150 REM *              TEL. 809/878-5877               *
160 REM *       VERSION 2.0 OF AUGUST 20, 1982         *
170 REM *      LAST REVISION ON AUGUST 23, 1982        *
180 REM ************************************************
190 :
200 REM   THIS PROGRAM RUNS IN A COMMODORE 40 COLUMNS (PET)
210 REM   MICROCOMPUTER WITH LEVEL 3.0 ROMS.
220 REM   THE PROGRAM WILL COUNT THE ELAPSED TIME BETWEEN A START
230 REM   SIGNAL FROM THE STARTING GATE AND A FINISH SIGNAL FOR UP
240 REM   TO EIGHT RACING CARS. THE TIME IS MEASURED IN SECONDS
250 REM   WITH A RESOLUTION OF 4 MILLISECONDS.
260 :
270 PRINT "";
280 PRINT SPC(7)"BOY SCOUTS PINEWOOD DERBY": PRINT
290 PRINT SPC(13)"RACE PROGRAM": PRINT
300 PRINT SPC(18)"BY": PRINT
310 PRINT SPC(14)"MIGUEL A FEYJOO": PRINT
320 PRINT SPC(12)"P. O. BOX 1441": PRINT
330 PRINT SPC(10)"ARECIBO, P. R. 00613": PRINT
340 POKE 52,0: POKE 53,120: REM SET TOP OF BASIC MEMORY
350 REM LOAD ASSEMBLY LANGUAGE PROGRAM
360 FOR I = 30720 TO 31114
370 READ D%: POKE I,D%
380 NEXT I
390 PRINT "ENTER DATE (MM/DD/YY)"
400 INPUT DATE$
410 PRINT: PRINT "ENTER TIME (HHMMSS)"
420 INPUT HO$: TI$ = HO$: RA = 1
430 POKE 158,1: POKE 623,48
440 PRINT "ENTER RACE NUMBER        ";RA;"";: INPUT RI
450 PRINT "": IF RI <> 0 THEN RA = RI
460 PRINT"PRESS SPACE BAR WHEN RACE NO.";RA;"IS READY": PRINT
470 GET A$: IF A$ =CHR$(32) THEN 490
480 GOSUB 970: GOTO 460
490 TS$ = TI$:SYS 30720: REM GO TO MEASURE RACE TIMES
500 TS$ = LEFT$(TS$,2)+":"+MID$(TS$,3,2)+":"+RIGHT$(TS$,2)
510 REM GO PICK UP RACE TIMES FROM ASSEMBLY PROGRAM
520 FOR I = 1 TO 8
530 TI(I) = 0
540 FOR J = 1 TO 3
550 K0 = PEEK(31042+I+J*8)
560 K1 = INT(K0/16)
570 TI(I) = INT((TI(I)+(10*K1+K0-16*K1)*10^(3-2*J))*1000)/1000
580 NEXT J
590 O(I) = I
600 NEXT I
610 F = 0
620 FOR J = 7 TO 1 STEP -1
630 IF TI(O(J+1))>=TI(O(J)) THEN 660
640 F = 1
650 T =O(J): O(J)=O(J+1): O(J+1)=T
660 NEXT J
670 IF F = 1 THEN 610
680 REM NOW WE DISPLAY RACE RESULTS
690 PRINT "";SPC(1)"PINEWOOD DERBY RACE NUMBER";RA
700 PRINT SPC(5);DATE$;SPC(4);TS$
710 PRINT SPC(2)"PLACE";SPC(4)"TRACK NO.";SPC(4)"TIME"
720 PRINT
730 FOR I = 1 TO 8
740 PRINT SPC(3);I;SPC(7);O(I);SPC(6);TI(O(I)): PRINT
750 NEXT I
```

```
760 PRINT SPC(1)"FOR THE BOY SCOUTS OF AMERICA"
770 PRINT SPC(14)"BY"
780 PRINT SPC(8)"MIGUEL A FEYJOO"
790 PRINT "PRESS 'P' TO GET PRINTED RECORD OF RACE"
800 PRINT "OR PRESS 'C' TO CONTINUE WITH NEXT RACE";
810 GET A$: IF A$ <> "P" AND A$ <> "C" THEN 810
820 IF A$ = "C" THEN PRINT"": RA = RA+1: GOTO 430
830 OPEN 4,4: PRINT#4
840 PRINT#4,SPC(20)"PINEWOOD DERBY RACE NUMBER";RA: PRINT#4
850 PRINT#4,SPC(24);DATE$;SPC(4);TS$: PRINT#4
860 PRINT#4,SPC(21)"PLACE";SPC(4)"TRACK NO.";SPC(4)"TIME"
870 PRINT#4
880 FOR I = 1 TO 8
890 PRINT#4,SPC(22);I;SPC(7);O(I);SPC(6);TI(O(I)): PRINT#4
900 NEXT I
910 PRINT#4,SPC(20)"FOR THE BOY SCOUTS OF AMERICA"
920 PRINT#4: PRINT#4,SPC(23)"IN HATO REY, PUERTO RICO"
930 PRINT#4: PRINT#4,SPC(33)"BY"
940 PRINT#4: PRINT#4,SPC(27)"MIGUEL A FEYJOO"
950 PRINT#4: PRINT#4: PRINT#4: CLOSE 4
960 PRINT: RA = RA+1: GOTO 430
970 REM SUBROUTINE TO PRINT TIME
980 HO$ = TI$: HH$ = LEFT$(HO$,2)
990 MM$ = MID$(HO$,3,2): SS$ = RIGHT$(HO$,2)
1000 PRINT "THE TIME IS ";HH$;" HOURS ";MM$;" MIN. AND ";SS$;" SEC."
1010 RETURN
1020 :
1030 REM BELOW ARE THE DATA STATEMENTS OF THE ASSEMBLY PROGRAM
1040 DATA 32,41,226,169,0,141,67,232,173,75,232,9,17,141,75,232
1050 DATA 169,248,141,72,232,169,0,141,74,232,173,76,232,9,1,141
1060 DATA 76,232,173,65,232,162,18,189,98,121,157,128,128,202,208,247
1070 DATA 162,49,142,211,128,232,142,35,129,232,142,115,129,232,142,195
1080 DATA 129,232,142,19,130,232,142,99,130,232,142,179,130,232,142,3
1090 DATA 131,169,255,141,65,121,169,0,162,32,157,66,121,202,16,250
1100 DATA 169,2,45,77,232,240,249,169,60,141,74,232,173,65,232,141
1110 DATA 64,121,205,65,121,240,28,77,65,121,162,7,10,144,11,72
1120 DATA 238,74,121,173,74,121,157,66,121,104,202,16,239,173,64,121
1130 DATA 141,65,121,162,7,10,144,33,72,248,24,169,4,125,91,121
1140 DATA 157,91,121,144,18,169,0,125,83,121,157,83,121,144,8,169
1150 DATA 0,125,75,121,157,75,121,104,216,202,16,217,162,7,189,117
1160 DATA 121,141,5,121,141,220,120,189,125,121,141,4,121,141,219,120
1170 DATA 189,66,121,240,8,24,105,48,160,11,153,170,129,160,7,185
1180 DATA 131,121,141,230,120,189,75,121,72,41,15,32,0,121,104,74
1190 DATA 74,74,74,32,0,121,192,0,208,229,202,16,193,76,16,121
1200 DATA 24,105,48,153,170,129,136,192,4,208,4,169,46,208,244,96
1210 DATA 173,64,121,9,0,240,23,173,79,232,9,0,208,6,141,64
1220 DATA 121,76,114,120,169,2,45,77,232,240,236,76,108,120,173,75
1230 DATA 232,41,239,141,75,232,173,76,232,41,254,141,76,232,96,170
1240 DATA 0,0,8,7,6,5,4,3,2,1,8,0,0,0,0,0
1250 DATA 0,0,0,25,25,25,25,25,25,25,25,36,36,36,36,36
1260 DATA 36,36,36,84,82,65,67,75,32,32,84,73,77,69,32,32
1270 DATA 32,32,80,79,83,128,129,129,129,130,130,130,131,214,38,118
1280 DATA 198,22,102,182,6,75,75,75,83,83,91
1290 END: REM THIS IS THE END OF THE PROGRAM
READY
100 REM ************************************************
110 REM *    BOY SCOUTS PINEWOOD DERBY RACE PROGRAM    *
120 REM *                       BY                     *
130 REM *                MIGUEL A FEYJOO               *
140 REM *     P.O. BOX 1441, ARECIBO, P. R. 00613      *
150 REM *               TEL. 809/878-5877              *
160 REM *        VERSION 1.0 OF AUGUST 19, 1982        *
170 REM *      LAST REVISION ON AUGUST 20, 1982        *
180 REM ************************************************
190 :
```

```
200 REM   THIS PROGRAM RUNS IN A COMMODORE CBM 8032 MICROCOMPUTER.
210 REM   THE PROGRAM WILL COUNT THE ELAPSED TIME BETWEEN A START
220 REM   SIGNAL FROM THE STARTING GATE AND A FINISH SIGNAL FOR UP
230 REM   TO EIGHT RACING CARS. THE TIME IS MEASURED IN SECONDS
240 REM   WITH A RESOLUTION OF 4 MILLISECONDS.
250 :
260 PRINT "";
270 PRINT SPC(15)"BOY SCOUTS PINEWOOD DERBY": PRINT
280 PRINT SPC(21)"RACE PROGRAM": PRINT
290 PRINT SPC(26)"BY": PRINT
300 PRINT SPC(20)"MIGUEL A FEYJOO": PRINT
310 PRINT SPC(20)"P. O. BOX 1441": PRINT
320 PRINT SPC(18)"ARECIBO. P. R. 00613": PRINT
330 POKE 52,0: POKE 53,120: REM SET TOP OF BASIC MEMORY
340 REM LOAD ASSEMBLY LANGUAGE PROGRAM
350 FOR I = 30720 TO 31114
360 READ D%: POKE I,D%
370 NEXT I
380 PRINT "ENTER DATE (MM/DD/YY)"
390 INPUT DATE$
400 PRINT: PRINT "ENTER TIME (HHMMSS)"
410 INPUT HO$: TI$ = HO$: RA = 1
420 POKE 158,1: POKE 623,48
430 PRINT "ENTER RACE NUMBER        ";RA;"";: INPUT RI
440 PRINT "": IF RI <> 0 THEN RA = RI
450 PRINT"PRESS SPACE BAR WHEN RACE NO.";RA;"IS READY": PRINT
460 GET A$: IF A$ =CHR$(32) THEN 480
470 GOSUB 960: GOTO 450
480 TS$ = TI$: SYS 30720: REM GO TO MEASURE RACE TIMES
490 TS$ = LEFT$(TS$,2)+":"+MID$(TS$,3,2)+":"+RIGHT$(TS$,2)
500 REM GO PICK UP RACE TIMES FROM ASSEMBLY PROGRAM
510 FOR I = 1 TO 8
520 TI(I) = 0
530 FOR J = 1 TO 3
540 K0 = PEEK(31042+I+J*8)
550 K1 = INT(K0/16)
560 TI(I) = INT((TI(I)+(10*K1+K0-16*K1)*10^(3-2*J))*1000)/1000
570 NEXT J
580 O(I) = I
590 NEXT I
600 F = 0
610 FOR J = 7 TO 1 STEP -1
620 IF TI(O(J+1))>=TI(O(J)) THEN 650
630 F = 1
640 T =O(J): O(J)=O(J+1): O(J+1)=T
650 NEXT J
660 IF F = 1 THEN 600
670 REM NOW WE DISPLAY RACE RESULTS
680 PRINT "";SPC(20)"PINEWOOD DERBY RACE NUMBER";RA
690 PRINT SPC(24);DATE$;SPC(4);TS$
700 PRINT SPC(21)"PLACE";SPC(4)"TRACK NO.";SPC(4)"TIME"
710 PRINT
720 FOR I = 1 TO 8
730 PRINT SPC(22);I;SPC(7);O(I);SPC(6);TI(O(I)): PRINT
740 NEXT I
750 PRINT SPC(20)"FOR THE BOY SCOUTS OF AMERICA"
760 PRINT SPC(33)"BY"
770 PRINT SPC(27)"MIGUEL A FEYJOO"
780 PRINT "PRESS 'P' TO GET PRINTED RECORD OF RACE"
790 PRINT "OR PRESS 'C' TO CONTINUE WITH NEXT RACE";
800 GET A$: IF A$ <> "P" AND A$ <> "C" THEN 800
810 IF A$ = "C" THEN PRINT"": RA = RA+1: GOTO 420
820 OPEN 4,4: PRINT#4
830 PRINT#4,SPC(20)"PINEWOOD DERBY RACE NUMBER";RA: PRINT#4
840 PRINT#4,SPC(24);DATE$;SPC(4);TS$: PRINT#4
```

```
850 PRINT#4,SPC(21)"PLACE";SPC(4)"TRACK NO.";SPC(4)"TIME"
860 PRINT#4
870 FOR I = 1 TO 8
880 PRINT#4,SPC(22);I;SPC(7);O(I);SPC(6);TI(O(I)): PRINT#4
890 NEXT I
900 PRINT#4,SPC(20)"FOR THE BOY SCOUTS OF AMERICA"
910 PRINT#4: PRINT#4,SPC(23)"IN HATO REY, PUERTO RICO"
920 PRINT#4: PRINT#4,SPC(33)"BY"
930 PRINT#4: PRINT#4,SPC(27)"MIGUEL A FEYJOO"
940 PRINT#4: PRINT#4: PRINT#4: CLOSE 4
950 PRINT: RA = RA+1: GOTO 420
960 REM SUBROUTINE TO PRINT TIME
970 HO$ = TI$: HH$ = LEFT$(HO$,2)
980 MM$ = MID$(HO$,3,2): SS$ = RIGHT$(HO$,2)
990 PRINT "THE TIME IS ";HH$;" HOURS ";MM$;" MINUTES AND ";SS$;" SECONDS"
1000 RETURN
1010 :
1020 REM BELOW ARE THE DATA STATEMENTS OF THE ASSEMBLY PROGRAM
1030 DATA 32,78,224,169,0,141,67,232,173,75,232,9,17,141,75,232
1040 DATA 169,248,141,72,232,169,0,141,74,232,173,76,232,9,1,141
1050 DATA 76,232,173,65,232,162,18,189,98,121,157,4,129,202,208,247
1060 DATA 162,49,142,167,129,232,142,71,130,232,142,231,130,232,142,135
1070 DATA 131,232,142,39,132,232,142,199,132,232,142,103,133,232,142,7
1080 DATA 134,169,255,141,65,121,169,0,162,32,157,66,121,202,16,250
1090 DATA 169,2,45,77,232,240,249,169,60,141,74,232,173,65,232,141
1100 DATA 64,121,205,65,121,240,28,77,65,121,162,7,10,144,11,72
1110 DATA 238,74,121,173,74,121,157,66,121,104,202,16,239,173,64,121
1120 DATA 141,65,121,162,7,10,144,33,72,248,24,169,4,125,91,121
1130 DATA 157,91,121,144,18,169,0,125,83,121,157,83,121,144,8,169
1140 DATA 0,125,75,121,157,75,121,104,216,202,16,217,162,7,189,117
1150 DATA 121,141,5,121,141,220,120,189,125,121,141,4,121,141,219,120
1160 DATA 189,66,121,240,8,24,105,48,160,11,153,170,129,160,7,185
1170 DATA 131,121,141,230,120,189,75,121,72,41,15,32,0,121,104,74
1180 DATA 74,74,74,32,0,121,192,0,208,229,202,16,193,76,16,121
1190 DATA 24,105,48,153,170,129,136,192,4,208,4,169,46,208,244,96
1200 DATA 173,64,121,9,0,240,23,173,79,232,9,0,208,6,141,64
1210 DATA 121,76,114,120,169,2,45,77,232,240,236,76,108,120,173,75
1220 DATA 232,41,239,141,75,232,173,76,232,41,254,141,76,232,96,170
1230 DATA 0,0,8,7,6,5,4,3,2,1,8,0,0,0,0,0
1240 DATA 0,0,0,25,25,25,25,25,25,25,25,36,36,36,36,36
1250 DATA 36,36,36,84,82,65,67,75,32,32,84,73,77,69,32,32
1260 DATA 32,32,80,79,83,129,130,130,131,132,132,133,134,170,74,234
1270 DATA 138,42,202,106,10,75,75,75,83,83,91
1280 END: REM THIS IS THE END OF THE PROGRAM
READY
END OF MAE PASS!
```

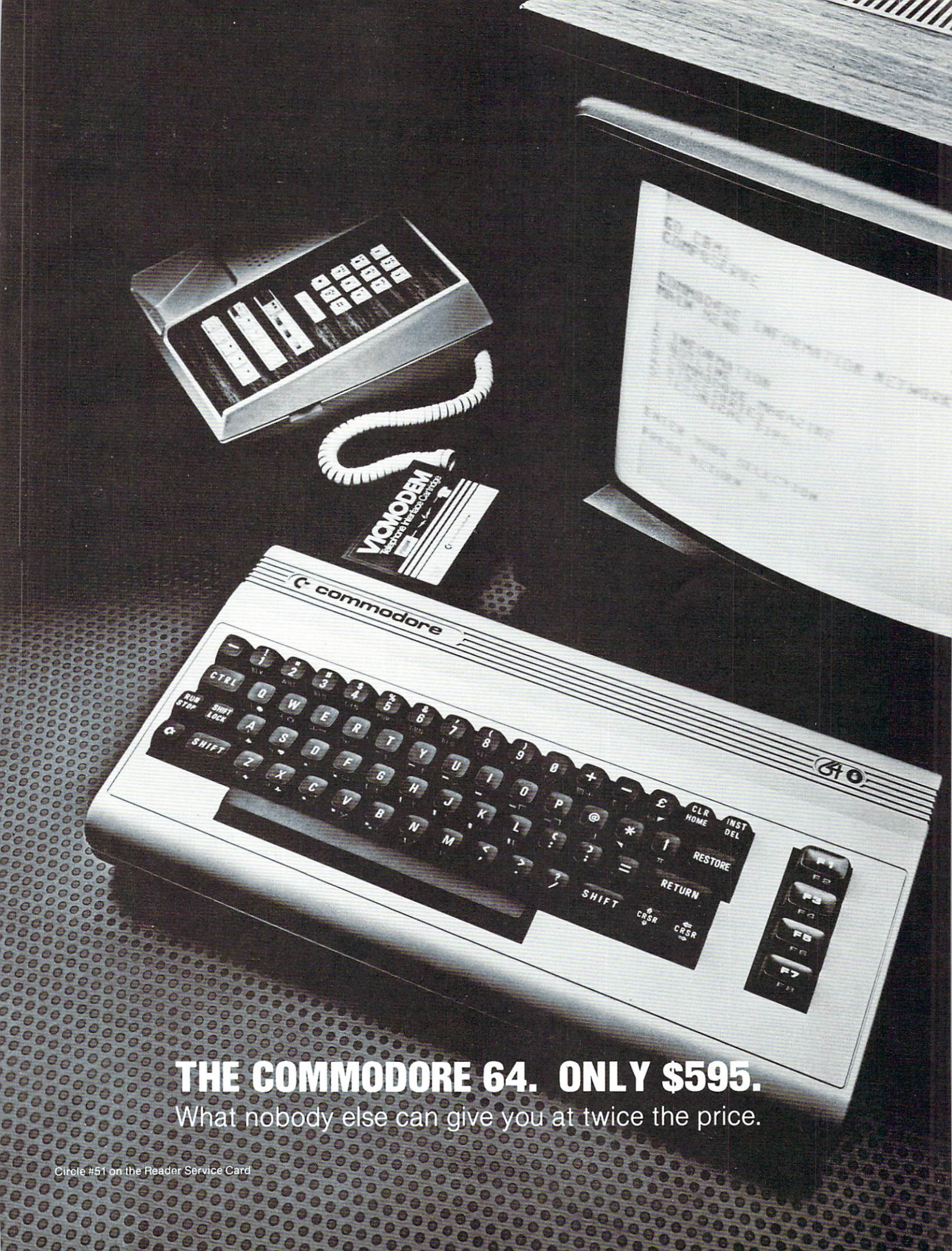
```
0100 ;     ***********************************************
0110 ;     *   Boy Scouts Pinewood Derby Race Program    *
0120 ;     *                    by                       *
0130 ;     *              Miguel A Feyjoo                *
0140 ;     *     P.O. Box 1441, Arecibo, P. R. 00613     *
0150 ;     *              Tel. 809/878-5877              *
0160 ;     *       Version 2.0 of August 20, 1982        *
0170 ;     *     Last Revision on August 23, 1982        *
0180 ;     ***********************************************
0190
0200 ; This program runs in a Commodore 40-columns (PET)
0210 ; microcomputer with level 3.0 ROMS.
0220 ; The program will count the elapsed time between a start
0230 ; signal from the starting gate and a finish signal for up
0240 ; to eight racing cars. The time is measured in seconds
0250 ; with a resolution of 4 milliseconds.
0260
```

```
                    0270              .OS                   ;generate object code
                    0280              .BA $0400             ;program start location
                    0290
                    0300 ; The following bytes are used to allow the RUN command
                    0310 ; to start program from BASIC after loading.
                    0320
0400- 00 0D 04      0330              .BY 0 13 4 10 0 158   ;10 SYS
0403- 0A 00 9E
0406- 28 33 30      0340              .BY '(30720)' 0 0 0   ;30720 Dec = 7800 Hex
0409- 37 32 30
040C- 29 00 00
040F- 00
                    0350              .BA $7800             ;Re-start at this location
                    0360
                    0370 ;          +++++++++++ Definitions ++++++++++++
                    0380
                    0390 CLRSCRN      .DE $E229             ;Clear screen sys routine
                    0400 PIA.DRA      .DE $E841             ;PIA Data Reg port A
                    0410 PIA.DDA      .DE $E843             ;PIA Data Dir. reg. A
                    0420 PIA.ACR      .DE $E84B             ;PIA Auxiliary Control Reg
                    0430 PIA.PCR      .DE $E84C             ;PIA Peripheral Cont. Reg
                    0440 PIA.IFR      .DE $E84D             ;PIA Interrupt Flag Reg.
                    0450 PIA.PPA      .DE $E84F             ;PIA Parallel Port A
                    0460 PIA.TM2      .DE $E848             ;PIA TiMer 2
                    0470 PIA.SHR      .DE $E84A             ;PIA SHift Register
                    0480 DLOC         .DE $8000             ;Display start location
                    0490
                    0500 ; ++++++++++++ Clear Screen ++++++++++++
                    0510
7800- 20 29 E2      0520              JSR CLRSCRN           ;Clear screen
                    0530
                    0540 ; ++++++++ Initial set up of input port ++++++++
                    0550
7803- A9 00         0560              LDA #$00              ;load A reg with zeros
7805- 8D 43 E8      0570              STA PIA.DDA           ;Set user port to input
7808- AD 4B E8      0580              LDA PIA.ACR           ;read data of aux cont reg
780B- 09 11         0590              ORA #%00010001        ;Enable shift register
780D- 8D 4B E8      0600              STA PIA.ACR           ;and enable port A latch
7810- A9 F8         0610              LDA #248              ;Set A reg to 248
7812- 8D 48 E8      0620              STA PIA.TM2           ;Set timer microseconds
7815- A9 00         0630              LDA #0                ;Load A register with 0
7817- 8D 4A E8      0640              STA PIA.SHR           ;Set shift register to 0
781A- AD 4C E8      0650              LDA PIA.PCR           ;read peripheral cont. reg
781D- 09 01         0660              ORA #1                ;Add bit 0 to A register
781F- 8D 4C E8      0670              STA PIA.PCR           ;Set CA1 to pos edge trig.
7822- AD 41 E8      0680              LDA PIA.DRA           ;read to reset CA1 flag
                    0690
                    0700 ; ++++++++++ Label Screen ++++++++++
                    0710
7825- A2 12         0720              LDX #$12              ;Put track time pos on dis
7827- BD 62 79      0730 LDDATA       LDA DDATA-1,X         ;Get data from table
782A- 9D 80 80      0740              STA DLOC+128,X
782D- CA            0750              DEX
782E- D0 F7         0760              BNE LDDATA            ;End of display
7830- A2 31         0770              LDX #$31              ;Put track no. on display
7832- 8E D3 80      0780              STX DLOC+211
7835- E8            0790              INX
7836- 8E 23 81      0800              STX DLOC+291
7839- E8            0810              INX
783A- 8E 73 81      0820              STX DLOC+371
783D- E8            0830              INX
783E- 8E C3 81      0840              STX DLOC+451
7841- E8            0850              INX
7842- 8E 13 82      0860              STX DLOC+531
7845- E8            0870              INX
7846- 8E 63 82      0880              STX DLOC+611
```

```
7849- E8          0890              INX
784A- 8E B3 82    0900              STX DLOC+691
784D- E8          0910              INX
784E- 8E 03 83    0920              STX DLOC+771          ;end of screen labeling
                  0930
                  0940 ; ++++++++++++++ Time measurements ++++++++++++++
                  0950
7851- A9 FF       0960              LDA #$FF              ;load all ones to A reg
7853- 8D 41 79    0970              STA LASTPA            ;set last port A value
7856- A9 00       0980              LDA #0                ;Clear A register
7858- A2 20       0990              LDX #32               ;Count number
785A- 9D 42 79    1000 CLTIME       STA POS,X             ;clear time counters
785D- CA          1010              DEX                   ;Decrement counter
785E- 10 FA       1020              BPL CLTIME            ;Clear all counters
7860- A9 02       1030 STARTTST     LDA #2                ;Test for a logic one
7862- 2D 4D E8    1040              AND PIA.IFR           ;at CA1 input
7865- F0 F9       1050              BEQ STARTTST          ;if not try again
7867- A9 3C       1060              LDA #%00111100        ;set one millisecond clock
7869- 8D 4A E8    1070              STA PIA.SHR           ;set shift reg. free run
786C- AD 41 E8    1080 RDPORTA      LDA PIA.DRA           ;Read input port A
786F- 8D 40 79    1090              STA PORTA             ;Save on port A loc.
7872- CD 41 79    1100              CMP LASTPA            ;did any car finished?
7875- F0 1C       1110              BEQ INCTIME           ;if no cars finished
7877- 4D 41 79    1120              EOR LASTPA            ;Find which car finished
787A- A2 07       1130              LDX #7                ;set index counter
787C- 0A          1140 FINDCAR      ASL A                 ;shift left into carry
787D- 90 0B       1150              BCC CKNXTK            ;if not this car
787F- 48          1160              PHA                   ;Save shifted input data
7880- EE 4A 79    1170              INC POSCOUNT          ;increase position count
7883- AD 4A 79    1180              LDA POSCOUNT          ;Get count
7886- 9D 42 79    1190              STA POS,X             ;put into position slot
7889- 68          1200              PLA                   ;get shifted data again
788A- CA          1210 CKNXTK       DEX                   ;Check next track
788B- 10 EF       1220              BPL FINDCAR           ;if not last, do again
788D- AD 40 79    1230              LDA PORTA             ;Get input data back
7890- 8D 41 79    1240              STA LASTPA            ;Save data
7893- A2 07       1250 INCTIME      LDX #7                ;Load index to inc timers
7895- 0A          1260 CKACTRK      ASL A                 ;Check for active track
7896- 90 21       1270              BCC NEXTTK            ;if track not active
7898- 48          1280              PHA                   ;Save input data
7899- F8          1290              SED                   ;Set decimal add mode
789A- 18          1300              CLC                   ;Clear carry
789B- A9 04       1310              LDA #4                ;Load increment
789D- 7D 5B 79    1320              ADC TILSG,X           ;add to least sig digit
78A0- 9D 5B 79    1330              STA TILSG.X           :store new value
78A3- 90 12       1340              BCC NOCARRY           ;if no carry
78A5- A9 00       1350              LDA #0                ;Clear A register
78A7- 7D 53 79    1360              ADC TIMID,X           ;add carry to next digit
78AA- 9D 53 79    1370              STA TIMID,X           ;store new value
78AD- 90 08       1380              BCC NOCARRY           ;if no carry
78AF- A9 00       1390              LDA #0                ;Clear A reg. again
78B1- 7D 4B 79    1400              ADC TIMSG,X           ;add to most sig. digit
78B4- 9D 4B 79    1410              STA TIMSG,X           ;store new value
78B7- 68          1420 NOCARRY      PLA                   ;get input data
78B8- D8          1430              CLD                   ;end decimal mode
78B9- CA          1440 NEXTTK       DEX                   ;check next track
78BA- 10 D9       1450              BPL CKACTRK           ;check next active track
                  1460
78BC- A2 07       1470              LDX #7                ;start display routine
78BE- BD 75 79    1480 D1           LDA HIBYTE,X          ;Set up display
78C1- 8D 05 79    1490              STA D6+2              ;Load display adr hibyte
78C4- 8D DC 78    1500              STA D2+2              ;Load pos address hibyte
78C7- BD 7D 79    1510              LDA LOBYTE,X          ;set up low addres byte
78CA- 8D 04 79    1520              STA D6+1              ;store on display address
78CD- 8D DB 78    1530              STA D2+1              ;store on pos low address
78D0- BD 42 79    1540              LDA POS,X             ;display position
```

```
78D3- F0 08     1550            BEQ D3              ;if 0 do not display
78D5- 18        1560            CLC                 ;clear carry
78D6- 69 30     1570            ADC #$30            ;change number to ASCII
78D8- A0 0B     1580            LDY #11             ;display 11 loc. to right
78DA- 99 3F 79  1590 D2         STA VARLOC,Y        ;variable address
78DD- A0 07     1600 D3         LDY #7              ;display time
78DF- B9 83 77  1610 D4         LDA TIMLOC-2,Y      ;get time location
78E2- 8D E6 78  1620            STA D5+1            ;set LOBYTE
78E5- BD 4B 79  1630 D5         LDA TIMSG,X         ;variable address
78E8- 48        1640            PHA                 ;save data
78E9- 29 0F     1650            AND #$0F            ;mask HIBYTE
78EB- 20 00 79  1660   .        JSR DISPLAY         ;put digit on screen
78EE- 68        1670            PLA                 ;get HIBYTE
78EF- 4A        1680            LSR A               ;shift right
78F0- 4A        1690            LSR A               ;again
78F1- 4A        1700            LSR A               ;again
78F2- 4A        1710            LSR A               ;four times
78F3- 20 00 79  1720            JSR DISPLAY         ;display next digit
78F6- C0 00     1730            CPY #0              ;check for last digit
78F8- D0 E5     1740            BNE D4              ;if not do again
78FA- CA        1750            DEX                 ;do next track
78FB- 10 C1     1760            BPL D1              ;do all eight tracks
78FD- 4C 10 79  1770            JMP CKEND           ;go to end or next pulse
                1780
7900- 18        1790 DISPLAY    CLC                 ;display digit on screen
7901- 69 30     1800            ADC #$30            ;convert to ASCII
7903- 99 3F 79  1810 D6         STA VARLOC,Y        ;variable address
7906- 88        1820            DEY                 ;set Y for next digit
7907- C0 04     1830            CPY #4              ;if Y=4 then put "."
7909- D0 04     1840            BNE D7              ;if not 4 then continue
790B- A9 2E     1850            LDA #$2E            ;ASCII code for "."
790D- D0 F4     1860            BNE D6              ;go to print decimal point
790F- 60        1870 D7         RTS                 ;end of display subroutine
                1880
7910- AD 40 79  1890 CKEND      LDA PORTA           ;get port A data
7913- 09 00     1900            ORA #0              ;check for end
7915- F0 17     1910            BEQ RETBAS          ;Return to BASIC if 0
7917- AD 4F E8  1920 PULSE      LDA PIA.PPA         ;get parallel port data
791A- 09 00     1930            ORA #0              ;check for end
791C- D0 06     1940            BNE TSTPLS          ;go test for pulse
791E- 8D 40 79  1950            STA PORTA           ;load 0 into PORTA
7921- 4C 72 78  1960            JMP RDPORTA+6       ;update counters
7924- A9 02     1970 TSTPLS     LDA #2              ;check for next time pulse
7926- 2D 4D E8  1980            AND PIA.IFR         ;at CA1 signal
7929- F0 EC     1990            BEQ PULSE           ;waait for inout oulse
792B- 4C 6C 78  2000            JMP RDPORTA         ;if pulse go read port A
792E- AD 4B E8  2010 RETBAS     LDA PIA.ACR         ;Turn off clock
7931- 29 EF     2020            AND #%11101111      ;and sound
7933- 8D 4B E8  2030            STA PIA.ACR         ;return reg. to normal
7936- AD 4C E8  2040            LDA PIA.PCR         ;also this register
7939- 29 FE     2050            AND #%11111110      ;must return to normal
793B- 8D 4C E8  2060            STA PIA.PCR         ;now
793E- 60        2070            RTS                 ;Return to BASIC
                2080
                2090 ;   ++++++++++ Data storage area ++++++++++
                2100
793F-           2110 VARLOC     .DS 1               ;Variable location storage
7940-           2120 PORTA      .DS 1               ;PORTA storage location
7941-           2130 LASTPA     .DS 1               ;LASTPA storage location
7942-           2140 POS        .DS 8               ;POS storage block
794A-           2150 POSCOUNT   .DS 1               ;POSCOUNT storage location
794B-           2160 TIMSG      .DS 8               ;TIME MSG byte sto. block
7953-           2170 TIMID      .DS 8               ;TIME MID byte sto. block
795B-           2180 TILSG      .DS 8               ;TIME LSG byte sto. block
7963- 54 52 41  2190 DDATA      .BY 'TRACK ' ' TIME    POS'
7966- 43 4B 20
```

```
7969- 20 54 49
796C- 4D 45 20
796F- 20 20 20
7972- 50 4F 53
7975- 80 81 81   2200 HIBYTE      .BY $80 $81 $81 $81 $82 $82 $82 $83
7978- 81 82 82
797B- 82 83
797D- D6 26 76   2210 LOBYTE      .BY $D6 $26 $76 $C6 $16 $66 $B6 $06
7980- C6 16 66
7983- B6 06
7985- 4B 4B 4B   2220 TIMLOC      .BY $4B $4B $4B $53 $53 $5B
7988- 53 53 5B
                 2230
                 2240 END.PRGM    .EN                    ;End program
END OF MAE PASS!
 ---  LABEL FILE:  ---
CKACTRK =7895          CKEND =7910          CKNXTK =788A
CLRSCRN =E229          CLTIME =785A         D1 =78BE
D2 =78DA               D3 =78DD             D4 =78DF
D5 =78E5               D6 =7903             D7 =790F
DDATA =7963            DISPLAY =7900        DLOC =8000
END.PRGM =798B         FINDCAR =787C        HIBYTE =7975
INCTIME =7893          LASTPA =7941         LDDATA =7827
LOBYTE =797D           NEXTTK =78B9         NOCARRY =78B7
PIA.ACR =E84B          PIA.DDA =E843        PIA.DRA =E841
PIA.IFR =E84D          PIA.PCR =E84C        PIA.PPA =E84F
PIA.SHR =E84A          PIA.TM2 =E848        PORTA =7940
POS =7942              POSCOUNT =794A       PULSE =7917
RDPORTA =786C          RETBAS =792E         STARTTST =7860
TILSG =795B            TIMID =7953          TIMLOC =7985
TIMSG =794B            TSTPLS =7924         VARLOC =793F
//0000,798B,798B
```



THE COMMODORE 64. ONLY $595.

What nobody else can give you at twice the price.

Circle #51 on the Reader Service Card

Microphys Programs

PET

VIC-20

2048 Ford Street
Brooklyn, New York 11229
(212) 646-0140

FOREIGN AND DOMESTIC DISTRIBUTOR INQUIRIES WELCOMED.

PET

VIC-20

Microphys, a leader in educational software development, is pleased to announce the release of several recreational software programs for use with the Commodore VIC-20 microcomputer. The VIC programs, described below, require a 3-K expansion cartridge and utilize the VIC's excellent color graphics and sound capabilities. Each program retails for $15 and is accompanied by complete instructions.

PROGRAM DESCRIPTIONS

PV901 - Missile Math: this program presents in a game format, an opportunity for youngsters (ages 5-15) to practice and develop the basic skills of addition, subtraction, multiplication, and division. Four levels of difficulty in each skill area may be selected. Problems in a given skill are randomly generated and missiles are launched at correct answers. The computer displays the results on each program run and may be directed to generate the same sequence of problems so that review and 'match play' (against an opponent) are possible.

PV350 - Cryptograms: this program permits the generation of 'secret' messages which are to be decoded. These cryptograms are displayed along with their unique code number classifications. To decode a cryptogram, the program is run from line 9000. Family members can challenge each other with their individually created messages. If you enjoy solving the cryptograms appearing in newspapers and crossword puzzle magazines, this program is perfect for you. Note: two VIC users may exchange encoded messages. User 1 creates a secret message and transmits this to user 2. The code number will permit user 2 to have **his** VIC decipher the message should he encounter any difficulty.

PV340-349 - Anagrams: this series of programs provides an educational challenge for virtually all age groups. The VIC randomly generates scrambled words which are to be identified. Two clues are provided in order to assist in this process. The clues in the school and college categories are generally definitive in nature. Many of the words used are part of the Microphys Spelling and Vocabulary series for the associated grade levels. Thus, reading, vocabulary, and spelling skills are reinforced by these Anagram programs. Note: the same sequence of words generated may be requested so that 'match play' is possible. There are 5 level-of-difficulty categories each consisting of two programs.

PV340-341 Recreational	PV342-343 College	PV344-345 High School
PV346-347 Junior High	PV348-349 Elementary	

PV375-380 - Wheel-of-Fortune Word Games: this series of programs represents an exciting challenge for every member of the family. Players try to fill in missing letters in a randomly generated title or phrase and earn and lose points according to the graphic display on a 'Wheel-of-Fortune'. The scores of as many as four players are displayed, 1000 points being required to win a given game.

PV375 Song Titles	PV376 Famous Places	PV377 Entertainers
PV378 Statesmen	PV379 Scientists	PV380 Sports Figures

PV601-644 - Missile Spelling: this series of 36 programs enables youngsters in grades 4 through 12 to practice and develop basic spelling skills. Each program contains 60 graded words. The VIC randomly selects groups of 5 words, one of which is spelled incorrectly. Missiles are launched in order to destroy the word misspelled. The words chosen for grades 7 - 12 correspond to the Microphys Vocabulary series. Note: there are 4 programs in each grade level.

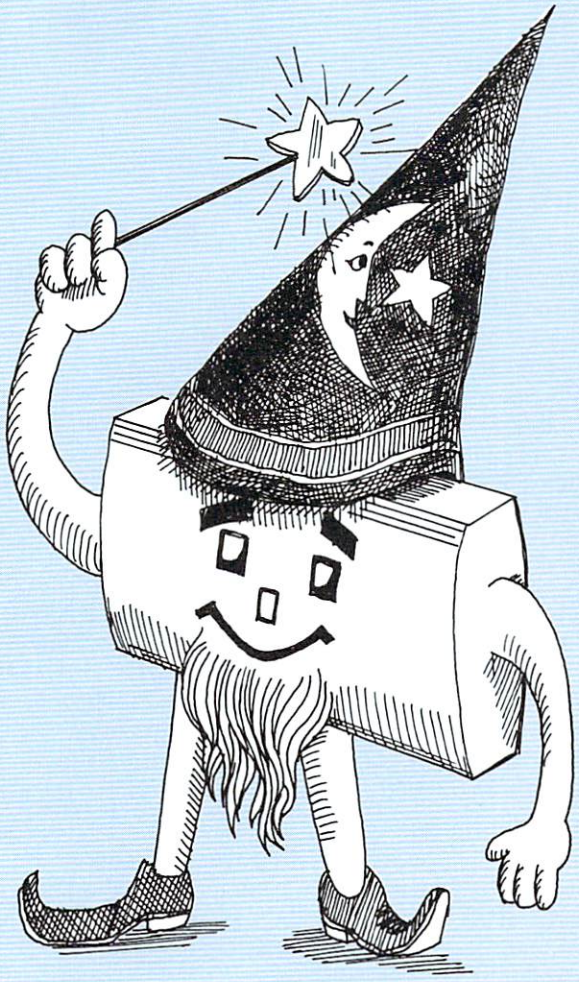
PV601-604 Grade 12	PV606-609 Grade 11	PV611-614 Grade 10
PV616-619 Grade 9	PV621-624 Grade 8	PV626-629 Grade 7
PV631-634 Grade 6	PV636-639 Grade 5	PV641-644 Grade 4

PV401-460 - Vocabulary: each vocabulary program randomly generates graded words which are to be defined. A sentence, in which the word is properly used, is displayed when an incorrect response is made. Using this contextual clue, a second opportunity to define the word is given. Reading and spelling skills are also reinforced as a more powerful vocabulary is developed. There are 10 programs in each grade level.

PV401-405 and PV431-435 Grade 12	PV406-410 and PV436-440 Grade 11
PV411-415 and PV441-445 Grade 10	PV416-420 and PV446-450 Grade 9
PV421-425 and PV451-455 Grade 8	PV426-430 and PV456-460 Grade 7

Educators should write for the new Microphys Fall Catalog which describes over 200 programs for use in Chemistry, Physics, Calculus, Mathematics, Vocabulary, and Spelling classes on both the high school and college levels.

VIC-20
The friendly computer



The VIC Magician

Working With Random Numbers

by
Michael S. Tomczyk
Product Marketing Manager

Your VIC 20 has a built-in ability to generate RANDOM NUMBERS using a special command called RND. But what's a random number? The best way to show you how random numbers work is to give you an example that DOESN'T use a computer.

To see how random numbers work, try this: Take 10 pieces of paper and write a number from 1 to 10 on each piece. Next, put the 10 pieces of paper into a hat or other container where you can't see them. Now, cover your eyes and draw out one piece of paper. What number did you get? That number is a RANDOM number! Now put the number BACK IN THE HAT, mix up the papers and draw again.

Each time you draw a number, put the number back in the hat so there are always 10 pieces of paper to choose from. Keeping 10 numbers in the hat means you always have 10 RANDOM NUMBERS in the hat. When you take a number, you DON'T know which number is going to come next, but you DO know that the number will be between 1 and 10. This is the basis for RANDOM NUMBERS.

In programming, random numbers usually have a RANGE. This means there's an UPPER LIMIT and a LOWER LIMIT to the numbers you can draw. In our "hat" example, the range of numbers is 1 to 10. The lower limit is "1" and the upper limit is "10" . . . which means . . . ANY NUMBER FROM 1 TO 10 CAN COME UP "AT RANDOM" WHEN YOU DRAW.

Now let's see how a computer handles random numbers. Here's a program which generates 5 completely random numbers:

```
10 FORX=1T05:PRINTRND(X):NEXT
```

This program tells the computer to pick 5 numbers "out of the air"—in other words, "random numbers"—and PRINT them. But these numbers all have decimal points (a number with "decimals" is 5.532 . . . a number without decimals is just 5 . . . the numbers to the RIGHT of the decimal point are called "decimals"). Most uses for random numbers require WHOLE NUMBERS without any decimal places.

You can make your random numbers always come out as whole numbers by using the INT function, which cuts off any decimal places. To see how INT works, type this: PRINTINT(5.236) and press the RETURN key. The computer will cut off the numbers on the right side of the decimal point and display only the "whole number" on the left side of the decimal point, which is 5.)

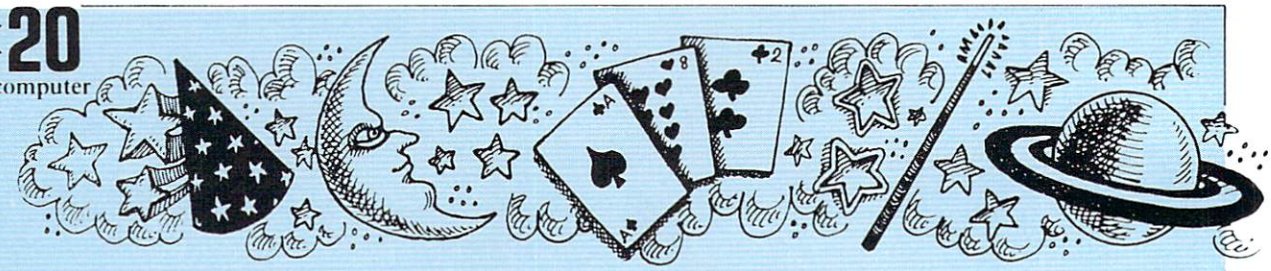
A Random Number Formula

Here's a "formula" which creates a RANDOM NUMBER within some RANGE of numbers that you define:

```
INT (RANGE * RND(1)) + LOWER LIMIT
```

The INT command tells the computer to cut off any decimal places and only give you whole numbers like 1, 45, or 320 instead of numbers like 1.223, 45.6677, or 320.59. Whole numbers are easier to work with when using random numbers.

LOWER LIMIT means the lowest number you want the computer to choose from, and **RANGE** means how many numbers are in the total group. For example, if you want to choose a random number from 1 to 5, the LOWER LIMIT will be 1 and the RANGE will be 5. If you want to choose a random number from 15 to 20, the LOWER LIMIT will be 15 and the upper limit will still be 5 because you are still choosing from a total group of 5 numbers. If you're choosing numbers from 2 to 100, the LOWER LIMIT will be 2 and the RANGE will be 98. See how it works? ▶



PROGRAMMING TIP: To get a plus sign (+) just type the plus sign on the VIC keyboard . . . DON'T SHIFT THE PLUS SIGN. If you hold down the SHIFT key and type the PLUS key, you'll get a graphic symbol that LOOKS LIKE a plus sign, but really isn't. The plus sign is typed WITHOUT SHIFTING. The graphic criss-cross symbol is typed while holding down the shift key.

Let's use the formula to ask the VIC to choose one number at random between 1 and 5. This is a lot like shuffling 5 identical cards numbered 1 to 5 and laying them face down on a table. If you can only choose one card, which card will you get? The result will be random. Try this program (be sure to "balance" your parentheses by putting an equal number of left and right parentheses . . . there should be FOUR parentheses, two left and two right, as shown):

```
10 PRINTINT(5*RND(1))+1
```

RUN this program three or four times (keep typing RUN and pressing the RETURN key). The VIC20 keeps PRINTing a number from 1 to 5. The numbers are PRINTed randomly, in no special order. First you might get a 5, then a 3, but whatever you get, the number is ALWAYS one of these numbers: 1,2,3,4,5.

Multiple Random Numbers

Let's use a FOR . . . NEXT loop to tell the VIC20 to choose TEN random numbers from 1 to 100. We'll use the same formula, except now the RANGE is 100 and the LOWER LIMIT is 1. Type this line and RUN it:

```
10 FORX=1T010:PRINTINT(100*RND(1))+1:NEXT
```

The FOR . . . NEXT loop tells the VIC20 to display 10 numbers chosen at random from between 1 and 100. Yes, but what PRACTICAL use is this? Well, what if we chose a range from 0 to 15 . . . the numbers used to change the color of the screen?

Flipping a Coin

Here's another program that has the computer "flip" a coin 15 times . . . notice that in Line 20 we make the variable A equal to a random number represented by our entire formula which has 1 as the lower limit and 2 as the upper limit. In other words, the program "flips" between two numbers at random, just like a coin toss flips between 2 sides of a coin. The RANGE can be ANY TWO NUMBERS as long as they're consecutive (in order). For example, you could change 1 and 2 to 15 and 16, 200 and 201, etc. RUN this program several times to see how your imaginary "coin" randomly comes up heads or tails, with no real pattern:

```
10 FORN=1TO20
20 A=INT(2*RND(1))+1
30 IFA=1THENPRINT"HEADS"
40 IFA=2THENPRINT"TAILS"
50 NEXT
```

LINE 10 is a FOR . . . NEXT "counting loop" which tells the VIC how many times to perform the action . . . in this case, flip the coin. Note that every program action between the FOR . . . section in Line 10 and the NEXT command in Line 50 will be performed. That's how you use a FOR . . . NEXT loop. Put the actions you want to accomplish between the FOR and NEXT. (When there is nothing between the FOR and NEXT parts of the loop, the VIC interprets it as a "time delay" loop).

LINE 20 is the random number formula. Since we're only flipping a coin here, we only need two choices. We will tell the computer to choose either a 1 or a 2 and then arbitrarily assign "heads" to one number and "tails" to the other. The random number is generated AUTOMATICALLY. Just put the formula in your program and when the VIC reaches the formula, it automatically assigns a random number to the variable A, from the range you've defined.

LINES 30 and 40 use IF . . . THEN statements to set up conditions which determine HEADS or TAILS. Every time the VIC generates the random number called "A" the IF . . . THEN statements PRINT the word "HEADS" if the random number is 1 and "TAILS" if the random number is 2.

LINE 50 contains the NEXT command which matches the FOR . . . statement set up in Line 10. The result is that the ENTIRE PROGRAM occurs 20 times, including the random number being generated, and either a "HEADS" or "TAILS" message printed.

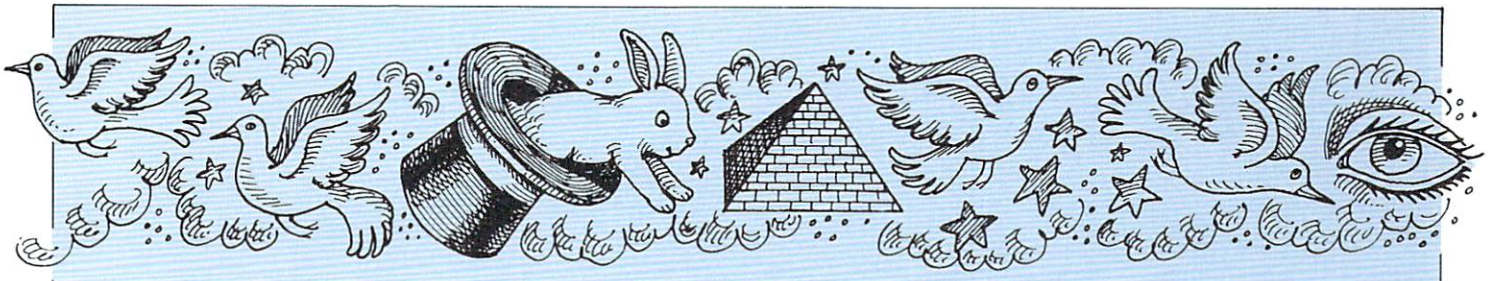
Random Music

Now let's use the VIC's musical capabilities to generate some "random music." Did you ever hear that strange computer music they sometimes use on TV shows about computers? You know . . . those "bleep . . . bloop . . . bleep" sounds? Well, here's some "bleep . . . bloop . . . bleep" music using random numbers:

```
10 V=36878:S1=36875:POKEV,15
20 N=INT(50*RND(1))+200
30 POKES1,N
40 FORT=1TO50:NEXT
50 POKES1,0:GOTO20
```

Hold down the RUN/STOP key and press the RESTORE key to stop the program.

LINE 10 defines V (for volume) and S1 for the speaker we want to use. Then we POKEV,15 to set the volume at its highest setting. See the SOUND AND MUSIC section of your user's manual for more information.



LINE 20 is our random number formula. Here, we're going to use the formula to choose which MUSICAL NOTE to play. The LOWER LIMIT is 200 and the RANGE is 50. This means the VIC will play musical note values at random from 200 to 250 (from the table on Page 135 of your user's guide). Try changing 200 to 150 for a lower range of notes.

LINE 30 plays a musical note by POKEing "speaker 1" with a musical note value (N) which comes from our random number formula in Line 20.

LINE 40 is a time delay loop which tells the VIC to keep playing the musical note for a count of 50 (about the duration of a 32nd note).

LINE 50 turns off the speaker by POKEing it with a value of 0. Then the program goes back to Line 20 to get another random musical note.

If you want an interesting variation, try PRINTing some characters as the notes are being played. Try adding this line:

```
35 PRINT"BLEEP. . .";
```

don't forget the semicolon (;)

Next, let's try using random numbers to PRINT a "crazy quilt" of colored symbols on the screen . . .

Crazy Quilt Screen Colors

The next example combines the RND command with the VIC's color capabilities . . . but first, let's write a program that DOESN'T use RND. This program fills the screen with the "#" symbol, in "crazy quilt" colors . . . but the colors appear IN ORDER, one at a time. Type the word NEW and press the RETURN key to erase your last program, then enter and RUN this program:

```
10 X=1
20 C$=" CTRL BLK CTRL WHT CTRL RED CTRL
   CYN CTRL PUR CTRL BLUE CTRL YEL"
30 PRINTMID$(C$,X,1)"#";
40 X=X+1
50 IFX=9THENGOTO10
60 GOTO30
```

Hold down the CTRL key and type the color keys shown

(Press RUN/STOP to STOP the program)

Like your crazy quilt? This crazy quilt isn't really crazy because it has a PATTERN, which means it isn't RANDOM. The program PRINTs a black symbol (0 = black) then a white symbol (1 = white) and so on up to 8 colors, then it REPEATS those colors in the same order, over and over again. Here's a quick explanation of how the program works:

LINE 10 defines the numeric variable X (it could be any letter) as the number 1.

LINE 20 defines the string variable C$ as the 8 "VIC color commands" which you typed in between the quotation marks (remember you can PRINT color and editing commands just like you PRINT letters, numbers and graphics. Notice that the VIC displays REVERSE GRAPHIC CHARACTERS on the screen when you hold down CTRL key and type a color key. This is a special VIC feature that shows you where color commands are in your program).

LINE 30 uses the MID$ function, a special BASIC command that lets the computer select any character from a "string" of characters in quotation marks, by COUNTING over the left side of the string. In our example, **PRINTMID$ (C$,X,1)** tells the VIC to PRINT a character from the "string" of color commands.

C$ identifies which string we are using (in case we have several strings in our program).

X means count X spaces over from the left and use that color command. At this point X equals 1 so the VIC takes the first character in the string, which is CTRL BLK, (the BLACK color command).

The number 1 in parentheses means do it one time.

The "#" symbol in Line 30 could be any symbol . . . try using a different graphic symbol, if you like.

LINE 40 changes X so that X is now 1+1, or 2. Now when the program goes back to Line 30, X will be 2 and the SECOND color command (WHITE) will be used.

LINE 50 only lets X go up to the number 8. If it reaches 9, the VIC resets X to 1 by going back to Line 10 (X = 1) and starting over. The effect is to PRINT the "#" mark in eight colors, then go back and do it over again.

Random Colors

Let's use the same program, but instead of displaying the "#" in the same 8 colors over and over, we'll display the symbol using RANDOM COLORS.In other words, the computer will mix up the colors without any special order, just like drawing the colors out of a hat.

Edit your program so it looks like this (if you're a beginning programmer, it might be easiest to type the word NEW, hit the RETURN key, then retype the program as shown):

```
10 X=INT(8*RND(1))+1
20 C$=" CTRL BLK CTRL WHT CTRL RED
```

```
CTRL CYN CTRL PUR CTRL BLUE CTRL YEL"
30 PRINTMID$(C$,X,1)"#";
40 GOTO10
```

Type RUN and hit the RETURN key to see the program. (Press RUN/STOP to STOP the program)

Now your crazy quilt makes RANDOM colors . . . the graphic symbols don't keep repeating the same order. Instead, they have an unpredictable pattern which really makes the quilt crazy! Here's what we did to change the program:

LINE 10 is the standard random formula, with a lower limit of 1 and a range of 8.

LINE 20 is the same as before. (For an interesting effect, try using 2 or more graphic symbols inside quotation marks instead of just the single "#" symbol)

LINE 30 is also the same.

LINE 40 makes the program go back to Line 10, where X becomes a new random number. The program keeps going back, using a new different random number (and color) each time.

Like those colors? The LOWER LIMIT is 1 and the RANGE is 8. Just for fun, try changing the lower limit to 3 . . . this eliminates the first two colors (black and white) and only PRINTs the colors from the third color in the string to the 8th color. If you want to eliminate the color white from the group (which only shows up on the screen as a blank square) you can eliminate the "WHITE" setting from Line 20 and reduce the RANGE to 7 in Line 10.

RANDOM Guessing Game

Here's a little guessing game which asks you to guess a number generated randomly by the computer, and tells you if you're too high, too low, or if you guess right . . . you might want to try adding your own sound effects, colors, graphics or other cosmetic touches:

```
10 N=INT(10*RND(1))+1
20 PRINT" CLR/HOME GUESS A NUMBER FROM
   1 TO 10 AND PRESS RETURN":INPUTA
30 IFA>NTHENPRINT" CLR/HOME TOO
   HIGH!":FORT=1TO700:NEXT:GOTO20
40 IFA<NTHENPRINT" CLR/HOME TOO LOW!":
   FORT=1TO700:NEXT:GOTO20
50 IFA=NTHENPRINT" CLR/HOME RIGHT!"
60 POKE36878,15:FORF=150TO250STEP2:
   POKE36876,F:NEXT:POKE36876,0
70 FORT=1TO700:NEXT:GOTO10
```

LINE 10 is the random number formula. Here we're guessing any number from 1 to 10 so the lower limit is 1 and the range is 10.

LINE 20 clears the screen, then PRINTs the opening "prompt" message that tells you what to do. INPUT A displays a question mark and waits for you to enter a number . . . then defines that number as the "numeric variable" A.

LINE 30 tells the VIC that if the number you typed in (called A) is larger than the random number the computer generated (called N), then PRINT the "Too High!" message, count to 700 (the FOR . . . NEXT loop) and go to Line 20 to repeat the message.

LINE 40 works just like Line 30, except here if your Answer (A) is less than the random Number (N) the VIC generated, the message "Too Low!" is PRINTed before going back to Line 20 to repeat the message.

LINE 50 determines whether the Answer (A) equals the random number (N) and if it does, it PRINTs the "RIGHT!" message and lets the program drop through to the next line . . .

LINE 60 sets the volume to its highest setting (POKE36878,15), then uses a FOR . . . NEXT loop to set up a short whooping sound effect. FORF=150TO250 STEP2 moves the VIC's sound generator through a series of musical note values from 150 (low) to 250 (high) 2 notes at a time (STEP2). When the program hits POKE36876,F, the VIC POKEs speaker number 36876 (see your user's guide) with the first note value (150). The NEXT command makes the VIC POKE the speaker with the next number in the FOR . . . NEXT loop which is 152, then 154, and so on up to 250. This creates the whooping sound effect. Finally, we turn off the speaker by POKEing it with a zero (POKE36876,0).

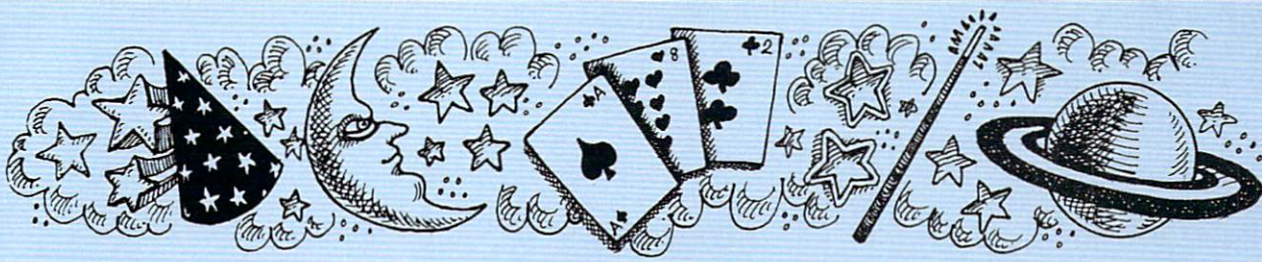
LINE 70 includes a time delay loop which keeps the "RIGHT!" message on the screen for a short duration before going back to choose another random number. Notice that here we use GOTO10 because we want to have the computer generate a new random number to guess. If you said GOTO20 here, the VIC would give you the "GUESS" message but use the same number you just used.

PROGRAMMING NOTE: Using IF . . . THEN Statements

When using IF . . . THEN statements, you can put MORE THAN ONE ACTION after the THEN portion and let the program "fall through" to the additional actions contained in the lines that follow. Here, the program is set up so it branches back to an earlier line in the program when it hits the first two IF . . . THEN statements in Lines 30 and 40. But when it hits Line 50, it CONTINUES BEYOND that line and performs the actions on the lines that follow (i.e. musical sound effect, time delay) until the program reaches the end of Line 70 which tells it to GOTO Line 10 and start over.

Using IF . . . THEN statements this way can help you set up whole series of actions (graphics, sound effects, etc.). The structure works like this (using imaginary program lines):

```
100 IF (condition) THEN (first action)
110 (More actions)
120 (More actions)
130 GOTO or GOSUB (line number)
140 (Next IF . . . THEN statement or continue program)
```



Writing Games in BASIC

Part 1 . . . Animation & Random Numbers

by
Michael S. Tomczyk

Where do you learn how to write game programs? There aren't any schools you can go to. The few books on the subject are written in computerese (an obscure dialect understood only by engineers and wizards).

This 3-part series will try to fill a tiny part of the void of gamewriting information, by giving you some interesting and creative programming techniques you can use to write games.

In Part I, we'll use the VIC's random number feature to animate VIC graphic symbols so they move randomly, change color randomly or make random sounds.

In Parts II and III, we'll explore ways to move objects across the screen and design a simple game . . . then we'll design a game using your own PROGRAMMABLE CHARACTERS.

A Quick Animation Lesson

Here's a quick list of the steps we are going to follow . . . in order . . . in the animation programs we'll be explaining:

1. **Clear the screen at the beginning of the animation.**
2. **Set up a random number formula.**
3. **Display a symbol on the screen by POKEing a SCREEN MEMORY LOCATION with the POKE NUMBER of the symbol we're using.**
4. **Match the SCREEN MEMORY LOCATION of the symbol we're using with a COLOR MEMORY LOCATION. Every time you POKE a symbol into a different location, you have to POKE a matching color memory location.**
5. **Don't forget to keep ERASING the symbol as you move it. You do this by POKEing a blank space (POKE NUMBER 32) into the symbol's previous screen memory location.**
6. **Insert sound effects as appropriate.**

We'll begin by CLEARing the screen. Type this line and press RETURN:

10 PRINT" SHIFT CLR/HOME "

Now . . . turn to Page 144 in your VIC user's guide ("Personal Computing On the VIC20"). The SCREEN CHARACTER CODES and COLOR CODES MEMORY MAP represent all the locations on your television screen where you can place graphic symbols (including letters and numbers). Note that the TOP grid is used to position the symbol, and the BOTTOM grid is used to set the color at that position. So if you POKE a symbol into location 7680 (the top lefthand corner of your screen), you ALSO have to POKE a color setting into location 38400 which matches it.

For our example, we're going to use the heart on the "S" key on your keyboard. We begin by turning to Page 142 and finding the POKE NUMBER of the heart. From the chart, you can see that the number for the heart is 83.

Now let's find a SCREEN LOCATION where we want to put the heart. How about the top lefthand corner? The grid on Page 144 tells us the LOCATION is 7680. So our first line number includes the command POKE 7680,83. Try it:

20 POKE7680,83

Type RUN and press RETURN. Nothing happens! That's because you still haven't set the COLOR MEMORY LOCATION! Remember we said you have to set the matching color location EVERY TIME you POKE or MOVE a symbol on the screen? Type the word LIST and press RETURN.

Now look at Page 144. The COLOR MEMORY LOCATION which matches 7680 is 38400. Next you need to choose a color from the ones listed on Page 143. We'll choose green. The number of green is 5 (one less than the numbers on the keyboard color keys). So we have to POKE 38400,5 to make our heart turn green. Type this:

30 POKE38400,5

NOW type RUN and press RETURN! Presto, you have a solid green heart in the top lefthand corner of your screen!

For our first "random animation" example, we're going to POKE a row of green hearts at the top of the screen; but first we'll do it WITHOUT using random numbers. Type NEW to erase your previous program and enter this program:

```
10 PRINT" SHIFT CLR/HOME "
20 FORB=7680TO7701
30 FORBC=38400TO38421
40 POKEB,83:POKEBC,5
50 NEXTBC:NEXTB
```

RUN this program. It runs PAINFULLY SLOW! How can you animate anything moving this slowly? Well, the only reason it moves so slowly is because some of the programming elements slowed it down . . . we'll speed it up by rearranging things. Type NEW and retype your program like this:

```
10 PRINT" SHIFT CLR/HOME "
20 FORBC=38400TO38421:POKEBC,5:NEXT
30 FORB=7680T07701
40 POKEB,83:NEXT
```

Now RUN it! A lot faster, right? That's because we took some shortcuts. The following comments will explain what we did:

LINE 10 clears the screen.

LINE 20 contains the secret of why the second program RUNs faster. The second version uses a FOR . . . NEXT loop to POKE the entire top row of COLOR memory locations BEFORE we poke the symbol into the screen memory locations. The previous program POKEd one color memory location, then one screen memory location, then repeated the process one location at a time, which was too slow.

LINE 30 sets up a FOR . . . NEXT loop for the SCREEN MEMORY locations of our heart.

LINE 40 uses the POKE NUMBER of the heart (83) to POKE hearts into the top row of screen memory locations (as defined by the FOR . . . setup in Line 30). POKEB,83 actually means: POKE7680,83 and the NEXT command in this line results in POKE7683,83 . . . POKE7682,83 . . . and so on until the heart has been POKEd into the top row of memory locations. The hearts appear fast here because all the color memory locations have already been set.

The Heartmaking Program

This program POKEs a row of hearts across the top of the screen . . . RANDOMLY! First it POKEs a red heart, then "counts to 800" and POKEs a green heart . . . and keeps going until the top row is filled with green hearts. After the row is filled, it keeps on POKEing hearts. This is your first "random animation." Type NEW and hit the RETURN key to erase your previous program and enter this program and RUN it:

```
10 PRINT" SHIFT CLR/HOME "
20 L=INT(21*RND(1))+7680
30 POKEL,83:POKEL+30720,2
40 FORT=1TO800:NEXT
50 POKEL+30720,5
60 GOTO20
```

LINE 10 clears the screen.

LINE 20 is a random number formula which sets the LOWER LIMIT at 7680 which is the upper left screen location. It sets the RANGE at 21 . . . which means the formula will generate random numbers from 7680 to 7701, which is the TOP ROW of screen locations.

LINE 30 puts a heart at the random screen location generated by Line 20, by POKEing that location with the value of the heart, which is 83. Then it colors the heart RED by POKEing the matching color location with the number 2 (red). There's an important programming tip here. We use L as the variable to define the constantly changing screen memory location, and we use L + 30720 as the matching color setting. You can match the color setting to ANY SCREEN LOCATION simply by adding 30720 to the screen location number. For example, if the first screen location is 7680, the color location is 7680 + 30720, which is 38400 . . . a quick check of the grid in your user's manual confirms this.

LINE 40 is a time delay loop which has the VIC "count" to 800 before changing the red heart to green (see Line 50).

LINE 50 changes the color of the heart we just POKEd to green. We're doing this because we want to show you the heart when it is first randomly POKEd onto the screen, and the easiest way to do this is to color it red, then change the color. Notice that the red hearts keep appearing even after the entire top row is filled with green hearts, because the program keeps going back to put more random hearts in the top row. Sometimes before the row is completely filled, a red heart appears on top of a green heart that's already there . . . again, because the hearts are being placed on the screen randomly, and can appear anywhere in the range of locations from 7680 to 7701.

You can add RANDOM SOUND to your heartmaking program by adding these 2 lines to your program:

```
35 POKE36878,15:POKE36876,INT(100*RND(1))
   +150
45 POKE36876,0
```

LINE 35 POKEs the volume setting (36878) to its highest level (15). Then it turns on speaker number 3 (36876) by POKEing that number with a random number whose LOWER LIMIT is 150 and whose range is 100. This musical setting causes the VIC to POKE music location 36876 with ONE musical note value from 150 to 250. Normally a time delay would be required here to hold the note for some DURATION, but by putting this line before the time delay in Line 40, we can use the delay in Line 40 as our note duration.

LINE 45 turns off the speaker after it has played the note. If you didn't turn off the speaker here it would keep playing.

The result of this program is to create a random musical note every time a new heart is POKEd onto the screen. This creates a random "bleeping" sound which is very "computerese" and makes a nice accompaniment to our random hearts.

A Screen Full of Hearts

This program puts hearts across the ENTIRE SCREEN (506 different locations) and uses the same principles used in the previous programs, as well as a new RANDOM COLOR feature. Here's the program:

```
10 PRINT" SHIFT CLR/HOME "
20 L=INT(506*RND(1))+7680
30 POKEL,83
35 C=INT(7*RND(1))+0
40 POKEL+30720,C
45 POKE36878,15:POKE36876,INT(50*RND(1))+150
50 FORT=1TO100:NEXT
55 POKE36876,0
60 GOTO20
```

Most of the elements are similar to those we used in the previous programs, so we'll just explain the parts that are different:

LINE 20 changes the locations from a range of 7680 to 7701 (the top row) to a range of 7680 to 8186 (the full screen, 506 locations). The LOWER LIMIT is 7680 and the RANGE is 506.

LINE 30 puts a heart at the random location determined by Line 20 by POKEing that location (L) with the heart POKE NUMBER (83).

LINE 35 sets up RANDOM COLOR by setting the LOWER LIMIT to 0 and the RANGE to 7 . . . these are the color settings you have to POKE to make your hearts turn color. We have defined the variable "C" as this random color setting, so C will always be a color setting number from 0 to 7 and will be randomly generated.

LINE 40 POKEs the color location with the random value of C.

LINE 45 generates a random musical note each time a heart is displayed. The LOWER LIMIT is 150 and the RANGE is 50 so the musical note values generated will be from 150 to 200.

LINE 50 has a faster time delay than the previous program.

LINE 55 turns off the musical "speaker."

LINE 60 goes back to Line 20 to get another random location where the next heart will be POKEd.

Summary

This brief introduction should give you some programming techniques to start experimenting with. Our next issue will contain more gamewriting techniques and . . . a real game! In the meantime, keep experimenting and see what you can come up with. The VIC's sound and graphics capabilities will surprise you when you start using them to write games.

Finally . . . a plug for Commodore . . . one of our best gamewriting tools is an inexpensive program on tape called the "Programmable Character Set/Gamegraphics Editor." The program comes with a detailed booklet that shows you how to create your own programmable characters and use them in your program. It's one of Commodore's best programming bargains . . . see your Commodore dealer for more information.

Meet the VIC 20

Part 2: What Comes In the Box

by
Neil Harris

One of the beauties of the VIC 20 is that you can use it as it comes out of the box, without buying any extras. The package includes the VIC keyboard, containing all the computer circuits and "brains" of the machine. You also get everything you need to connect to any TV set and to the plug in the wall, as well as the now-famous "Friendly User Manual," warranty information, and magazine subscription forms.

The TV hookup takes only a couple of minutes, and the only tool needed is a screwdriver.

Three components are included for TV hookup. The RF modulator takes the audio and video signals from the VIC and mixes them together to form a standard TV signal. A switch on the modulator lets you select to use either channel 3 or 4, whichever is free in your area.

Next there is a short cable with an RCA plug on both ends. This cable

plugs into the jack on the modulator. The other end plugs into the third piece, the TV switchbox. You can get a longer cable if you want to move the VIC farther from the TV. Go to any audio or video store and ask for a 75 ohm shielded cable with RCA male plugs.

The switchbox is the same as those used for video games. It has plugs for the cable from the modulator and for the antenna from your TV, and a connector to the VHF terminals on the back of your TV. This lets you use the TV and the VIC without having to disconnect wires each time.

Some people will want to use the VIC with a video monitor instead of a TV. The VIC has standard audio and video signals coming out the port in the back. To hook up to a monitor you won't use any of the three components used for normal TVs. Instead, you'll need a monitor cable. These can be purchased at some audio stores, like Sam Goody. Ask for an audio cable with a 5-pin DIN plug on one end and RCA male plugs on the other. Most monitors use RCA plugs for audio and video inputs, but you can also buy adapters from RCA to any other type of connector needed.

You can also build your own cable by following the pinout diagram in the back of the VIC manual. Just take the proper video signal, either 75 ohm (video low) or 300 ohm (video high), along with the audio signal. Hook up the ground connection to both audio and video, and you're all set.

The VIC connects to the wall with the power supply, a rugged black box with a VIC plug on one end and a wall plug on the other. This power supply always draws some power, even when the VIC is turned off, so it should be kept unplugged when not being used.

Once the TV and power connections are made, you're in business. When you turn on the VIC, you can begin computing. More on this in the next installment.

The warranty card registers you as a VIC owner. Sending it in will get you new product announcements and other important news from Commodore. Your sales receipt is what really warrants your machine, so you should hang on to it and send back the warranty card as soon as possible.

The warranty period on the VIC is the industry-standard 90 days. If there is a problem with your computer within the warranty period, just bring it back to where you bought it and they'll give you a brand new one. After the 90 days, you should send your machine directly to Commodore, either in Santa Clara, California or in King of Prussia, Pennsylvania. There is a flat rate repair charge for VICs of $55. Within 48 hours of receipt, a factory reconditioned VIC is on its way back to you, complete with a new 90 day warranty.

You get subscription forms both for this magazine and *Power/Play,* which is oriented toward the home applications and entertainment aspects of Commodore computers. The magazines are the best place to find out what's new in the world of VIC computing. Besides articles and programs, they have ads from many of the small companies that manufacture accessories for the VIC, from software to motherboards.

Finally, but very importantly, you get the book called *Personal Computing on the VIC 20,* subtitled "a friendly computer guide." The first thing this book tells you is how to hook up the computer, complete with a trouble-shooting chart that helps you if something doesn't seem to be working right. Then, before you have *any* idea what you're doing, you start typing in simple programs and working with the computer. You learn by doing. There are sections on the keyboard, color graphics, animation, sound, and beginning programming.

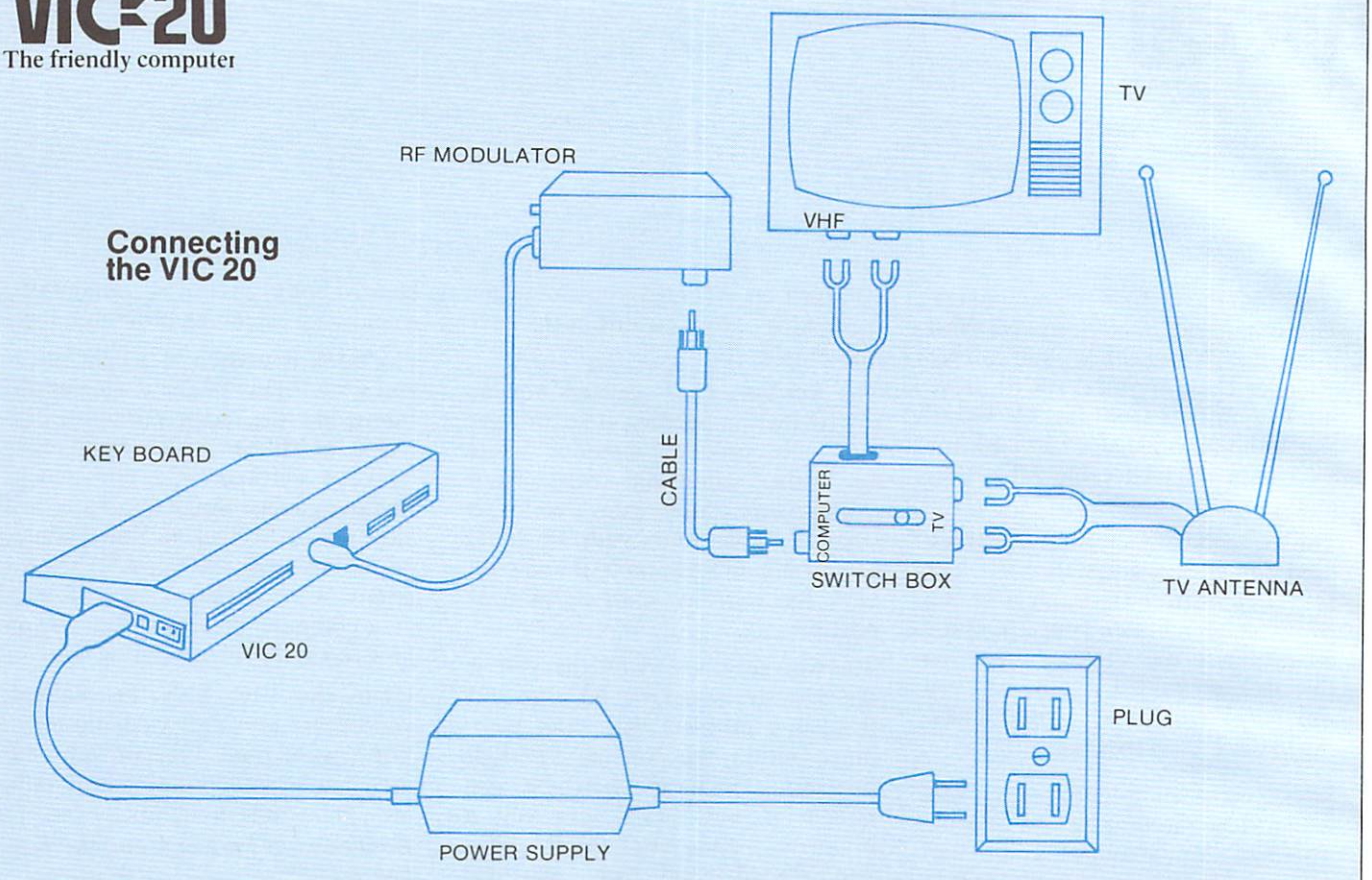
The book is written in a light informal style. Many of the examples are funny, like the recipe program. The recipe for Mike's Chicken Soup reads "Take 1 chicken. Kill and pluck, Remove giblets. Boil 4 qts. water in a large pot. Add chicken. Boil 2 hours, or until house smells good." The programs were designed to minimize your computer-anxiety and show you how fun and easy computing is.

Using the VIC with the manual, you'll learn elementary computing in just a short time. Like we say, children can begin computing in 15 minutes, and adults in slightly longer.

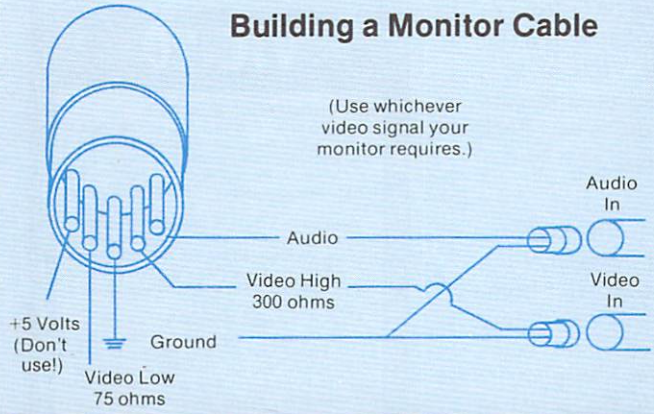
Continued in Part 3: Using the Keyboard

VIC-20
The friendly computer

Connecting the VIC 20



Building a Monitor Cable



Circle #27 on the Reader Service Card

Circle #29 on the Reader Service Card

Now . . . The Ultimate In Wordprocessing For The Commodore Computer. WORDCRAFT ULTRA!

Wordcraft ULTRA™ turns your Commodore microcomputer into one of the world's most advanced word processing systems, incorporating features previously found only in systems priced thousands of dollars higher!

- True proportional spacing with inter-character and inter-word spacing!
- Multi-user with up to 8 CPU's sharing one or more disk drives and printers with no extra hardware required other than a cable!
- Screen layout matches the printed document! You'll love the "what you see is what you get" feeling.
- Continuous centering, delete and insert, movement of text, search and replace, tab and indent, bold print and underline . . . a seemingly endless list of features designed to make text editing as simple and complete as possible!
- Wordcraft ULTRA runs on the CBM 8032, 8096 or Madison Computer's Z-RAM™ board. The Z-RAM Board not only expands your computer to 96K but also adds CP/M™.

Why settle for less when you get so much more with Wordcraft ULTRA on the Commodore Computer!

Contact Your Nearest Commodore Dealer Today . . .
You'll Be So Glad You Did!

In The East Call:
COMPUTER MARKETING SERVICES INC.
300 W. Marlton Pike, Suite 26
Cherry Hill, New Jersey 08002
(609) 795-9480

In The West Call:
CIMARRON CORPORATION
666 Baker Street, Suite 319
Costa Mesa, California 92626
(714) 641-1156

Circle #28 on the Reader Service Card

Z-RAM is a trademark of Madison Computer
CP/M is a trademark of Digital Research
Wordcraft Ultra is a trademark of Dataview Ltd.

Using a Joystick on the PET

Draw pictures using a joystick and keyboard

by
Elizabeth Deal

Joysticks have been in use on the PET/CBM computers since the pioneer days of the original ROMs. Atari™ joysticks, as well as any push-button or ground-the-line gadget can be attached to the Parallel User Port of the PET according to instructions in Chuck Johnson's article in the *Commodore PET Users Club Newsletter,* Vol. 1, #2 (undated, probably 1979). The decoding standard described in his article is also used by CURSOR™ magazine.

Many games can be easily recoded from keyboard use to joysticks. There were articles by Harvey Herman in *COMPUTE* #4 and #5 and by David Hook in the *Transactor,* Vol. 3, issues 2 and 3, on the subject. Several recent games I have seen use both the keyboard and joysticks, the program decides who is talking to it, so you have the best of two worlds.

As an illustration of using both keyboard and joystick I have submitted a drawing routine that kids around here enjoy using. You do not need a joystick to use the program; having the keyboard connected is rather advisable.

The program was written for the Upgrade PET. I think that any 40 or 80 column system can use the code as is, though I can't be absolutely sure.

Coding Notes

The joystick decoding standard is that of Chuck Johnson. Variable JJ$ holds the relevant keyboard equivalents of joystick position, with the joystick centered returning a value of 5, equivalent to "don't move," button returns 0 (unused here). No motion is the last value in JJ$, it translates to 5, all other 5's are irrelevant. Joystick, arbitrarily, numbered #1 is used. A method to decode either joystick is at the end of the listing. Non-standard hookup may benefit from changing JJ$.

Drawing is in quarter graphics, using a fast Paul Higginbottom routine from the *Transactor* (Vol. 3, issue 5) reprinted also in the *Commodore* (U.S.) June/July 1982 issue. The key routine which builds his lookup table is in lines 560–580. The rest of his routine has been scattered.

While drawing, jumping over characters or erasing points, it is sometimes desirable to know where you are. For that reason a slightly misleading, full character cursor leads the way and points in the direction of motion. That cursor is pretty enlightening in watching the poor PET put it all together for each point.

A Butterfield screen width (SW) test is in line 430 *(Transactor,* Vol. 3, issue 5).

Lines 180–280 are the main draw, jump, erase loop where directions 1 to 9 are handled. All nondirectional entries are out of the loop in 300–410. We process those lines if and only if keyboard or joystick is doing nothing. You may wish to track down several variables. I$, IV, KY are related to user action; X and Y type variables relate to absolute and relative positions. When M and L are tacked on, these are maximum and minimum permitted values. Array SC holds line addresses (to avoid multiply/divide in loop); array DM contains directional cursors and feedback for the user options. AV is important; it controls the skip over non-quarter-graphics mode (S = skip, N = no skip). This mode works independently of the draw, jump, erase mode.

The keyboard is active only when the joystick is not. Unless you have three hands, let go of the joystick to use the keyboard.

User Options

The program begins in Jump mode so you can position the cursor yourself. To begin drawing press D, to erase points press E, to jump over anything press J. Wrong keys default to the last mode.

The program also begins in a No-skip mode, it draws over whatever is on the screen. To hop over any non-quarter graphic characters (text, overlay picture), press R to switch to

the S-mode, press R again to be in the N-mode when needed.

Letters, D,J,E,S,N are displayed on the bottom line of the screen for feedback. We do not draw on this line, this being controlled by YM in line 530.

Clearing the screen by mistake upsets kids. A safety is therefore built in—do clear-screen twice to clear it. Otherwise you can get out gracefully by pushing any key other than clear-screen. The cursor will land in the 0.0 position, jump mode controls. The same setup will happen when CLS is pressed.

Up-arrow command permits you to add your own subroutines. At the moment it points at no code. Needless to say, disposition of the picture is in your hands. You may add additional options in the hookup to a larger program. With this assumption in mind a RETURN is coded in line 210. You may of course change it to END.

Pressing Q quits this subroutine.

For illustration purposes the screen is not cleared at first, so you can get the feel in using R. You may clear the screen at any time. You may wish to recode by doing CHR$(147) in place of the first (dummy) CHR$(142) in line 540. You may wish to do nothing should this subroutine be a part of drawing on an already prepared display.

A Parting Note on Accuracy

A joystick is a pretty clumsy device when accuracy counts. Refinements of drawing or erasing are easier done using the keyboard. Even though the keyboard acts in the program as if there were repeating keys (so long as you hold the key it does its thing), it is possible to tap it for one or two characters. Quarter graphics are just that, two widths in one, so tap the right number and watch the cursor for precision.

It's not as easy as you may think: in making corrections or in trying to move 'just a little bit' you'll get an idea of what it feels like to BE a computer having to do dumb work. You'll have to think precisely the way your PET does in lines 250–70. ▶

```
100 REM----------------------------
110 REM JOYSTICK/KEYBOARD DRAWING
120 REM ELIZABETH DEAL
130 REM----------------------------
140 IFAA=0THENGOSUB430:GOSUB560:AA=1
150 IV=5:X1=XM:Y1=YM:DD=3:CL=SC(INT(Y1/2)):CV=DM(5):POKECL,CV
160 Q=0:KY=5:PK=5:AV=0:SP=32:AN=48:KX=255:K5=5:K1=1:K9=9:K$=CHR$(0)
170 REM---JS/KB LOOP
180 IV=JS%(PEEK(JS)ANDJ1):IFIV,¢K5THENKY=IV:GOTO220
190 GETI$:IFI$=""THENIFPEEK(KM),¢KXTHENKY=PK:GOTO220
200 IV=ASC(I$+K$):KY=IV-AN:IFKY,K1ORKY¢K9THENGOSUB300:IFEGOTO180
210 IFQTHENRETURN:,END¢
220 POKECL,CV:X=X1+XX(KY):Y=Y1+YY(KY)
230 IFX,XMORX¢XLORY,YMORY¢YLTHENX=X1:Y=Y1:GOTO280
240 CL=SC(Y%(Y))+X%(X):V1=PEEK(CL):IFAVTHENIFV1,¢SPTHENIFI(V1)=QGOTO280
250 SQ=AM(XANDAM,YANDAM):ONDDGOTO260,270,280
260 POKECL,C(I(V1)ORSQ):GOTO280
270 POKECL,C(I(V1)ANDNOTSQ)
280 CV=PEEK(CL):POKECL,DM(PK):PK=KY:X1=X:Y1=Y:GOTO180
290 REM---USR,QUIT,CLS,DRAW,ERASE,JUMP,RESET SKIP MODE
300 E=1:IF(IVAND127)=94THENGOSUB600:RETURN
310 IFIV=82THENAV=1-AV:POKESE,DM(Z1-AV):RETURN
320 IFIV=81THENE=0:Q=1:RETURN
330 IFIV=147THENGOSUB380:PRINTI$;:CV=32:DD=3:X1=XM:Y1=YM:RETURN
340 CM=DD:DD=IV-67:IFIV=74THENDD=3
350 IFDD,0ORDD¢3THENDD=CM:PK=5
360 POKESE-2,DM(DD+Z2):RETURN
370 REM---CONFIRM CLS
380 V=PEEK(SC):POKESC,191
390 GETI$:IFI$=""GOTO390
400 IFASC(I$),¢IVTHENI$=K$
410 POKESC,V:RETURN
420 REM---SETUP1
430 SC=32768:SS=SC+1024:V1=PEEK(SS):SW=80:POKESS,96:IFPEEK(SC)=96THENSW=40
440 POKESS,V1:BL=SC+24*SW:SE=BL+SW-1:KM=151:IFPEEK(50003)=0THENKM=525
450 POKE59459,0:JS=59471:J1=15:J2=1:JJ$="55505798513254 65"
460 DIM JS%(15):FORJ=1TO16:JS%(J-1)=ASC(MID$(JJ$,J))-48:NEXTJ
470 FORI=1TO3:Q=I-2:FORJ=0TO6STEP3:XX(I+J)=Q:NEXTJ,I
480 FORI=1TO9:YY(I)=INT((I-1)/3)-1:NEXTI
490 DIM SC(24),X%(SW*2),Y%(49):FORJ=0TO24:SC(J)=BL-J*SW:NEXTJ
500 FORJ=0TO2*SW-1:X%(J)=J/2:NEXTJ:FORJ=0TO49:Y%(J)=J/2:NEXTJ
510 Z1=14:Z2=9:DIM DM(Z1):FORJ=1TOZ1:READDM(J):NEXTJ
520 DATA 76,33,122,60,43,62,79,30,80,4,5,10,19,14
530 XL=2*SW-1:YL=49:XM=0:YM=2
540 POKE59468,12:PRINTCHR$(142)CHR$(142);:RETURN
550 REM---SETUP2, PAUL HIGGINBOTTOM
560 DIM I(255),C(15),AM(1,1):AM=1:FORI=0TO15:READC(I):I(C(I))=I:NEXT
570 FORI=0TO1:FORJ=0TO1:AM(J,I)=(J+1)*4 I:NEXTJ,I:RETURN
580 DATA 32,123,108,98,126,97,127,252,124,255,225,254,226,236,251,160
590 REM---SAVE/DUMP/NORMAL GRAPHICS/OTHER UTILITIES
600 RETURN: NOTHING HERE
610 REM----------------------------
620 REM JOYSTICK  #1 J1= 15 J2= 1
630 REM           #2 J1=240 J2=16
640 REM IV=JS(((PEEK(JS)ANDJ1))/J2)
650 REM----------------------------
READY.
```

PROGRAM REVIEW

The Commodore BASIC Integer Compiler

by Ron Straley

There are different versions of the Commodore BASIC Integer Compiler to choose from, depending on the system you have. You have to specify your system configuration: 8032,8050; 8032,4040; 4032,4040.

This review was written using the 4032,4040 version, but, from what I can gather, all versions operate in the same way. The only difference is in screen and disk format.

When you get the compiler, the first thing you should do is sit down and read the manual. There are a few things you have to comply with to make Commodore BASIC compatible with the Integer Compiler.

The manual begins with an explanation of what a compiler is and what it does. Then it lists all the conventions used throughout the text and explains that there are two versions included: the Resident Compiler and the Disk Compiler. Then it has you load in the Resident Compiler, just so you can see a compiler work.

The Resident Compiler has all the parameters already set, and is intended to be used for demonstrations and small subroutines. You cannot directly save a program compiled in this fashion, but instead must use the computer's built-in monitor to save a program for later use. The examples provided are short programs, and any programs you put in must also be short, since the Resident Compiler will accept line numbers only up to 49, and you have only 1182 bytes to work with.

After the program has compiled it will list the line numbers and the logical errors that occurred. The errors can be looked up in the manual, where a short explanation is given for the cause of each error.

While the Resident Compiler is compiling your program, watch for a "?" to appear. This is a pointer to a command that the compiler could not recognize. However, the compiler will continue to execute to the end of the BASIC program. It cannot be stopped after you have given the compile command, without aborting the compile.

Next you move on to the world of real compilers, with the Disk Compiler. Now is the time to get your cup of coffee and cigarettes, since this is something of a treat (or a challenge) if you have not had any dealings with machine language or compilers. Let me say here that the Commodore people have put a lot of effort into their section on the Disk Compiler and have done a super job on a seemingly insurmountable task.

The first thing you should remember is that this is an integer compiler and its main function should be only to compile subroutines that need to run at a high rate of speed, such as calculations of integer numbers, math functions or any I/O functions that need the screen, disk or printer. It will compile a complete BASIC program, but highspeed keyboard input of operator decision routines, for example, are a waste of time, since the compiled program usually will be longer than the original BASIC version. Why waste memory on things that don't need increased speed?

Under the Disk Compiler you must set up the storage addresses yourself (the Resident Compiler did this for you). You must specify the locations for SUBS, STRINGS, CODE, DATA, POINTERS. This should not be a problem, since there are plenty of examples.

The rest of the Disk Compiler program will operate in the same manner as the Resident Compiler, except for some of the advanced commands, which are what makes this such an excellent compiler.

For instance, the SET command allows you to specify memory locations as variables. This is useful in using the PET registers. SET DDR%@ 59459, for example, allows you to use DDR% = 255. Now the direction of the user port is output. SET ULC%@ 59468 allows you to use ULC% = 12 or ULC% = 14 to program control upper and lower case.

The WEDGE command allows you to put a wedge in the CHRGET routine, to enable you to write your own commands. You should study how the CHRGET routine works before playing with this one.

INTERRUPT lets you look at 200 bytes of zero page every 1/60th of a second. There are six related commands that are used in compiled BASIC:

1. SCHED allows you to inject another routine into the interrupt routine to be executed every 1/60th of a second.
2. DSCHED allows the interrupt routine to return to its original condition.
3. INTERRUPT is the command you must use as the first statement following SCHED.
4. RTI terminates all interrupt routines and passes control to the computer.
5. DI sets the processor interrupt mask so no interrupts can occur.
6. EI clears the interrupt mask so interrupts can occur.

There is also a set of software SWITCHES that control compiler options:

S#2 allows the stop key to be enabled when set.
S#5 omits the initialization of certain zero page variables that are normally initialized with a compiled program.
S#7 causes the two character sets to be toggled.

There are four IEEE commands for logical files:

1. TALK# Same as CMD in BASIC.
2. UNTALK# Same as Print#1:Close1 in BASIC.
3. LISTEN# This allows all GET# or INPUT# to receive data from an opened file.
4. UNLISTEN# This stops the LISTEN command.

These are important only if you wish to take advantage of the speed of the IEEE bus.

A few more commands that add to the advantages of this compiler are:

1. DELAY If used as DELAY A B, this allows you to delay execution of the system by value of A times B units of system time.
2. PUSH Registers A,X and Y are saved on the stack.
3. POP Restore registers back from stack.
4. PASS Allows a value to be passed back to BASIC after a USR function.
5. JUMP Allows you to jump to a machine code routine specified.
6. EXTRACT array name, REPLACE array name, GET, INPUT, WAIT—these are all pretty much self-explanatory.

There is a section dealing with interfacing the compiled programs with BASIC, a section on error messages, and a section for helping solve any problems you may be having. There is also a good description of the BASIC commands, which is helpful because some of the commands will not work as normally written in BASIC.

This should give you some idea of what this package can do. As I have said it is a powerful package if treated correctly, and should be worth the purchase price. It would also be a good thing to have if you are trying to learn machine language programming.

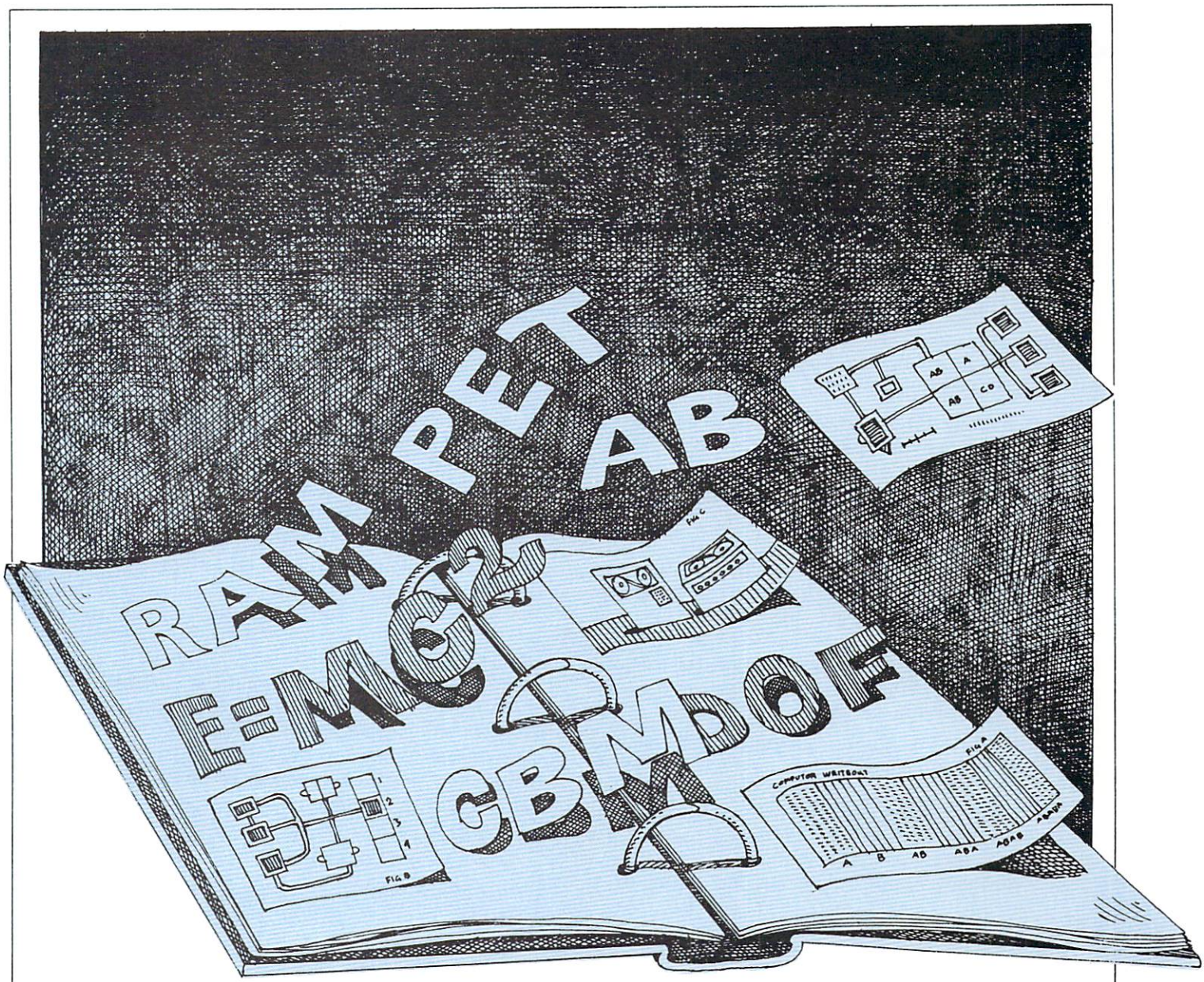
The only fault I found was that it would not compile TAB, SPC or MID$ correctly. For example, the MID$ function MID$ (A$,2) will return only the second character from A$. To get more than one character from a string you have to put it in a loop as follows:

```
10FORJ=5T09:B$=MID$(A$,J):C$=C$+B$:NEXTJ
```

This will make C$ equal to characters 5 through 9 of A$.

I haven't worked out the TAB and SPC functions yet, but there is a way to do them, also.

All in all, I found it to be an excellent package.



Commodore 64K Memory Expansion Board

by
Ira Neal

The Commodore 64K Memory Expansion Board adds 64K-bytes of memory to the Commodore 8032, providing a total of 96K-bytes of available RAM. The expansion memory board can be used with commercial software packages like Wordcraft, Wordcraft Ultra, VisiCalc, and Silicon Office or with the extended BASIC language provided with the board.

The add-on memory is mapped into four 16K-byte blocks. Only two of these blocks can reside in memory at one time. The expansion RAM is mapped in the same address space normally allocated for the system ROMs, I/O registers and screen memory (See Fig. 1). The expansion board is disabled at power up, so the 8032 will display 31743 bytes free. The expansion RAM is accessed by a write-only control register at location $FFF0 (See Fig. 2). The function of the control bits in this register are as follows:

Control Register Bit 0—When equal to 1, addresses $8000 through $BFFF on the Expansion Memory Board are write protected. The screen is not protected if screen peek through is enabled. When equal to 0, the addresses are not write protected.

Control Register Bit 1—When equal to 1, addresses $C000 through $FFFF on the Expansion Memory Board are write protected. The I/O registers are not write protected if I/O peek through is enabled. When equal to 0, the addresses are not protected.

Control Register Bit 2—When equal to 1, block 1 is selected. When equal to 0, block 0 is selected. These blocks are 16K-bytes and reside at locations $8000 through $BFFF.

Control Register Bit 3—When equal to 1, block 3 is selected. When equal to 0, block 2 is selected. These

blocks are 16K-bytes and reside at locations $C000 through $FFFF.

Control Register Bit 4—Reserved. No function.

Control Register Bit 5—When equal to 1, screen peek through is enabled. This allows the screen memory at locations $8000 through $87FF to be accessed.

Control Register Bit 6—When equal to 1, I/O peek through is enabled. This allows the I/O register at locations $E800 through $EFFF to be accessed.

Control Register Bit 7—When equal to 1, enables the expansion memory and the above registers. Bit 7 defaults to 0 on power up.

A diskette containing programs for testing and controlling the expansion memory is supplied with the board. These include: '8032.MEM.PRG' a diagnostic test program, 'EXPANDED DEMO' a demonstration of the use of the expansion memory as a 'fast disk', 'EXPANDED-BASIC' a program to add expanded memory functions to Commodore BASIC, 'ADD-ON-MON' a TIM monitor with addition functions for the expanded memory, and 'ADD-ON-LOAD' a program that enables the loading of different operating systems.

EXPANDED-BASIC is a software routine that is loaded into high memory ($7800-$7BE0), leaving 29K bytes of contiguous BASIC program space in lower memory. This program is a psuedo-cache memory system, or 'fast-disk', that allows users to store or 'cache' programs and data in the expansion RAM area for ultra high-speed access. The following instructions are added to BASIC:

RECALL
Format: !r,0:"filename",s(u)(p), device
Purpose: Cache a file from disk.

LOAD
Format: !l,"program name"
Purpose: Move data from ADD-ON to BASIC text area.

OVERLAY
Format: !o,"program name"
Purpose: Overlays data from ADD-ON to current program in BASIC text area.

EXECUTE
Format: !e,"program name"
Purpose: Clears BASIC text area, Moves data from ADD-ON to BASIC text area and executes program.

QUIT
Format: !q
Purpose: Turn off expanded BASIC functions.

The programs and files are placed in the expansion memory in a contiguous manner. If data will not fit in the first expansion memory bank, wraparound will occur. All 64K-bytes are available; however, the number of cached files is limited to ten.

ADD-ON-LOAD is a special loader which loads one of the three special versions of BASIC, provided on the supplied diskette. These BASIC systems are loaded into the expansion RAM to enable the execution of 40-column or 2.0 BASIC programs on the 8032. The supplied BASIC systems are: 'BASIC 2.0', a copy of 2.0 BASIC with a patch to re-initialize the 8032's screen controller; 'BASIC 4.0/40', a copy of the BASIC found in 4000 series machines; 'BASIC 4.0/80', a copy of the BASIC that is resident in the 8032.

In assembly language programming all 96K-bytes of memory are available to the advanced assembly language programmer. They must remember that monitor calls can only be executed with the expansion RAM disabled, and the expansion RAM banks must be switched to access more than 64K-bytes of contiguous memory.

The 64K Expansion Memory Board can be installed quickly with the supplied instructions. After installation is complete the correct operation can be verified by the execution of the 'EXPANDED DEMO' and '8032.MEM.PRG'. The 'EXPANDED DEMO' program caches four programs in the expanded RAM and executes them one at a time. The '8032.MEM.PRG' is a diagnostic test program for the expansion RAM. C=

Circle #34 on the Reader Service Card

Figure 1

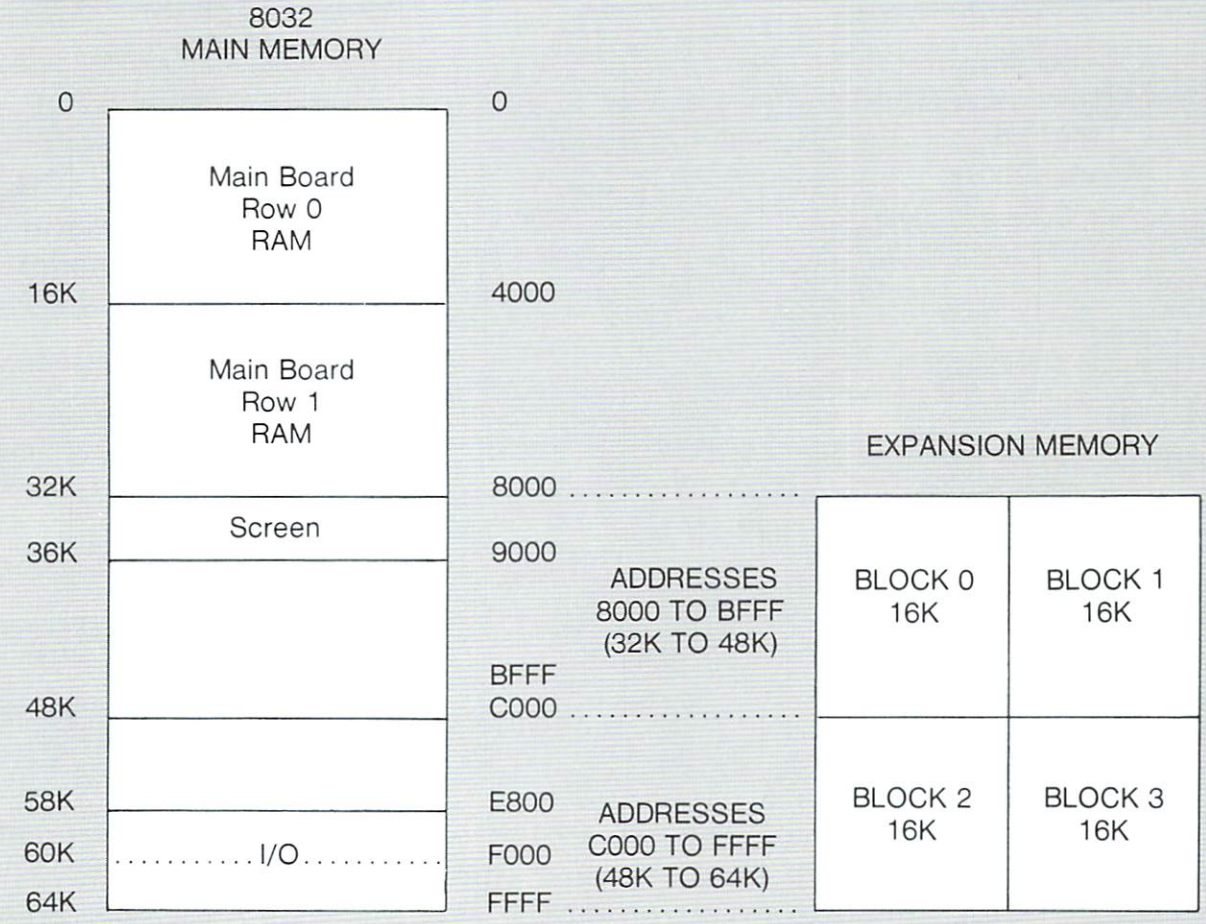
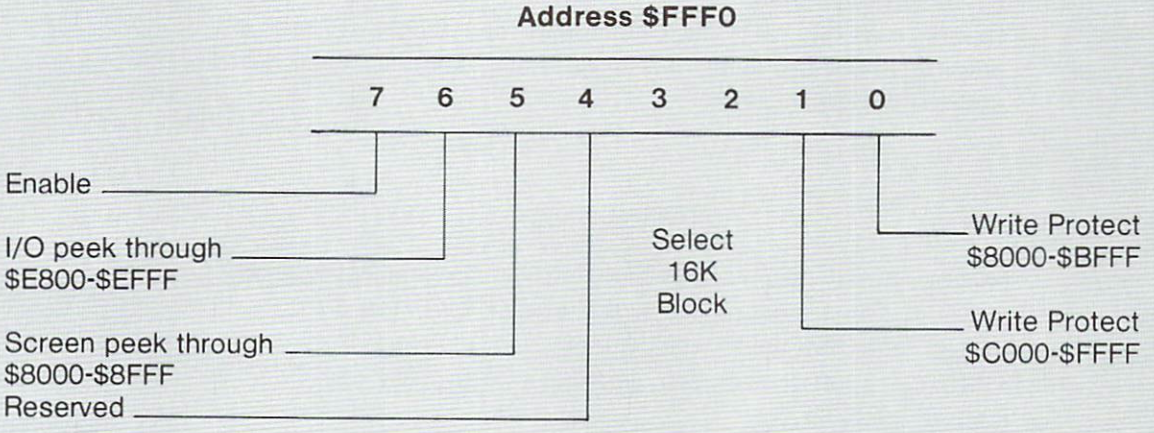


Figure 2



0 0	2 and 0
0 1	2 and 1
1 0	3 and 0
1 1	3 and 1

NEW PRODUCT DEVELOPMENTS

Company:
Leading Edge Products, Inc.
225 Turnpike Street
Canton, MA 02021
800-343-6833
(Mass. 617-828-8150)

Product:
BM-12EN high resolution, 12" non-glare green screen monitor—This new introduction has a video amp bandwidth of 20 MHz, which results in a sharper, more precise screen image. In addition, the BM-12EN monitor incorporates a non-glare screen that adds sharpness to the image and reduces eye strain and fatigue from intensive use.

Leading Edge Products is the exclusive U.S. distributor of the BM-12EN, as well as the "Mean Green" BM-12AU and the color composite BM-1400CL monitors.

All Leading Edge monitors are fully covered by an exclusive one-year warranty by which any faulty monitor will be repaired or replaced.

Price: $105.00

Company:
Xitel, Inc.
2678 North Main Street #1
Walnut Creek, CA 94596
415-944-9277

Product:
Model CX-6401 printer interface—Allows the Commodore 64 and VIC 20 computers to use any type of standard parallel interface printer. The interface connects to the serial port/bus on the computer, not the user port, allowing the user port to remain open for use by a modem or RS232 peripheral device. The CX-6401 eliminates the handshaking problems of the RS232C cartridge, switches modes to correct ASCII problems, and requires no expensive RS232C cable or external power source. It also allows use of multiple printers.

The CX-6401 comes standard with a six-foot serial bus cable and connector for direct connection to the computer and a one-foot parallel cable for connection to the printer, so installation is simple.

Price: $89.95

Product:
Model CX-6402 full RS232 interface—For use with Commodore 64 and VIC 20 computers. The CX-6402 allows use of any RS232C peripherals, such as modems, printers, etc. The interface connects to the computer's user port, and allows full utilization of all eight active handshaking RS232C signal lines. It features full RS232C logic levels, operates at all VIC 20/Commodore 64 baud rates and requires neither an external RS232C cable nor an external power source. Comes ready to use and easy to install.

Price: $59.95

Company:
Sun Research
Box 210
New Durham, NH 03855
603-859-7110

Product:
MAYDAY Uninterruptible Power Systems—Provide power backup when voltage drops below a specified level, and protection against power surges. Available in three different types: MAYDAY fast-switchable series with a 120Hz square wave output, MAYDAY60 fast switchable series with a 60Hz sine wave output and MAYDAY60+C continuous series with non-switching 60Hz sine wave output.

Price: Contact company

Company:
Specialised Software, Inc.
P.O. Box 2
Wilmot, WI 53192
414-862-6968

Product:
Complex Arithmetic—Given any two complex numbers, modulus and angle of each is found, with modulus and angle of result under addition, subtraction, multiplication and division. Introduction manual and operations guide with worked-through examples.

Price: $27.50

Product:
Statistics—Finds Z-score, T-score, mean, standard deviations, medians and chi-square values from raw data.

Price: $27.50

Product:
Chemical Equations—The Luteri-Scalzi Matrix is used to automatically balance chemical equations.

Price: $27.50

Circle #37 on the Reader Service Card

NEW PRODUCT DEVELOPMENTS

Product:
Curve-Fitting, Master Program—Takes data and tells which curve-fitting operation fits the data best. Provides an error analysis of each model tried, as well as a "best fit" model output. Includes the following curve-fitting methods: logarithmic, exponential, Gaussian least squares, polynomial synthesis, Langrange interpolation and Chebyshev interpolation. Includes introduction manual and operations guide with worked-through examples.

Price: $100.00

Product:
Electrochemistry—Simulates complete physical chemistry of an electroplating bath. Operations guide with worked-through examples.

Price: $200.00

Product:
Multilayer Stress Analysis—Determines, from temperature shifts and mechanical properties, the shear, compression and tensile stresses in each layer of a bonded, heterogeneous multilayer structure. Includes operations guide and worked-through examples.

Price: $150.00

Company:
Midnite Software Gazette
635 Maple
Mt. Zion, IL 62549
217-864-5320

Product:
The Whole PET Catalog—A collection of over 500 independent reviews of products for Commodore computers, drawn from the first seven issues of the Midnite Software Gazette. Now merged with The Paper, another independent publication supporting Commodore products, Midnite is a widely respected resource in 41 states and 11 countries.

Price:
Whole PET Catalog: $10.00
Subscription to Midnite: $20 U.S., $25 Canadian, $45 overseas. 6 bi-monthly issues

---

Company:
Computer Case Company
5650 Indian Mound Court
Columbus, OH 43213
614-868-9464

Product:
Carrying cases for Commodore 64 or VIC 20—The CM703 case will hold a Commodore 64 or VIC 20 computer along with one or two 1541 disk drives and power supply. The CM704 case will hold the computer, datassette recorder and power supply. Both cases also have ample room for additional equipment, working papers, manuals, etc.

The cases are constructed of luggage material with hard sides, padded handles, brass hardware and key locks. Rubber pads provide furniture protection, and steel lugs on the bottom protect the case when transporting. The outside is covered in high quality scuff-resistant textured vinyl. Tops are easily removed so the equipment can be operated without removal from the case.

Price: Contact company

Company:
Strictly Commodore
47 Coachwood Place N.W.
Calgary, Alberta T3H 1E1
Canada

Product:
Strictly Commodore (Newsletter)—Contains 4 to 6 program listings to enhance the value of your VIC 20, PET, CBM and Super-PET, detailed reviews of major products, programming hints and other material of interest to Commodore computer owners. Also offers a software exchange for owner-developed software.

Price: $15 for 6 bi-monthly issues

Company:
Hayden Book Company, Inc.
50 Essex Street
Rochelle Park, NJ 07662
201-843-0550

Product:
BASIC Computer Programs for Business, Volume 2 $13.95
Cryptanalysis for Microcomputers $14.95
Codes, Ciphers and Computers $13.95
Problem-Solving Principles for ADA Programmers: Applied Logic, Psychology and Grit $9.95 C

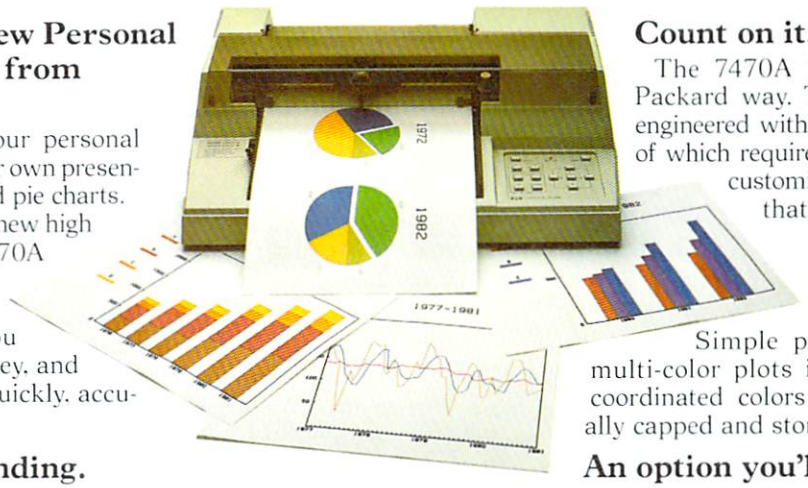
Plot your next meeting yourself.

Read how 2 pens can become your best presentation tools.

Introducing the New Personal Computer Plotter from Hewlett-Packard.

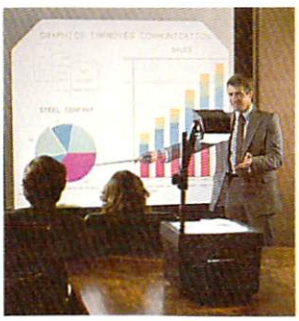
Now you can use your personal computer to generate your own presentation charts, graphs, and pie charts. How? Simply add on the new high quality, low cost HP 7470A Personal Computer Plotter.

The 7470A helps you save time and save money, and lets you communicate quickly, accurately and *effectively.*



Quicker understanding.

Data, when visualized graphically, becomes information fast. Charts and bar graphs can make any presentation clearer and more readily understood. But asking your staff to produce the graphics manually for your next presentation doesn't ensure accuracy or artistic talent. And going to outside graphics suppliers can be costly. Combined with your personal computer, the new HP 7470A plotter does the communicating for you. Quickly. Logically. And with off-the-shelf software available from most HP dealers.



Fast and pretty.

The 7470A gives you high plotting speed with excellent line quality...faster than any competitive small plotter. On top of all that, it comes in an attractive design package that looks nice on your desk. And it does it for only $1,550. (U.S.A. domestic suggested retail price.)

Count on it.

The 7470A is built the Hewlett-Packard way. To last. Designed and engineered with only a few parts, none of which require adjustment. And with customized integrated circuits that ensure reliability.

Pen pals.

The HP 7470A has two single-pen stables. Simple pen changes give you multi-color plots in your choice of ten coordinated colors. Pens are automatically capped and stored.

An option you'll want, too.

For only $95, you can also get a 17057 Overhead Transparency Kit that turns your plots into transparencies for overhead projectors. For "I need it tomorrow at 9:00 A.M.!" meetings, it's a necessity.

Start plotting your next presentation today. Clip and mail the coupon below. Now.

Mail the coupon below and we'll send you—absolutely free—a sample plot, a more detailed brochure, and a sample overhead transparency.

Then...stop in at your nearest Hewlett-Packard Dealer. See the HP 7470A in action. Once you see it demonstrated you'll find a hundred ways to make your own applause-winning presentations.

When performance must be measured by results

HEWLETT PACKARD

Seeing is believing. Send me a sample plot, an overhead transparency, and more detailed information.

Name _____ Title _____

Company _____

Address _____

City, State & Zip _____

Phone Number ( ) _____

My computer is _____

Send to: Hewlett-Packard, 16399 W. Bernardo Drive, San Diego, CA 92127—Attn: Nancy Carter 11203 CO O

Circle #39 on the Reader Service Card

CMS ACCOUNTING SYSTEM

CMS GENERAL ACCOUNTING SYSTEM

A fully interactive General Accounting System designed especially for the first time user. All input requests are fully prompted with complete verification of input data. Most reports may be printed either to the screen or the printer and started or stopped at any point. The user is led completely through each function by a series of highlighted prompts fully explaining the required input at each point. A professionally written instruction manual is included which shows sample reports generated by the system and further explains each step and prompt as it is encountered by the user. These user prompts, together with the detailed step by step manual, make the system extremely user friendly.

GENERAL LEDGER

Up to 1000 accounts on the Chart of Accounts. Fully departmentalized up to ninety-nine departments. A Cash Receipts and Disbursements Journal allows each check or deposit to be distributed to nine separate General Ledger accounts. Non cash postings may be entered through the General Journal. The Cash Journal and General Journal may be printed at any time either to the screen or to the printer. A non structured Chart of Accounts allows easy tailoring of the Financial Statements to fit varying user needs. Account balances are maintained for current month, quarter to date, and year to date. A detailed Budget Analysis Report may be printed at any time either to the screen or to the printer. The General Ledger program may be used alone or may be set up to accept postings from Accounts Receivable, Accounts Payable, Payroll, or other programs.

PAYROLL

Maintains current, quarterly, and yearly totals for 350 employees. Prints Payroll checks with full deduction and pay detail as well as year to date totals. Accommodates weekly, bi-weekly, semi-monthly, and monthly employees. Pays regular, overtime, holiday, and piecework hours. Handles federal and state deductions plus up to eight miscellaneous deductions and payments per employee. Prints Payroll Journal, Check Register, and an Absentee Report as well as state and federal 941 information and W-2 forms. The Payroll program may be used alone or set up to automatically update the General Ledger.

ACCOUNTS RECEIVABLE

Allows a total of 1800 invoices and customers per 8050 data disk. Each invoice line item may be distributed to a separate General Ledger income account. Prints invoices and monthly statements and allows for individualized messages on the statements. Finance charges may be automatically added to overdue invoices at a user determined rate and period. Complete invoice aging reports with user set aging breaks may be printed at any time either to the screen or to the printer. A special Overdue Invoice Report is also available for close monitoring of receivables. Handles credit memos as well as invoices. A Sales Tax Report is available showing total sales and sales taxes payable for up to nine separate sales tax rates. Total sales and sales commissions earned are automatically tracked for up to fifteen salesmen. The Accounts Receivable program may be used alone or set up to automatically update the General Ledger.

ACCOUNTS PAYABLE

Allows a total of 2500 invoices and vendors per 8050 data disk. Each invoice may be distributed among nine separate General Ledger expense accounts. Prints checks with complete voucher detail for up to thirteen invoices per check. Invoices may be paid by vendor range or by individual invoice numbers. Credit memos and term discounts are taken automatically. Full or partial payments may be made for each invoice. A Check Register may be printed at any time. Complete aging reports with user set aging breaks may be printed at any time either to the screen or to the printer. A special Cash Requirements Analysis Report is available to assist in determining upcoming cash requirements. The Accounts Payable program may be used alone or may be set up to automatically update the General Ledger.

OFFICIALLY APPROVED BY COMMODORE

CMS Software Systems

2204 Camp David Mesquite, TX 75149 214-285-3581

Comments on the Commodore 64 Memory Maps.

by
Jim Butterfield

At first glance, the Commodore 64 seems to be exactly the same as VIC except for the new sound/video chips. In a way, that's true: it's the same BASIC and the operating system does the same things.

Of course, you have forty columns on the screen, plus sprite graphics, plus good-sound three voice music synthesis. You have a vast amount more memory. But these seem almost like cosmetic additions.

Don't be fooled. The 64 is much more powerful than the VIC. It's mainly because of the extra memory, and the way it's handled.

You can (carefully) swap out BASIC and even the Kernal and substitute your own programs in RAM. It's not a trick for beginners; but it's not hard to do. So the 64 can drop BASIC in favor of another language—or a different BASIC. If you're sharp, you can do your own operating system. Here's an easy one: if there's some small thing you've always hated about BASIC, you can copy it into its matching RAM area . . . then swap in the RAM . . . then change the BASIC to do what you've always wanted it to do. You've never had such flexibility.

If you have been using a VIC, you may have already learned the PEEKs and POKEs that will apply to the 64. If you come from the PET/CBM environment, you'll need to adapt to the new places. All the PET/CBM addresses are there, but they are moved; as a bonus (and to help reduce the pain) there are many more user "hooks" to let you communicate with and control the computer.

A word on the ROM maps: these detailed addresses are intended primarily for study rather than inviting you to jump in and use them. Stay with the "jump table" for most of your work—it will make your programs much more portable.

After the VIC, it's nice to go back to a machine that keeps the screen in a fixed place, and for that matter always starts BASIC from the same address. It's not totally PET-compatible, since the screen occupies much lower memory and BASIC starts a little higher than before . . . but it's not hard to make the changes.

Don't let the technology changes fool you. The moment you touch the machine, you'll recognize its style. They may change the name, the chips, the memory and the price . . . but it's still a PET at heart. ▶

COMMODORE 64 MEMORY MAP

Compiled by Jim Butterfield

Hex	Decimal	Description
0000	0	Chip directional register
0001	1	Chip I/O; memory & tape control
0003-0004	3-4	Float-Fixed vector
0005-0006	5-6	Fixed-Float vector
0007	7	Search character
0008	8	Scan-quotes flag
0009	9	TAB column save
000A	10	0=LOAD, 1=VERIFY
000B	11	Input buffer pointer/# subscrpt
000C	12	Default DIM flag
000D	13	Type: FF=string, 00=numeric
000E	14	Type: 80=integer, 00=floating point
000F	15	DATA scan/LIST quote/memry flag
0010	16	Subscript/FNx flag
0011	17	0=INPUT;$40=GET;$98=READ
0012	18	ATN sign/Comparison eval flag
0013	19	Current I/O prompt flag
0014-0015	20-21	Integer value
0016	22	Pointer: temporary strg stack

0017-0018	23-24	Last temp string vector
0019-0021	25-33	Stack for temporary strings
0022-0025	34-37	Utility pointer area
0026-002A	38-42	Product area for multiplication
002B-002C	43-44	Pointer: Start-of-Basic
002D-002E	45-46	Pointer: Start-of-Variables
002F-0030	47-48	Pointer: Start-of-Arrays
0031-0032	49-50	Pointer: End-of-Arrays
0033-0034	51-52	Pointer: String-storage(moving down)
0035-0036	53-54	Utility string pointer
0037-0038	55-56	Pointer: Limit-of-memory
0039-003A	57-58	Current Basic line number
003B-003C	59-60	Previous Basic line number
003D-003E	61-62	Pointer: Basic statement for CONT
003F-0040	63-64	Current DATA line number
0041-0042	65-66	Current DATA address
0043-0044	67-68	Input vector
0045-0046	69-70	Current variable name
0047-0048	71-72	Current variable address
0049-004A	73-74	Variable pointer for FOR/NEXT
004B-004C	75-76	Y-save; op-save; Basic pointer save
004D	77	Comparison symbol accumulator
004E-0053	78-83	Misc work area, pointers, etc
0054-0056	84-86	Jump vector for functions

BUTTERFIELD ON COMMODORE

0057-0060	87-96	Misc numeric work area
0061	97	Accum#1: Exponent
0062-0065	98-101	Accum#1: Mantissa
0066	102	Accum#1: Sign
0067	103	Series evaluation constant pointer
0068	104	Accum#1 hi-order (overflow)
0069-006E	105-110	Accum#2: Exponent, etc.
006F	111	Sign comparison, Acc#1 vs #2
0070	112	Accum#1 lo-order (rounding)
0071-0072	113-114	Cassette buff len/Series pointer
0073-008A	115-138	CHRGET subroutine; get Basic char
007A-007B	122-123	Basic pointer (within subrtn)
008B-008F	139-143	RND seed value
0090	144	Status word ST
0091	145	Keyswitch PIA: STOP and RVS flags
0092	146	Timing constant for tape
0093	147	Load=0, Verify=1
0094	148	Serial output: deferred char flag
0095	149	Serial deferred character
0096	150	Tape EOT received
0097	151	Register save
0098	152	How many open files
0099	153	Input device, normally 0
009A	154	Output CMD device, normally 3
009B	155	Tape character parity
009C	156	Byte-received flag
009D	157	Direct=$80/RUN=0 output control
009E	158	Tp Pass 1 error log/char buffer
009F	159	Tp Pass 2 err log corrected
00A0-00A2	160-162	Jiffy Clock HML
00A3	163	Serial bit count/EOI flag
00A4	164	Cycle count
00A5	165	Countdown,tape write/bit count
00A6	166	Tape buffer pointer
00A7	167	Tp Wrt ldr count/Rd pass/inbit
00A8	168	Tp Wrt new byte/Rd error/inbit cnt
00A9	169	Wrt start bit/Rd bit err/stbit
00AA	170	Tp Scan;Cnt;Ld;End/byte assy
00AB	171	Wr lead length/Rd checksum/parity
00AC-00AD	172-173	Pointer: tape bufr, scrolling
00AE-00AF	174-175	Tape end adds/End of program
00B0-00B1	176-177	Tape timing constants
00B2-00B3	178-179	Pntr: start of tape buffer
00B4	180	1=Tp timer enabled; bit count
00B5	181	Tp EOT/RS232 next bit to send
00B6	182	Read character error/outbyte buf
00B7	183	# characters in file name
00B8	184	Current logical file
00B9	185	Current secndy address
00BA	186	Current device
00BB-00BC	187-188	Pointer to file name
00BD	189	Wr shift word/Rd input char
00BE	190	# blocks remaining to Wr/Rd
00BF	191	Serial word buffer
00C0	192	Tape motor interlock
00C1-00C2	193-194	I/O start address

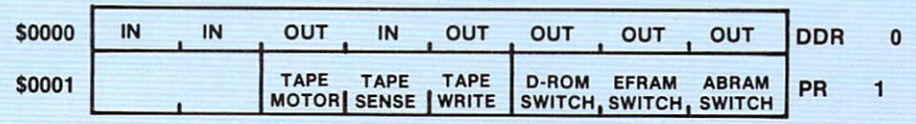
00C3-00C4	195-196	Kernel setup pointer
00C5	197	Last key pressed
00C6	198	# chars in keybd buffer
00C7	199	Screen reverse flag
00C8	200	End-of-line for input pointer
00C9-00CA	201-202	Input cursor log (row, column)
00CB	203	Which key: 64 if no key
00CC	204	0=flash cursor
00CD	205	Cursor timing countdown
00CE	206	Character under cursor
00CF	207	Cursor in blink phase
00D0	208	Input from screen/from keyboard
00D1-00D2	209-210	Pointer to screen line
00D3	211	Position of cursor on above line
00D4	212	0=direct cursor, else programmed
00D5	213	Current screen line length
00D6	214	Row where curosr lives
00D7	215	Last inkey/checksum/buffer
00D8	216	# of INSERTs outstanding
00D9-00F2	217-242	Screen line link table
00F3-00F4	243-244	Screen color pointer
00F5-00F6	245-246	Keyboard pointer
00F7-00F8	247-248	RS-232 Rcv pntr
00F9-00FA	249-250	RS-232 Tx pntr
00FF-010A	256-266	Floating to ASCII work area
0100-103E	256-318	Tape error log
0100-01FF	256-511	Processor stack area
0200-0258	512-600	Basic input buffer
0259-0262	601-610	Logical file table
0263-026C	611-620	Device # table
026D-0276	621-630	Sec Adds table
0277-0280	631-640	Keybd buffer
0281-0282	641-642	Start of Basic Memory
0283-0284	643-644	Top of Basic Memory
0285	645	Serial bus timeout flag
0286	646	Current color code
0287	647	Color under cursor
0288	648	Screen memory page
0289	649	Max size of keybd buffer
028A	650	Repeat all keys
028B	651	Repeat speed counter
028C	652	Repeat delay counter
028D	653	Keyboard Shift/Control flag
028E	654	Last shift pattern
028F-0290	655-656	Keyboard table setup pointer
0291	657	Keyboard shift mode
0292	658	0=scroll enable
0293	659	RS-232 control reg
0294	660	RS-232 command reg
0295-0296	661-662	Bit timing
0297	663	RS-232 status
0298	664	# bits to send
0299-029A	665	RS-232 speed/code
029B	667	RS232 receive pointer
029C	668	RS232 input pointer
029D	669	RS232 transmit pointer

BUTTERFIELD ON COMMODORE

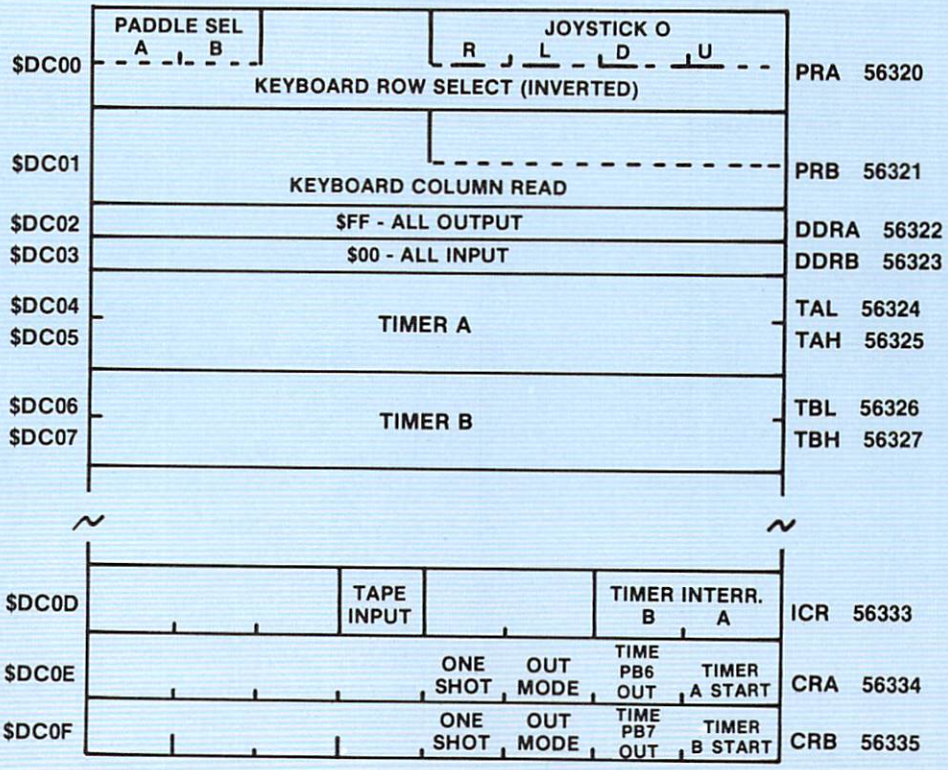
029E	670	RS232 output pointer	
029F-02A0	671-672	IRQ save during tape I/O	
02A1	673	CIA 2 (NMI) Interrupt Control	
02A2	674	CIA 1 Timer A control log	
02A3	675	CIA 1 Interrupt Log	
02A4	676	CIA 1 Timer A enabled flag	
02A5	677	Screen row marker	
02C0-02FE	704-766	(Sprite 11)	
0300-0301	768-769	Error message link	
0302-0303	770-771	Basic warm start link	
0304-0305	772-773	Crunch Basic tokens link	
0306-0307	774-775	Print tokens link	
0308-0309	776-777	Start new Basic code link	
030A-030B	778-779	Get arithmetic element link	
030C	780	SYS A-reg save	
030D	781	SYS X-reg save	
030E	782	SYS Y-reg save	
030F	783	SYS status reg save	
0310-0312	784-785	USR function jump	(B248)
0314-0315	788-789	Hardware interrupt vector	(EA31)
0316-0317	790-791	Break interrupt vector	(FE66)
0318-0319	792-793	NMI interrupt vector	(FE47)
031A-031B	794-795	OPEN vector	(F34A)
031C-031D	796-797	CLOSE vector	(F291)
031E-031F	798-799	Set-input vector	(F20E)
0320-0321	800-801	Set-output vector	(F250)
0322-0323	802-803	Restore I/O vector	(F333)
0324-0325	804-805	INPUT vector	(F157)
0326-0327	806-807	Output vector	(F1CA)
0328-0329	808-809	Test-STOP vector	(F6ED)
032A-032B	810-811	GET vector	(F13E)
032C-032D	812-813	Abort I/O vector	(F32F)
032E-032F	814-815	Warm start vector	(FE66)
0330-0331	816-817	LOAD link	(F4A5)
0332-0333	818-819	SAVE link	(F5ED)
033C-03FB	828-1019	Cassette buffer	
0340-037E	832-894	(Sprite 13)	
0380-03BE	896-958	(Sprite 14)	

```
03C0-03FE   960-1022  (Sprite 15)
0400-07FF  1024-2047  Screen memory
0800-9FFF  2048-40959 Basic ROM memory
8000-9FFF 32768-40959 Alternate:  ROM plug-in area
A000-BFFF 40960-49151 ROM:  Basic
A000-BFFF 49060-59151 Alternate:  RAM
C000-CFFF 49152-53247 RAM memory, including alternate
D000-D02E 53248-53294 Video Chip (6566)
D400-D41C 54272-54300 Sound Chip (6581 SID)
D800-DBFF 55296-56319 Color nybble memory
DC00-DC0F 56320-56335 Interface chip 1, IRQ (6526 CIA)
DD00-DD0F 56576-56591 Interface chip 2, NMI (6526 CIA)
D000-DFFF 53248-53294 Alternate:  Character set
E000-FFFF 57344-65535 ROM:  Operating System
E000-FFFF 57344-65535 Alternate:  RAM
FF81-FFF5 65409-65525 Jump Table, Including:
  FFC6 - Set Input channel
  FFC9 - Set Output channel
  FFCC - Restore default I/O channels
  FFCF - INPUT
  FFD2 - PRINT
  FFE1 - Test Stop key
  FFE4 - GET
```

Processor I/O Port (6510) Commodore 64

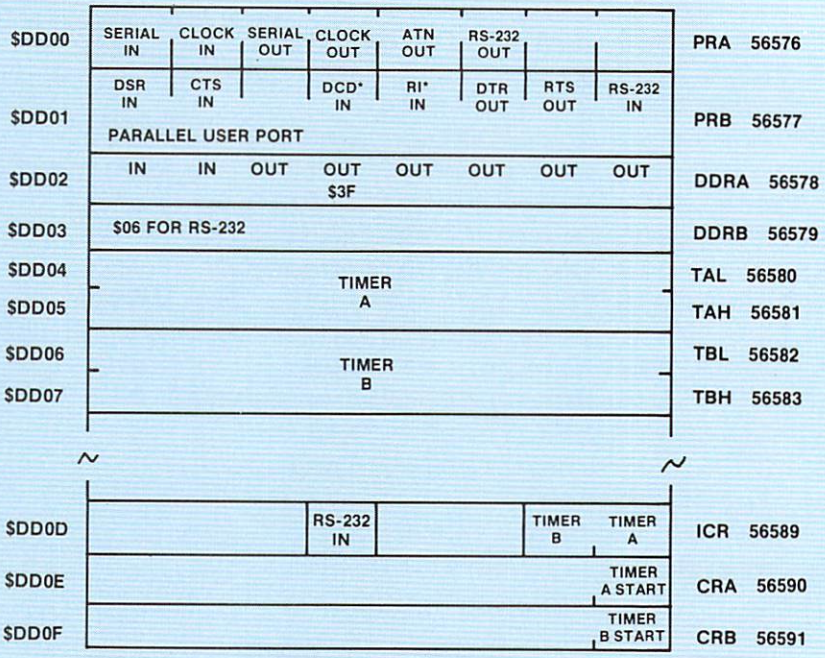


CIA 1 (IRQ) (6526) Commodore 64



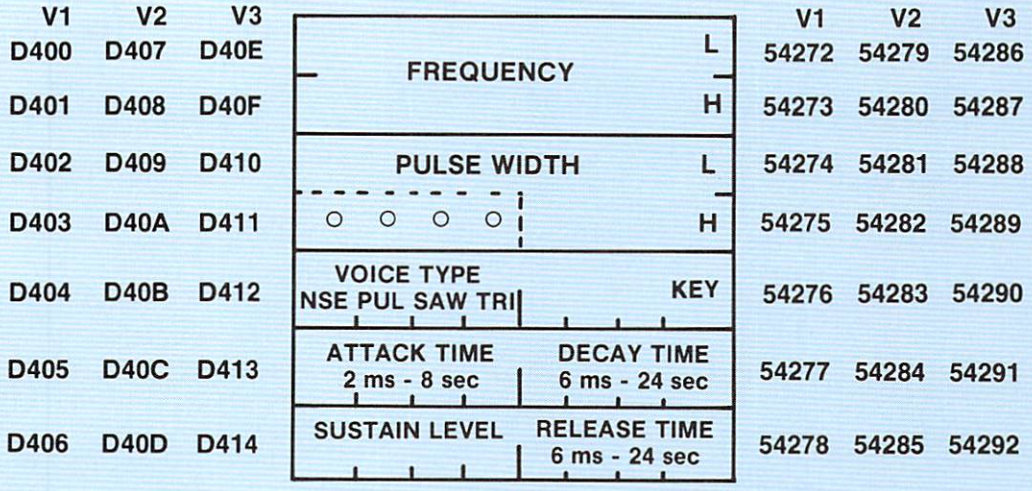
BUTTERFIELD ON COMMODORE

CIA 2 (NMI) (6526) Commodore 64

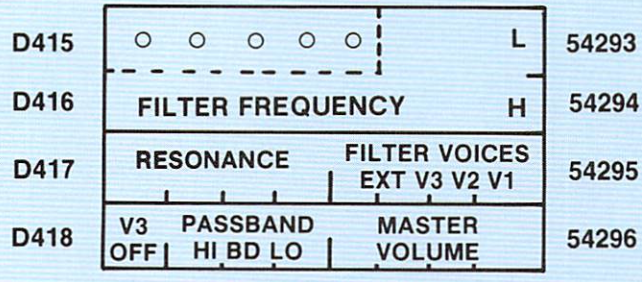


Connected but not used in system

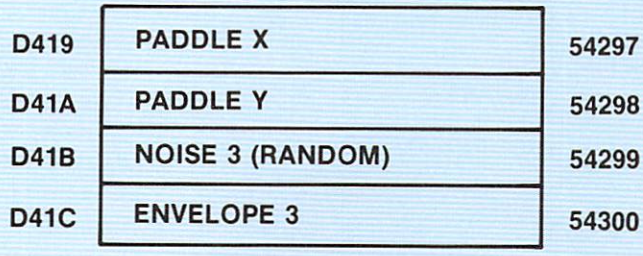
SID (6581) Commodore 64



VOICES
(WRITE ONLY)



FILTER & VOLUME
(WRITE ONLY)



SENSE
(READ ONLY)

SPECIAL VOICE FEATURES (TEST, RING MOD, SYNC) ARE OMITTED FROM THE ABOVE DIAGRAM.

BUTTERFIELD ON COMMODORE

```
Commodore 64 - ROM memory map

A000; ROM control vectors
A00C; Keyword action vectors
A052; Function vectors
A080; Operator vectors
A09E; Keywords
A19E; Error messages
A328; Error message vectors
A365; Misc messages
A38A; Scan stack for FOR/GOSUB
A3B8; Move memory
A3FB; Check stack depth
A408; Cneck memory space
A435; °out of memory°
A437; Error routine
A469; BREAK entry
A474; °ready.°
A480; Ready for Basic
A49C; Handle new line
A533; Re-chain lines
A560; Receive input line
A579; Crunch tokens
A613; Find Basic line
A642; Perform [NEW]
A65E; Perform [CLR]
A68E; Back up text pointer
A69C; Perform [LIST]
A742; Perform [FOR]
A7ED; Execute statement
A81D; Perform [RESTORE]
A82C; Break
A82F; Perform [STOP]
A831; Perform [END]
A857; Perform [CONT]
A871; Perform [RUN]
A883; Perform [GOSUB]
A8A0; Perform [GOTO]
A8D2; Perform [RETURN]
A8F8; Perform [DATA]
A906; Scan for next statement
A928; Perform [IF]
A93B; Perform [REM]
A94B; Perform [ON]
A96B; Get fixed point number
A9A5; Perform [LET]
AA80; Perform [PRINT#]
AA86; Perform [CMD]
AAA0; Perform [PRINT]
AB1E; Print string from (y.a)
AB3B; Print format character
AB4D; Bad input routine
AB7B; Perform [GET]
ABA5; Perform [INPUT#]
ABBF; Perform [INPUT]
ABF9; Prompt & input
AC06; Perform [READ]
ACFC; Input error messages
AD1E; Perform [NEXT]
AD78; Type match check
AD9E; Evaluate expression
AEA8; Constant - pi
AEF1; Evaluate within brackets
AEF7;    °)°
AEFF;    comma..
AF08; Syntax error
AF14; Check range
AF28; Search for variable
AFA7; Setup FN reference
AFE6; Perform [OR]
AFE9; Perform [AND]
B016; Compare
B081; Perform [DIM]
B08B; Locate variable
B113; Check alphabetic
B11D; Create variable
B194; Array pointer subrtine
B1A5; Value 32768
B1B2; Float-fixed
B1D1; Set up array
B245; °bad subscript°
B248; °illegal quantity°
B34C; Compute array size
B37D; Perform [FRE]
B391; Fix-float
B39E; Perform [POS]
B3A6; Check direct
B3B3; Perform [DEF]
B3E1; Check fn syntax
B3F4; Perform [FN]
B465; Perform [STR$]
B475; Calculate string vector
B487; Set up string
B4F4; Make room for string
B526; Garbage collection
B5BD; Cneck salvageability
B606; Collect string
B63D; Concatenate
B67A; Build string to memory
B6A3; Discard unwanted string
B6DB; Clean descriptor stack
B6EC; Perform [CHR$]
B700; Perform [LEFT$]
B72C; Perform [RIGHT$]
B737; Perform [MID$]
B761; Pull string parameters
B77C; Perform [LEN]
B782; Exit string-mode
B78B; Perform [ASC]
B79B; Input byte paramter
B7AD; Perform [VAL]
B7EB; Parameters for POKE/WAIT
B7F7; Float-fixed
B80D; Perform [PEEK]
B824; Perform [POKE]
B82D; Perform [WAIT]
```

```
B849; Add 0.5
B850; Subtract-from
B853; Perform [subtract]
B86A; Perform [add]
B947; Complement FAC#1
B97E; °overflow°
B983; Multiply by zero byte
B9EA; Perform [LOG]
BA2B; Perform [multiply]
BA59; Multiply-a-bit
BA8C; Memory to FAC#2
BAB7; Adjust FAC#1/#2
BAD4; Underflow/overflow
BAE2; Multiply by 10
BAF9; +10 in floating pt
BAFE; Divide by 10
BB12; Perform [divide]
BBA2; Memory to FAC#1
BBC7; FAC#1 to memory
BBFC; FAC#2 to FAC#1
BC0C; FAC#1 to FAC#2
BC1B; Round FAC#1
BC2B; Get sign
BC39; Perform [SGN]
BC58; Perform [ABS]
BC5B; Compare FAC#1 to mem
BC9B; Float-fixed
BCCC; Perform [int]
BCF3; String to FAC
BD7E; Get ascii digit
BDC2; Print °IN..°
BDCD; Print line number
BDDD; Float to ascii
BF16; Decimal constants
BF3A; TI constants
BF71; Perform [SQR]
BF7B; Perform [power]
BFB4; Perform [negative]
BFED; Perform [EXP]
E043; Series eval 1
E059; Series eval 2
E097; Perform [RND]
E0f9; ?? breakpoints ??
E12A; Perform [SYS]
E156; Perform [SAVE]
E165; Perform [VERIFY]
E168; Perform [LOAD]
E1BE; Perform [OPEN]
E1C7; Perform [CLOSE]
E1D4; Parameters for LOAD/SAVE
E206; Check default parameters
E20E; Check for comma
E219; Parameters for open/close
E264; Perform [COS]
E26B; Perform [SIN]
E2b4; Perform [TAN]
E30E; Perform [ATN]
E37B; Warm restart
E394; Initialize
E3A2; CHRGET for zero page
E3BF; Initialize Basic
E447; Vectors for $300
E453; Initialize vectors
E45F; Power-up message
E500; Get I/O address
E505; Get screen size
E50A; Put/get row/column
E518; Initialize I/O
E544; Clear screen
E566; Home cursor
E56C; Set screen pointers
E5A0; Set I/O defaults
E5B4; Input from keyboard
E632; Input from screen
E684; Quote test
E691; Setup screen print
E6B6; Advance cursor
E6ED; Retreat cursor
E701; Back into previous line
E716; Output to screen
E87C; Go to next line
E891; Perform §return†
E8A1; Check line decrement
E8B3; Check line increment
E8CB; Set color code
E8DA; Color code table
E8EA; Scroll screen
E965; Open space on screen
E9C8; Move a screen line
E9E0; Synchronize color transfer
E9F0; Set start-of-line
E9FF; Clear screen line
EA13; Print to screen
EA24; Synchronize color pointer
EA31; Interrupt - clock etc
EA87; Read keyboard
EB79; Keyboard select vectors
EB81; Keyboard 1 - unsnifted
EBC2; Keyboard 2 - shifted
EC03; Keyboard 3 - °comm°
EC44; Graphics/text contrl
EC4F; Set graphics/text mode
EC78; Keyboard 4
ECB9; Video chip setup
ECE7; Shift/run equivalent
ECF0; Screen ln address low
ED09; Send °talk°
ED0C; Send °listen°
ED40; Send to serial bus
EDB2; Serial timeout
EDB9; Send listen SA
EDBE; Clear ATN
EDC7; Send talk SA
EDCC; Wait for clock
EDDD; Send serial deferred
EDEF; Send °untalk°
```

```
EDFE; Send °unlisten°
EE13; Receive from serial bus
EE85; Serial clock on
EE8E; Serial clock off
EE97; Serial output °1°
EEA0; Serial output °0°
EEA9; Get serial in & clock
EEB3; Delay 1 ms
EEBB; RS-232 send
EF06; Send new RS-232 byte
EF2E; No-DSR error
EF31; No-CTS error
EF3B; Disable timer
EF4A; Compute bit count
F8D0; Check tape stop
F8E2; Set read timing
F92C; Read tape bits
FA60; Store tape chars
FB8E; Reset pointer
FB97; New character setup
FBA6; Send transition to tape
FBC8; Write data to tape
FBCD; IRQ entry point
FC57; Write tape leader
FC93; Restore normal IRQ
FCB8; Set IRQ vector
FCCA; Kill tape motor
FCD1; Check r/w pointer
FCDB; Bump r/w pointer
FD50; Initialize system constnts
FCE2; Power reset entry
FD02; Check 8-rom
FD10; 8-rom mask
FD15; Kernal reset
FD1A; Kernal move
FD30; Vectors
FD9B; IRQ vectors
FDA3; Initialize I/O
FDDD; Enable timer
FDF9; Save filename data
FE00; Save file details
FE07; Get status
FE18; Flag status
FE1C; Set status
FE21; Set timeout
FE25; Read/set top of memory
FE27; Read top of memory
FE2D; Set top of memory
FE34; Read/set bottom of memory
FE43; NMI entry
FE66; Warm start
FEB6; Reset IRQ & exit
FEBC; Interrupt exit
FEC2; RS-232 timing table
```

```
FED6; NMI RS-232 in
FF07; NMI RS-232 out
FF43; Fake IRQ
FF48; IRQ entry
FF81; Jumbo jump table
FFFA; Hardware vectors
EF59; RS232 receive
EF7E; Setup to receive
EFC5; Receive parity error
EFCA; Receive overflow
EFCD; Receive break
EFD0; Framing error
EFE1; Submit to RS232
F00D; No-DSR error
F017; Send to RS232 buffer
F04D; Input from RS232
F086; Get from RS232
F0A4; Check serial bus idle
F0BD; Messages
F12B; Print if direct
F13E; Get..
F14E;    ..from RS232
F157; Input
F199; Get.. tape/serial/rs232
F1CA; Output..
F1DD;    ..to tape
F20E; Set input device
F250; Set output device
F291; Close file
F30F; Find file
F31F; Set file values
F32F; Abort all files
F333; Restore default I/O
F34A; Do file open
F3D5; Send SA
F409; Open RS232
F49E; Load program
F5AF; °searching°
F5C1; Print filename
F5D2; °loading/verifying°
F5DD; Save program
F68F; Print °saving°
F69B; Bump clock
F6BC; Log PIA key reading
F6DD; Get time
F6E4; Set time
F6ED; Check stop key
F6FB; Output error messages
F72D; Find any tape headr
F76A; Write tape header
F7D0; Get buffer address
F7D7; Set buffer start/end pointers
F7EA; Find specific header
F80D; Bump tape pointer
F817; °press play..°
F82E; Check tape status
F838; °press record..°
F841; Initiate tape read
F864; Initiate tape write
F875; Common tape code
```

A BYTE OF WHIMSEY...

Actual Poster Size 29" wide x 24" deep

A funny thing happened here a few months ago...this fellow walked in with this very unsophisticated art he wanted us to use on the April/May cover of the COMMODORE Magazine. Gee, how could we tell him that it wasn't the "image" we wanted to project...after all, our computers might be *FUN*, but we have some very professional people using these things — doctors, lawyers, even indian chiefs. The powers that be took one look at it and said "Hey, you can't put a lot of pixies on our cover and expect people to take you seriously... there's a pixie playing frisbee with one of our disks, and another pixie floating down a waterfall with one of our business machines, there's a funny looking toad fooling around with a VIC, there's even some pixie who thinks he can access a data base from his treehouse... our readers want to see *PEOPLE* — people looking very serious, very productive, learning all sorts of great things, not a bunch of silly pixies. Well, we took a closer look, and ran it anyway...In fact, we even had it printed as a big four-color poster to hang in your *VERY* serious office, institution of learning, or, your own treehouse... So if you have just a BYTE OF WHIMSEY in you, fill out the attached coupon and send us your check or money order.

Return to:
COMMODORE MAGAZINE
COMMODORE BUSINESS MACHINES INC.
The Meadows, 487 Devon Park Drive
Wayne, Pa. 19087

Name _____

Address _____

City _____

State _____ Zip _____

Please enclose a check or money order for $5.00 for each poster ordered.

QUANTITY ____ TOTAL ENCLOSED ____

Dealer Inquiries Welcome

commodore COMPUTER

PROJECTIONS & REFLECTIONS

The leaves are starting to turn colors, my football team is 2-0, and our new Commodore 64 software products seem to be pleasing just about everyone. Just as Fall is a time for change, it is time for us to start thinking about software for our new 'P' and 'B' series computers. But before we do, I would like to take this opportunity to thank all of the software vendors who have helped make the Commodore 64 a successful product. Keep up the good work!

Now, for all of you software people who have not started to create software for the 64, what are you waiting for? Not only is this machine a credible, cost effective computer, but it is generating quite a substantial base on which you can sell your product. I am always looking for new, personal software products. If there is anything I can do to help expedite your software products to THE computer, please let me know.

I would also like to give due credit to some of our 'in house' people who are responsible for products like the PET emulator, Assembler development system, The WORD/NAME MACHINES, Mail Mate, WINI backup and restore system, and countless utilities. You know, the 64 has generated enough excitement around here that I, too, have even dusted off my old coding pencil to bring out a couple of products. Look out!

Speaking of giving credit, you must see the new book called *PROGRAMMING THE PET/CBM* by Raeto Collin West, from COMPUTE! I can't say enough about it. It is the best book I have ever seen for the Commodore product line. Not only is it complete, but correct as well. BUY IT.

Now, about the new 'P' and 'B' series computers. Enhanced 4.0 BASIC, 128 to 256K, color, sound, 40 and 80 column display, IEEE and RS232 . . . something for everyone. I feel that the new 'P' series is, without hesitation, the best buy on the market today (or tomorrow for that matter). The 'P' series includes all of the color, sound, and graphics capabilities that are inside the 64, as well as having up to 256k and built in IEEE, RS232, and game ports. So, for the person who is serious about having a machine that provides great color and sound, along with being able to use the larger peripherals for business processing, the 'P' series is it! There is no other way to describe the entire 'B' series except that it means 'business'. For the serious systems designer, this machine with a hard disk provides a very competitive, cost effective alternative to a lot of the 'mini' computers out on the market today.

So, think software. Commodore is doing everything we can to provide you with the machines that create the base for your software products. Write to us—let us know what you are doing and thinking along the lines of software.

Look, up in the sky! It's a bird, it's the Goodyear blimp, no it's . . . SOFTWARE MAN. Watch for him in the next issue.

Paul Goheen
Software Products Manager

COMMODORE 64 COMMODORE 64 COMMODORE 64

Continued from page 15.

LL
1. P.77, BYTE, second sentence
2. . . . you can actually have a total of 255 different combinations . . .
3. . . . you can actually have a total of 256 different combinations . . .

MM
1. P.80, STRUCTURE OF A SOUND PROGRAM
2. VOLUME, WAVEFORM CONTROL . . . EACH NOTE you play.
3. VOLUME, ATTACK/DECAY, SUSTAIN/RELEASE(ADSR), WAVEFORM CONTROL and HIGH FREQUENCY/LOW FREQUENCY. The first three settings are usually set ONCE at the beginning of your program. The high and low frequency settings must be set for EACH NOTE you play. The waveform control starts and stops each note.

NN
1. P.80, SAMPLE SOUND PROGRAM
2. 1. Set VOLUME at highest setting:
   10 POKE54296,15
3. First clear sound chip
   5 FORL=54272TO54296:POKEL,0:NEXT
   1. Set VOLUME at highest setting:
   10POKE54296,15

OO
1. P.80, SAMPLE SOUND PROGRAM
2. 2. Set ATTACK/DECAY levels
3. Set ATTACK/DECAY rates

PP
1. P.80, SAMPLE SOUND PROGRAM
2. 3. Set SUSTAIN/RELEASE rate to prolong note at a certain volume and release it.
3. 3. Set SUSTAIN/RELEASE to define level to prolong note and rate to release it.

QQ
1. P.80, SAMPLE SOUND PROGRAM
2. 40 POKE54273,17:POKE5427237
3. 40 POKE54273,17:POKE54272,37

RR
1. P.81, SAMPLE SOUND PROGRAM
2. 5. Set WAVEFORM to . . .
3. 5. Start WAVEFORM with . . .

SS
1. P.81, 7. WAVEFORM . . .
2. 7. Turn off the WAVEFORM CONTROL and ADSR settings.
   70 POKE 54276,0:POKE 54277,0:POKE54278,0
3. 7. Turn off note. 70 POKE54276,16

TT
1. P.81, MAKING MUSIC ON YOUR COMMODORE 64, program
2. 5 REM MUSICAL SCALE
   10 POKE 54296,15
   . . .
   50 READ A
3. 5 REM MUSICAL SCALE
   7 FORL=54272TO54296:POKEL,0:NEXT
   10 POKE 54296,15
   20 POKE 54277,7:POKE54278,133
   Sets Attack/Decay, Sustain/Release level (each note)
   50 READ A

UU
1. P.82, Sample program
2. 60 READ B
   . . .
   100 GOTO20
3. 60 READ B
   70 IFB=−1THENEND
   80 POKE 54273,A:POKE54272,B
   85 POKE 54276,17
   Start note
   90 FORT=1TO250:NEXT:POKE54276,16
   Let it play then stop note
   95 FORT=1TO50:NEXT
   Time for release
   100 GOTO20

VV
1. P.82, Last complete paragraph
2. To change the sound to a "harpsichord," change Line 30 . . . and RUN
3. To change the sound to a "harpsichord," change Line 85 to read POKE54276,33 and line 90 to read FORT=1TO250:NEXT:POKE54276,32 and RUN

WW
1. P.82, Last complete paragraph
2. . . . Changing the WAVEFORM can drastically change the sound by the COMMODORE 64 . . .
3. . . . Changing the WAVEFORM can drastically change the sound produced by the COMMODORE 64 . . .

XX
1. P.82, last sentence and POKE statement
2. . . . to a more "banjo" sound try changing line 20 to read:
   20 POKE54277,3.
3. . . . to a more "banjo" sound try changing lines 20 and 30 to read:
   20 POKE54277,3
   30 POKE54278,0 <——Sets no sustain for banjo effect.

YY
1. P.83, 1. VOLUME
2. . . . POKE 54296,15. The volume setting ranges from 0 to 15 but you'll use 15. The volume. . .
3. . . . POKE54296,15. The volume. . .

ZZ
1. P.83, 2. ADSR and WAVEFORM CONTROL SETTING
2. A sample waveform . . .
3. A sample waveform start setting . . .

AAA
1. P.83, 2. ADSR and WAVEFORM CONTROL SETTING
2. . . . the second number (17) represents a triangular . . .
3. . . . the second number (17) represents the start for a triangular . . .

BBB
1. P.83, ADSR and WAVEFORM CONTROL SETTINGS
2.

CONTROL SETTING	TRIANGLE	SAWTOOTH	PULSE	NOISE
. . .				

3.

	CONTROL REGISTER	Note Start/Stop Numbers TRIANGLE	SAWTOOTH	PULSE	NOISE
VOICE 1	54276	17/16	33/32	65/64	129/128
VOICE 2	54283	17/16	33/32	65/64	129/128
VOICE 3	54920	17/16	33/32	65/64	129/128

CCC
1. P.84, 2. ADSR and WAVEFORM CONTROL SETTING
2. . . . look at line 20 . . . this gave the scale a "harpsichord" effect.
3. . . . look at lines 85 and 90 in the musical scale program. In this program, immediately after setting the frequency in line 80, we set the CONTROL SETTING for VOICE 1 in Line 85 by POKEing 54276,17. This turned on the CONTROL for VOICE 1 and set it to a TRIANGLE WAVEFORM (17). In line 70 we POKE 54276,16, stopping the note. Later, we changed the waveform start setting from 17 to 33 to create a SAWTOOTH WAVEFORM and this gave the scale a "harpsichord" effect.

DDD
1. P.84, 3. ATTACK/DECAY SETTING.
2. The DECAY is . . . level back to zero.
3. The DECAY is the rate at which the note/sound falls from its highest volume level back to the SUSTAIN level.

EEE
1. P.84, 3. ATTACK/DECAY SETTING
2. . . . YOU CAN COMBINE ATTACK AND DECAY SETTINGS . . .
3. . . . YOU MUST COMBINE ATTACK AND DECAY SETTINGS . . .

FFF 1. P.85, 3. ATTACK/DECAY SETTING
2. 10 PRINT"HIT ANY KEY"
. . .
90 GOTO20
3. 5 FOR L=54272TO54296:POKEL,0:NEXT
10 PRINT"HIT ANY KEY"
20 POKE54296,15
30 POKE54277,64
40 POKE54273,17:POKE54272,37
60 GETK$:IFK$=""THEN60
70 POKE54276,17:FORT=1TO200:NEXT
80 POKE54276,16:FORT=1TO50:NEXT
90 GOTO20

GGG 1. P.85, 4. SUSTAIN/RELEASE SETTING
2. . . . Any note or sound can be sustained at its volume peak . . . you can even set the sustain level at its maximum (240) with no release to make a note play "indefinitely".
3. . . . Any note or sound can be sustained at any one of 16 levels.

HHH 1. P.86, 4. SUSTAIN/RELEASE SETTING
2. . . . how long the note will be held at peak volume . . .
3. . . . how long the note will be held at SUSTAIN volume . . .

III 1. P.86, 4. SUSTAIN/RELEASE SETTING
2. . . . combine a HIGH SUSTAIN LEVEL with a LOW RELEASE LEVEL . . .
3. . . . combine a HIGH SUSTAIN LEVEL with a LOW RELEASE RATE . . .

JJJ 1. P.86, Sample program
2. 10 PRINT"HIT ANY KEY"
. . .
90 GOTO20
3. 5 FORL=54272TO54296:POKEL,0:NEXT
10 POKE54296,15
20 POKE54277,64
30 POKE54278,128
40 POKE54273,17:POKE54272,37
50 PRINT"HIT ANY KEY"
60 GETK$:IFK$=""THEN60
70 POKE54276,17:FORT=1TO200:NEXT
80 POKE54276,16:FORT=1TO50:NEXT
90 GOTO60

KKK 1. P.86, Paragraph following sample program
2. In Line 45, we tell the computer . . . the "count" in Line 70.
3. In Line 30, we tell the computer to SUSTAIN the note at a HIGH SUSTAIN LEVEL (128 from chart above) . . . after which the tone is released in Line 80. You can vary the duration of a note by changing the "count" in Line 70. To see the effect of using the release function try changing Line 30 to POKE54278,89 (SUSTAIN = 80, RELEASE = 9).

LLL 1. P.87, Sample program
2. 10 V=54296:W=54276:A=54277:
. . .
40 POKEH,0:POKEL,0:POKEW,0
3. 5 FORL=54272TO54296:POKEL,0:NEXT
10 V=54296:W=54276:A=54277:S=54278:
H=54273:L=54272
20 POKEV,15:POKEA,190:POKES,89
POKE volume, attack/decay, sustain/release
30 POKEH,34:POKEL,75
POKE hi/lo freq. notes
40 POKEW,33:FORT=1TO200:NEXT
start note, let it play
50 POKEW,32
stop note

MMM 1. P.88, MICHAEL ROW THE BOAT ASHORE—1 MEASURE
2. 5 V=54296:W=54276:A=54277:
HF=54273:LF=54272:S=54278:PH
. . .
80 FORT=1TOD:NEXT:POKEHF,0:
POKEW,0
3. 2 FORL=54272TO54296:POKEL,0:NEXT
5 V=54296:W=54276:A=54277:HF=54273:
LF=54272:S=54278:PH=54275:PL=54274
10 POKEV,15:POKEA,88:POKEPH,15:
POKEPL,15:POKES,89
20 READH:IFH=-1THENEND
30 READL
40 READD
60 POKEHF,F:POKELF,L:POKEW,65
80 FORT=1TOD:NEXT:POKEW,64
85 FORT=1TO50:NEXT

NNN 1. P.127, RND(X) formula
2. N = INT (RND(1)*Y) + X
3. N = RND(1)*(Y−X) + X

OOO 1. P.129, OTHER FUNCTIONS, FRE(X)
2.
3. . . . the value of X. Note that FRE(X) will read out n negative numbers if the number of unused bytes is over 32K.

PPP 1. P.132, Paragraph 3
2. From BASIC, POKE 53272,29 will switch to upper case mode and POKE 53272,31 switches to lower case.
3. From BASIC, POKE 53272,21 will switch to upper case mode and POKE 53272,23 switches to lower case.

QQQ 1. P.136, numbers 129,149-155
2.
129
149
150
151
152
153
154
155
3.

same as 97	129
same as 117	149
same as 118	150
same as 119	151
same as 120	152
same as 121	153
same as 122	154
same as 123	155

RRR 1. P.141, PINOUTS, Control Port 1 & Control Port 2
2. Note . . . MAX. 100mA
3. Note . . . MAX. 50mA

SSS 1. P.142, Cartridge Expansion Slot
2.

Pin	Type
22	GND
21	CD0
20	CD1
19	CD2
18	CD3
17	CD4
16	CD5
15	CD6
14	CD7
13	DMA
12	BA

3.

Pin	Type
12	BA
13	DMA
14	D7
15	D6
16	D5
17	D4
18	D3
19	D2
20	D1
21	D0
22	GND

TTT 1. P.142, Cartridge Expansion Slot

2.

Pin	Type
11	ROML
10	1/02
9	EXROM
8	GAME
7	1/01
6	Dot Clock
5	CR/W
4	IRQ
3	+5V
2	+5V
1	GND

3.

Pin	Type
1	GND
2	+5V
3	+5V
4	IRQ
5	R/W
6	Dot Clock
7	I/O 1
8	GAME
9	EXROM
10	I/O 2
11	ROML

UUU 1. P.142, Cartridge Expansion Slot

2.

Pin	Type
Z	GND
Y	CA0
X	CA1
W	CA2
V	CA3
U	CA4
T	CA5
S	CA6
R	CA7
P	CA8
N	CA9

3.

Pin	Type
N	A9
P	A8
R	A7
S	A6
T	A5
U	A4
V	A3
W	A2
X	A1
Y	A0
Z	GND

VVV 1. P.142, Cartridge Expansion Slot

2.

Pin	Type
M	CA10
L	CA11
K	CA12
J	CA13
H	CA14
F	CA15
E	S02
D	NMI
C	RESET
B	ROMH
A	GND

3.

Pin	Type
A	GND
B	ROMH
C	RESET
D	NMI
E	S 02
F	A15
G	A14
H	A14
J	A13
K	A12
L	A11
M	A10

WWW.

```
1. P.142, Cartridge Expansion Slot
2.  1  2  3  4  5  6  7  8  9  10 11 12 13 14 15 16 17 18 19 20 21 22

    A  B  C  D  E  G  H  J  K  L  M  N  P  R  S  T  U  V  W  X  Y  Z
3.  22 21 20 19 18 17 16 15 14 13 12 11 10  9  8  7  6  5  4  3  2  1

    Z  Y  X  W  V  U  T  S  R  P  N  M  L  K  J  H  G  E  D  C  B  A
```

XXX 1. P.147, Program
2. 220 K$="Q2W3ER5T6Y7UI9O0P@—*↑↑"
3. 220 K$="Q2W3ER5T6Y7UI9O0P@—*+£ I↑"

YYY 1. P.150, FORMULA TOO COMPLEX
2. . . . work with.
3. . . . work with or a formula has too many parentheses.

ZZZ 1. P.154, MUSIC NOTE VALUES
2. . . .

3. Filter Settings

location	*contents*
54293	low cutoff frequency (0-7)
54294	high cutoff frequency (0-255)
54295	resonance (bits 4-7)
	filter voice 3 (bit 2)
	filter voice 2 (bit 1)
	filter voice 1 (bit 0)
54296	high pass (bit 6)
	band pass (bit 5)
	low pass (bit 4)
	volume (bits 0-3)

AAAA

1. P.157, APPENDIX O, SPRITE REGISTER MAP
2. . . . . .
. . . . . .
. . . . . .
17 11 RC8 ECM BMM BLNK RESL YSCL2 YSCL1 YSCL0
3. . . . . .
. . . . . .
. . . . . .
17 11 RC8 ECM BMM BLNK RSEL YSCL2 YSCL1 YSCL0 Y SCROLL/MODE

BBBB

1. P.158, APPENDIX O, SPRITE REGISTER MAP
2. . . . . .
. . . . . .
. . . . . .
22 16 N.C. N.C. RST MCM CSEL XSCL2 XSCL1 XSCL0
. . . . . .
. . . . . .
. . . . . .
24 18 VS13 VS12 VS11 CB13 CB12 CB11 CB10 N.C. SCREEN Char Mem
3. . . . . .
. . . . . .
. . . . . .
22 16 N.C. N.C. RST MCM CSEL XSCL2 XSCL1 XSCL0 X SCROLL MODE
. . . . . .
. . . . . .
. . . . . .
24 18 VS13 VS12 VS11 VS10 CB13 CB12 CB11 N.C. SCREEN Char Mem

CCCC 1. P.161, VOICE NUMBER 2, WAVEFORM POKE
2. 54288
3. 54283

DDDD 1. P.161, VOICE NUMBER 3, SUSTAIN/RELEASE POKE
2. 54286
3. 54285

EEEE 1. QUICK REFERENCE CARD, SYSTEM COMMANDS
2. WAIT X,Y,Z Program waits until contents of location X, when FORed with Z and ANDed with Y, is nonzero.
3. WAIT X,Y,Z Program waits until contents of location X, when EORed with Z and ANDed with Y, is nonzero.

Advertisers' Index

ABM Products 27
R.J. Brachman Assoc., Inc. 76
CMS Software Systems 80
Cascade Computerware 33
The Code Works 34
CompuSense 77, 81, 93, 95
Computer Case Company 83
Computer Marketing Services 67,69, 71
Computer Mat 33
Computer Specialties 42
Connecticut microComputer, Inc. 27
Cow Bay Computing 88
Cyberia, Inc. 28
Eastern House Software 4, 9, 19, 86
ETC 73
French Silk 66
Hewlett Packard 79
Human Engineered Software . . . . . IBC
Info Designs . . . . . IFC
INI, Inc. 76
Interlink 10
K8 Software 78
Leading Edge Products, Inc. . . . . OBC
LemData Products 37
Madison Computer 6
Metron Computer Systems 18
Microphys Programs 56
Microsignal 65
Micro Spec Ltd. 62
Micro World Electronix, Inc. 88
MIS 82
MSC Software 18
Peripherals Plus 11
Professional Software 3, 7
Skyles Electric 38
Spellmaster Systems 54
Toronto PET User Group 34
TOTL Software 74
TSASA, Inc. 69
United Microware Industries, Inc. 13
Wunderware 62

commodore

Your VIC 20 never had it so good!

VIC 20 owners rejoice! HES presents a complete range of software from our exciting series of games to our professional group of utility and language programs.

Our new cartridge programs include: HES MON, an indispensible monitor for assembly language programmers; HES WRITER, a word processing program; Turtle Graphics, a fun and easy way to learn computer programming; and VIC FORTH, a powerful language that is many times faster than BASIC, yet easier to use than assembly language.

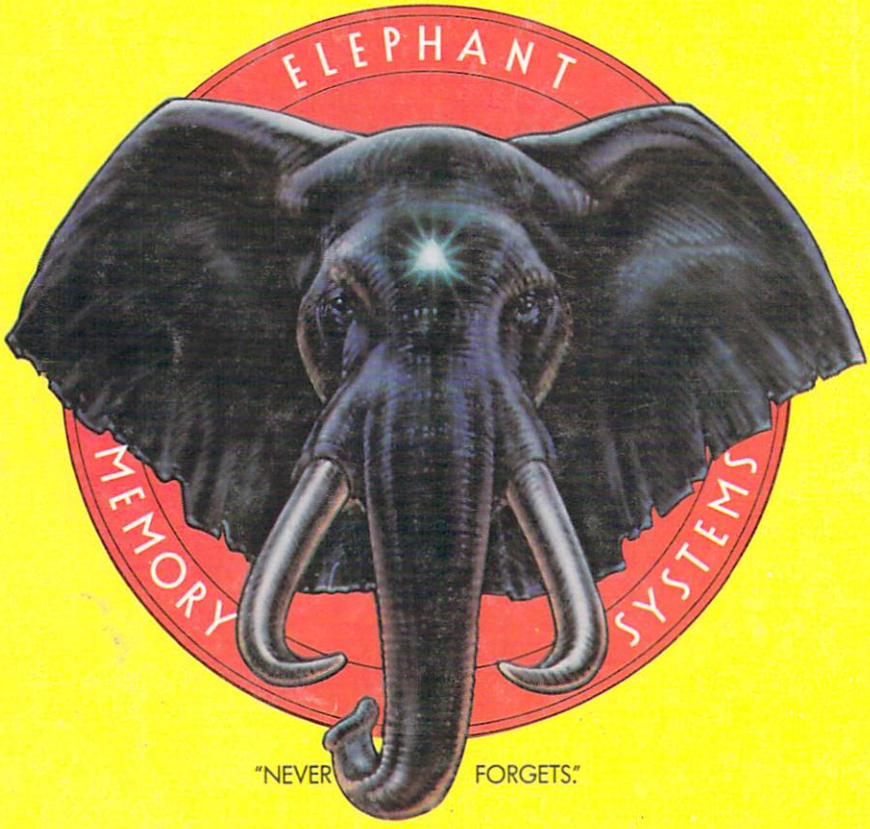
HES is committed to offering high-quality, well-documented computer programs on a continual basis. Look for our cartridge and cassette based software at your local dealer.

HES

Human Engineered Software
71 Park Lane
Brisbane, California 94005
Telephone 415-468-4110
Circle # 53 on Reader Service Card

VIC 20 is a registered TM of Commodore

REMEMBER:



MORE THAN JUST ANOTHER PRETTY FACE.

Says who? Says ANSI.

Specifically, subcommittee X3B8 of the American National Standards Institute (ANSI) says so. The fact is all Elephant™ floppies meet or exceed the specs required to meet or exceed all their standards.

But just who is "subcommittee X3B8" to issue such pronouncements?

They're a group of people representing a large, well-balanced cross section of disciplines—from academia, government agencies, and the computer industry. People from places like IBM, Hewlett-Packard, 3M, Lawrence Livermore Labs, The U.S. Department of Defense, Honeywell and The Association of Computer Programmers and Analysts. In short, it's a bunch of high-caliber nitpickers whose mission, it seems, in order to make better disks for consumers, is also to make life miserable for everyone in the disk-making business.

How? By gathering together periodically (often, one suspects, under the full moon) to concoct more and more rules to increase the quality of flexible disks. Their most recent rule book runs over 20 single-spaced pages—listing, and insisting upon—hundreds upon hundreds of standards a disk must meet in order to be blessed by ANSI. (And thereby be taken seriously by people who take disks seriously.)

In fact, if you'd like a copy of this formidable document, for free, just let us know and we'll send you one. Because once you know what it takes to make an Elephant for ANSI . . .

We think you'll want us to make some Elephants for you.

ELEPHANT.™ HEAVY DUTY DISKS.

For a free poster-size portrait of our powerful pachyderm, please write us.
Distributed Exclusively by Leading Edge Products, Inc., 225 Turnpike Street, Canton, Massachusetts 02021
Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.

Circle #54 on Reader Service Card