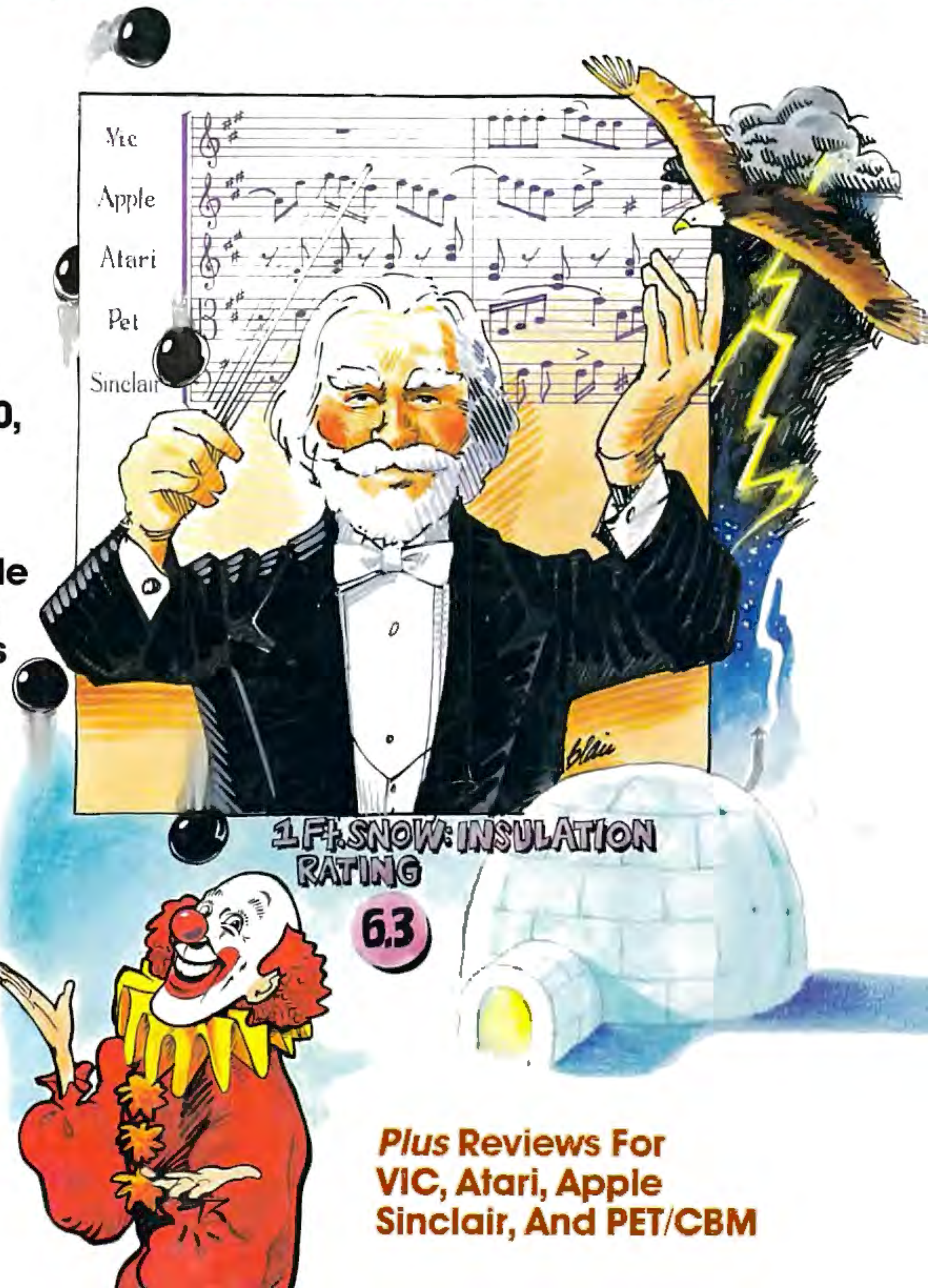
Sound Synthesis And The Personal Computer — Past, Present, And Future

COMPUTE!

$2.50
January
1983
Issue 32
Vol. 5, No. 1
63379 £1.85 in UK

The Leading Magazine Of Home, Educational, And Recreational Computing

Music, Sound, And The Personal Computer

Music And Sound For VIC, Apple, Atari, PET/CBM, And The Sinclair/Timex

The Juggler, Thunderbird: **Two Exciting Game Programs For VIC-20, Atari, And TRS-80 Color Computer**

Writing Transportable BASIC Programs For Personal Computers

A New Monthly Column: **Programming The TI-99/4A**

A Home Energy Calculator For Atari, Apple, VIC-20, PET/CBM, And Commodore 64

Plus **Reviews For VIC, Atari, Apple Sinclair, And PET/CBM**

0 74470 63379 01

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.
51 Fremont Street
Needham, MA 02194
(617) 444-5224
TELEX: 95 1579

TM WordPro is a Registered Trademark of Professional Software, Inc. WordPro was written by Steve Punter.
All specifications subject to change without notice

Introducing Snooper Troops™ detective series. Educational games that turn ordinary homes into Sherlock homes.

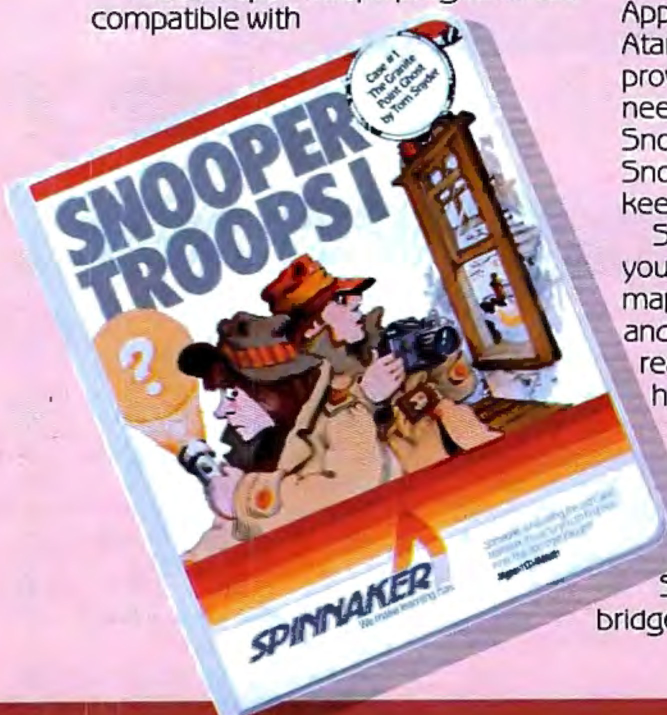
Where can you find educational games that your kids will really enjoy playing?

Elementary, my dear Watson. From Spinnaker.

Our Snooper Troops detective games are fun, exciting and challenging. And best of all, they have real educational value. So while your kids are having fun, they're learning.

As a Snooper Trooper, your child will have a great time solving the mysteries. But it will take some daring detective work. They'll have to question suspects, talk to mysterious agents, and even search dark houses to uncover clues.

The Snooper Troops programs are compatible with Apple,® IBM® and Atari® computers and provide your kids with everything they need: a SnoopMobile, a wrist radio, a SnoopNet computer, a camera for taking Snoopshots and even a notebook for keeping track of information.



Snooper Troops detective games help your children learn to take notes, draw maps, organize and classify information and they help develop vocabulary and reasoning skills. All while your kids are having a good time.

So if you want to find educational games that are really fun, here's a clue: Snooper Troops games are available at your local software store, or by writing to: Spinnaker Software, 215 First Street, Cambridge, MA 02142.

Spinnaker's early learning games will help make your children as smart as you tell everyone they are.

Your kids are pretty smart. After all, they're *your* kids.

Spinnaker can help make them even smarter. With a line of educational software that kids love to play.

Spinnaker games make the computer screen come to life with full color graphics and sound. And they're fun. Lots of fun. But they also have real educational value.

Some of our games help exercise your child's creativity. Others improve memory and concentration. While others help to improve your child's writing, vocabulary, and spelling skills.

And every Spinnaker game provides familiarity with the computer and helps your children feel friendly with the computer. Even if they've never used a computer before.

And Spinnaker games are compatible with the most popular computers: Apple,® Atari® and IBM.®

Our newest game, KinderComp™ (Ages 3-8) is a collection of learning exercises presented in a fun and exciting manner.

Rhymes and Riddles™ (Ages 4-9) is a letter guessing game featuring kids' favorite riddles, famous sayings and nursery rhymes.

Story Machine™ (Ages 5-9) lets children write their own stories and see them come to life on the screen.

And FACEMAKER™ lets your children create their own funny faces and make them wink, smile, wiggle ears (not your kids' ears, the ears on the screen), etc.

And we're introducing new games all the time.

So look for Spinnaker games at your local software retailer, or by writing to: Spinnaker Software, 215 First St., Cambridge, MA 02142. And show your kids how smart their parents really are.

SPINNAKER™
We make learning fun.

January 1983 Vol. 5, No. 1

FEATURES

EDUCATION AND RECREATION

REVIEWS

COLUMNS AND DEPARTMENTS

THE JOURNAL

NOTE: See page 222 before typing in programs.

GUIDE TO ARTICLES AND PROGRAMS

AP Apple, **AT** Atari, **P** PET/CBM, **V** VIC-20, **O** OSI, **C** Radio Shack Color Computer, **64** Commodore 64, **ZX** Sinclair ZX-81, **•** All or several of the above.

COMPUTE! The Journal for Progressive Computing (USPS: 537250) is published 12 times each year by Small System Services, Inc., P.O. Box 5406, Greensboro, NC 27403 USA. Phone: (919)275-9809. Editorial Offices are located at 625 Fulton Street, Greensboro, NC 27403. Domestic Subscriptions: 12 issues, $20.00. Send subscription orders or change of address (P.O. form 3579) to Circulation Dept., **COMPUTE!** Magazine, P.O. Box 5406, Greensboro, NC 27403. Second class postage paid at Greensboro, NC 27403 and additional mailing offices. ISSN 0194-357X.

EDITOR'S NOTES

Announcing A Significant New Magazine From COMPUTE! Publications. We've promised that 1983 would be an exciting year and are now willing to divulge one of the reasons why. *The Commodore Gazette*™ will premiere as a monthly in the spring of 1983. The Gazette will not impact **COMPUTE!** editorially or alter the current scope of **COMPUTE!**. The Gazette is planned as a laypersons's guide to consumer computing. It will be written for beginning and intermediate level owners and users of the VIC-20, 64, and Ultimax computers. Regular features will include best seller lists for recreational and educational software, reviews, new products, tutorials on home and educational applications, and much more. Written for entertainment as well as education, *The Commodore Gazette,* while appealing to users wishing to learn more about programming and computers, will also have continuing appeal for those who simply want to obtain maximum use from their computers in a non-technical way. Next issue we'll give you full details on the new magazine. **COMPUTE!** will continue to present its normal excellent range of information for the VIC-20 and Commodore 64.

A Call For Editors

COMPUTE! Publications, both our magazine and book publishing divisions, is looking for experienced staff members for our growing editorial needs. If you've been writing for **COMPUTE!**, or if you have meaningful editorial experience, we'd like to see a resumé as soon as possible. We're specifically interested in writers with experience using Atari, VIC-20, and related computer hardware. We are a progressive and growing company, with an excellent working environment and benefits, located in the attractive Piedmont area of central North Carolina. If you're interested, please send a resumé along with work history, salary expectations, and other pertinent information to Kathleen Martinek, Managing Editor, **COMPUTE!** Publications, Post Office Box 5406, Greensboro, NC 27403. Your inquiry will be treated with complete confidence. Mark the envelope "Personal and Confidential," please. Remember to include samples of your writing.

We cannot accept any telephone calls prior to submission of a resume, and will deal only with the individual interested in the position. We do not wish to work with personnel agencies.

Random Asides

You'll notice several enhancements in this issue as part of our continuing quest to better serve our readers. Among these are additions aimed at beginners. "Questions Beginners Ask" and a revised section on using our program listing conventions will become regular features of **COMPUTE!**.... We're setting new records again. Press run for this issue is an astonishing 156,000 magazines. We had to declare October, November, and December *sold-out* within weeks of publication date. It was only a few months ago we were collectively applauding the 100,000 mark... New personal computers are on the way from Mattel and NEC, among others... Atari may be dropping the rumored 600 given recent changes in competitor pricing. Look for a new competitor to the Commodore 64... Magnum publishes a monthly list of the 100 best-selling computer books in the US. It's compiled from industry sources. **COMPUTE! Books** has five titles in the top 100, and *COMPUTE!'s First Book of VIC* is number one for the second month in a row.

Reader Service Cards

A first for **COMPUTE!**. In the back of this issue, you'll find reader service cards for the very first time. Use them to request additional information from advertisers. Simply look up the advertiser in the advertising index and circle the appropriate number. Remember though, if you're in a hurry to contact a particular advertiser, it's probably best to write or call directly.

COMPUTE!'s New Look

As you'll notice as you explore this issue, we've made some subtle and significant changes in the overall design of the magazine. We think you'll find **COMPUTE!** even easier to read and enjoy. Thanks to everyone here for helping implement these changes.

Robert Lock


The Home Accountant.™
The #1 best-seller.
Any home finance package will balance your checkbook. But to become the #1 best-seller you've got to be something special.
The Home Accountant™ is.
It's the only one that prints a net worth statement and a personal finance statement. So you know exactly where you stand financially every day of the year. It will even print your checks, automatically.
Not only that, The Home Accountant™ lets you label every transaction. Just imagine sitting down to do your taxes and having every penny you've spent and earned neatly listed by category—and available at the touch of a button. It's an incredible time-saver.
You can also create bar, line and trend analysis graphs for every category—in color. It's great for realistic budgeting.
Sound amazing? Wait, there's more.
Let's say you write a check to pay your Visa. The Home Accountant™ automatically debits your checking account and credits your Visa account.
And it does this with every one of the two hundred* budget categories: credit cards, checking accounts, money markets, cash, rent checks, insurance payments—you customize your own financial package.
Check out The Home Accountant™ soon. You'll find it does a lot more than simply manage your money.
It manages your money simply.
*The Home Accountant™ is available for the Apple II/IBM Personal Computer/Atari 400/800 Computers/Osborne/TRS 80 Model III/Commodore VIC 64. The actual budget capacities will vary with each computer.
Continental Software
A Division of Arrays, Inc.
It sells the most, because it does the most!
Continental Software Co., 11223 South Hindry Avenue, Los Angeles, California 90045 Telephone (213) 417-3003 • (213) 417-8031

Small System Services, Inc. publishes:

COMPUTE!
The Journal For Progressive Computing

COMPUTE! Books

Corporate office:
625 Fulton Street,
Greensboro, NC 27403 USA

Mailing address: COMPUTE!
Post Office Box 5406
Greensboro, NC 27403 USA

Telephone: 919-275-9809

Coming In February

Special Games Issue:

How The Professionals Program Games

Four Exciting Games: *Mastermaze, Slalom, Copycat, And A Day At The Races*

VIC And PET High Resolution Plotters

Atari SuperFont Plus And Left-handed Joysticks

Apple High Resolution Painter

Color Computer Relocation Calculator

Subscription Information

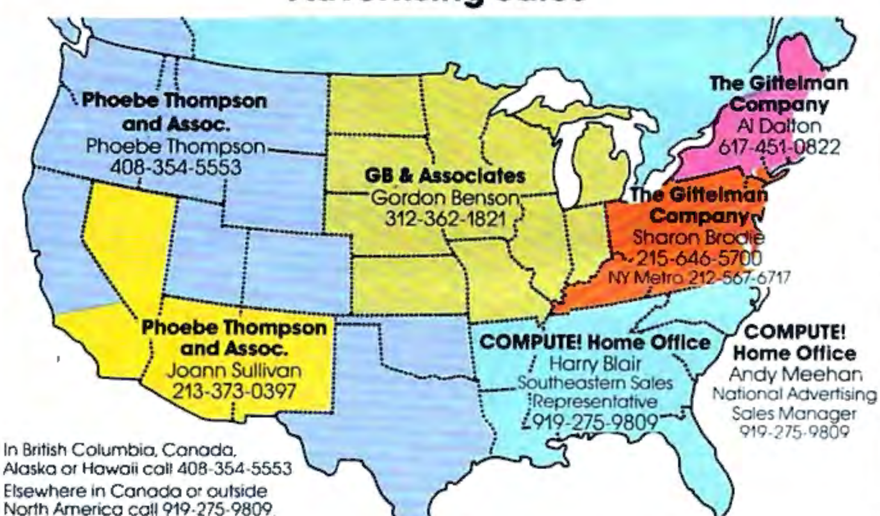
COMPUTE! Circulation Dept.
P.O. Box 5406
Greensboro, NC 27403

TOLL FREE
Subscription Order Line
800-334-0868
In NC 919-275-9809

COMPUTE! Subscription Rates (12 Issue Year):

US	(one yr.) $20 (two yrs.) $36 (three yrs.) $54
Canada and Foreign Surface Mail	$25
Air	
Europe, US Possessions	$38
Middle East, Central America and North Africa	$48
South America, South Africa, Far East and Australia	$88

Advertising Sales



In British Columbia, Canada, Alaska or Hawaii call 408-354-5553
Elsewhere in Canada or outside North America call 919-275-9809

Phoebe Thompson and Associates
101 Church Street
Suite 13
Los Gatos, CA 95030
PHOEBE THOMPSON

Phoebe Thompson and Associates
2556 Via Tejon
Palos Verdes Estates, CA 90274
JOANN SULLIVAN

GB & Associates
P.O. Box 335
Libertyville, IL 60048
GORDON BENSON

COMPUTE! Home Office
625 Fulton Street
Greensboro, NC 27403
HARRY BLAIR
Southeastern Sales Representative

The Gittelman Company
Statler Office Building
Suite 518
20 Providence Street
Boston, MA 02116
AL DALTON

The Gittelman Company
Summit Office Center
7266 Summit Avenue
Fort Washington, PA 19034
SHARON BRODIE

Address all advertising materials to:
Alice S. Wolfe
Advertising Production Coordinator
COMPUTE! Magazine
625 Fulton Street
Greensboro, NC 27403 USA

ASK THE READERS

The Editors and Readers of COMPUTE!

High Vs. Low Resolution

Could you explain the difference between hi-res and low-res graphics?

Mike Porter

The essential difference is that when a computer does not have the high resolution option, you are limited to a set of built-in graphics characters. These characters, like the letters of the alphabet, will be crisp and clear, but you cannot create your own special characters.

A great deal can be accomplished, however, by combining the different symbols of a built-in set. Cubes, stairs, and many other pictures can be created. It's like having perhaps 128 different shapes of paper. You can put them together in thousands of ways, but you can't customize them individually by cutting them with scissors.

High resolution, on the other hand, permits you to control the individual pixels (dots) anywhere on the TV screen. This means that you can create detailed figures of your own design – perhaps the Greek alphabet or an image of a starship – and that curved lines will look more like true curves. High resolution generally adds to the price of a computer, but does provide more flexible graphics, more visual possibilities. Alternatively, it is usually possible to add an optional high resolution capability to computers which do not offer it as a standard feature.

Typing Programs From COMPUTE!

I have seen several programs in **COMPUTE!** that have a "^" symbol in them and there is no such character on my keyboard. Did I miss it in the instruction manual (I triple-checked)? This upside-down V has stumped me. Also, what is the "–" symbol for? I know that an underline means to type the shifted version of whatever character is underlined, but what do you do when "–" stands alone?

Jim Lockridge

The "^" symbol indicates an "up-arrow" symbol (↑) on Commodore computers and represents "to the power of" something. Whenever you see it, type the key with the arrow pointing upwards with respect to the keyboard. Hold down the SHIFT key and hit the SPACE BAR when you see an underlined blank. See the "COMPUTE!'s Listing Conventions" page in each issue.

Atari 400: Can You Add Memory?

I am getting a computer in a few months, but I have a problem. Can the Atari 400 be expanded to hold more memory than it comes with? I constantly see ads for RAM expansion boards, but the ads for the 400 computer itself say "16K RAM (non-expandable)." Well, which is it?

Scott Bonder

Officially, the Atari 400 cannot be internally expanded. However, you can replace the 16K memory board inside your computer with a 32 or 48K board. Atari does not manufacture them, but several third party vendors do. There is even a 64K RAM board sold. Be aware, however, that opening your Atari 400 to replace the board might void your warranty.

How Can Chips Address 128K?

I have two questions, one of which has been bothering me for some time. How can an 8-bit chip such as the new MOS 6509 and 6510 address more than 64K? Commodore's new P128 computer will have 128K RAM. How is this possible? Although the Commodore 64 has 64K RAM, you have stated that only 38K (52K for M.L.) is available for programming. Does this mean that software that had been embedded in ROM in previous CBM machines must now be soft-loaded upon power-up, or is it contained in a plug-in ROM cartridge?

My second question is if programs written for the Commodore 64 will run on the P128, and vice versa. I am particularly interested in the compatibility of the various plug-in cartridges that will become available.

Ron Dagostino

The 6510 chip will not address more than 64K of memory. The 6509 can, however, through a technique known as "bank switching," whereby large blocks of memory (banks) *may* have the same addresses. *The microprocessor must then select which to use from among the banks. Details on exactly how the 6509 makes this selection are not yet available.*

In any case, the 64K limit on memory addressing is not because the 6500 family of chips are eight-bit processors. Rather, it is a result of the chips having 16 address lines ($2^{16} = 65536$). For example, the 8086

LAST NIGHT WE EXCHANGED LETTERS WITH MOM, THEN HAD A PARTY FOR ELEVEN PEOPLE IN NINE DIFFERENT STATES AND ONLY HAD TO WASH ONE GLASS...

That's CompuServe, The Personal Communications Network For Every Computer Owner

And it doesn't matter what kind of computer you own. You'll use CompuServe's Electronic Mail system (we call it Email™) to compose, edit and send letters to friends or business associates. The system delivers any number of messages to other users anywhere in North America.

CompuServe's multi-channel CB simulator brings distant friends together and gets new friendships started. You can even use a scrambler if you have a secret you don't want to share. Special interest groups meet regularly to trade information on hardware, software and hobbies from photography to cooking and you can sell, swap and post personal notices on the bulletin board.

There's all this and much more on the CompuServe Information Service. All you need is a computer, a modem, and CompuServe. CompuServe connects with almost any type or brand of personal computer or terminal and many communicating word processors. To receive an illustrated guide to CompuServe and learn how you can subscribe, contact or call:

CompuServe

Information Service Division, P.O. Box 20212
5000 Arlington Centre Blvd., Columbus, OH 43220

800-848-8990
In Ohio call 614-457-8650

An H&R Block Company

Inside Apple

Apple Computer Inc., 20525 Mariani Avenue, Cupertino, California 95014 Vol. 1 No. 2

For the authorized Apple dealer nearest you, call 800-538-9696 (800-662-9238 in California.)

Fruitful Connections.

There are more people in more places making more accessories and peripherals for Apples than for any other personal computer in the world.

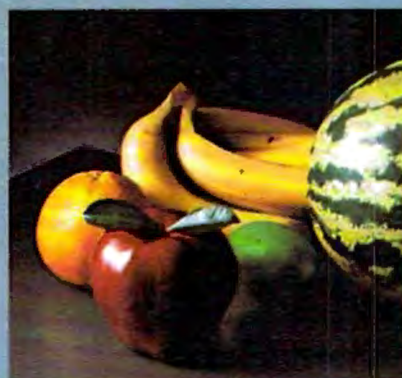
Thanks to those people — in hundreds of independent companies — you can make the humblest 1978 Apple II turn tricks that are still on IBM's Wish List for 1984.

But now we're coming out with our very own line of peripherals and accessories for Apple® Personal Computers.

For two very good reasons.

First, compatibility. We've created a totally kluge-free family of products designed to take full advantage of all the advantages built into every Apple.

Second, service and support.



Now the same kindly dealer who keeps your Apple PC in the pink can do the same competent job for your Apple hard-disk and your Apple daisywheel printer.

So if you're looking to expand the capabilities of your Apple II or III, remember:

Now you can add Apples to Apples.

A joy to behold.

The new Apple Joystick II is the ultimate hand control device for the Apple II.

Why is it such a joy to use?

With two firing buttons, it's the first ambidextrous joystick — just as comfortable for lefties as righties.

Of course, it gives you 360° cursor control (not just 8-way like some game-oriented devices) and full X/Y coordinate control.

And the Joystick II contains high-quality components and switches tested to over 1,000,000 life cycles.

Which makes it a thing of beauty. And a joystick forever.

Gutenberg would be proud.

Old Faithful Silentype® has now been joined by New Faithfuls, the Apple Dot Matrix Printer and the Apple Letter Quality Printer.

So now, whatever your budget and your needs, you can hook your Apple to a printer that's specifically designed to take advantage of all the features built into your Apple. With no compromises.

The 7x9 Apple Dot Matrix Printer is redefining "correspondence quality" with exceptional legibility. With 144x160 dots per square inch, it can also create high resolution graphics.

The Apple Letter Quality Printer, which gets the words out about 33% faster than other daisywheel printers in its price range, also offers graphics capabilities. See your authorized Apple dealer for more information and demonstrations. Because, unfortunately, all the news fit to print simply doesn't fit.

Up the creek without a paddle?

Or lost in space? Or down in the dungeons?

Whatever your games, you'll be happy to know that someone has finally come out with game paddles built to hold up under blistering fire. Without giving you blisters.

Apple Hand Controller II game paddles were designed with one recent discovery in mind:

People playing games get excited and can squeeze very, very hard.

So we made the cases extra rugged. We used switches tested to 3,000,000 life cycles. We shaped them for holding hands and placed the firing button on the right rear side for maximum comfort.

So you'll never miss a shot.

A storehouse of knowledge.

If you work with so much data or so many programs that you find yourself shuffling diskettes constantly, you should take a look at Apple's ProFile™, the personal mass storage system for the Apple III Personal Computer.

This Winchester-based 5-megabyte hard disk can handle as much data as 35 floppies. Even more important for some, it can access that data about 10-times faster than a standard floppy drive.

So now your Apple III can handle jobs once reserved for computers costing thousands more.

As for quality and reliability, you need only store one word of wisdom:

Apple.

Launching pad for numeric data.

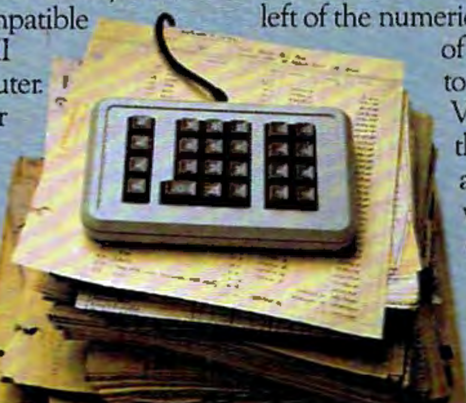
Good tidings for crunchers of numerous numbers:

Apple now offers a numeric keypad that's electronically and aesthetically compatible with the Apple II Personal Computer. So you can enter numeric data faster than ever before.

The Apple Numeric Keypad II has a standard calculator-style layout. Appropriate, because unlike some other keypads, it can actually function as a calculator.

The four function keys to the left of the numeric pad should be of special interest to people who use VisiCalc.® Because they let you zip around your work sheet more easily than ever, adding and deleting entries.

With one hand tied behind your back.



VisiCalc is a registered trademark of VisiCorp. Inc.

microprocessor used in the IBM Personal Computer provides a 20-bit address which can directly address one megabyte of memory ($2^{20} = 1048576$).

The Commodore 64 has 64K of memory. BASIC ROM takes up part of this space, and quite a bit of RAM is used by the computer for pointers, screen memory, sprites, etc. This is why only 38K is normally available to the user. (See **COMPUTE!**, *October 1982, for a 64 memory map.) You can bank-switch the 64 by POKEing address 1. POKE 1,6, for example, makes BASIC go away and the RAM "behind" it is available. The bottom three bits of address 1 are memory control bits and direct the computer's attention to the available alternatives. Of course, what takes control of the machine when BASIC is gone is up to you.*

Look for an article on controlling the 64's "hidden" RAM in an upcoming **COMPUTE!**.

Commodore is not yet sure whether the P128 will be software compatible with the 64.

Can Atari Make Tapes For Other Computers?

It is my understanding that the programmable tone generators in Atari also generate the baud rate and tones used for cassette recording. If this is so, it should be possible to save programs in Kansas City Standard or other two tone formats, making possible the use of Atari editing features to prepare cassette tapes for other computers. How can this be actually done?

Steven S. Coles

While the Kansas City standard cassette interface is in use by many computers, it should be noted that it is far from a full standard. Most computers deviate from the original standard one way or another. One nearly universal deviation is to operate at 1200 baud rather than the original 300 baud spec.

The Kansas City standard uses a frequency of 1200 Hz to indicate a "0" bit and a frequency of 2400 Hz to indicate a "1" bit. The frequency change is performed when the waveform crosses the zero voltage level. The actual data transfer is usually done through as USART, but it is quite possible to simulate this in software. At this point we run into a brick wall: just about everybody saves the data on the cassette in their own way, so you will have to find out what method is used for the computer you wish to adapt to. If at this point you are still interested in giving it a go, I suggest you go down to your local library and drag out the April 1977 issue of BYTE. On page 40 you will find an excellent article by Carl Helmers on how to do it.

VIC Custom Characters

I recently added 8K to my VIC-20. This causes new locations in RAM to be assigned to the start of BASIC, the screen area, and color control area.

There is a technique for using custom characters on the 5K VIC-20 that involves moving down the end of the BASIC RAM working area from page 30 to page 28 and loading pages 29 and 30 with the custom characters. Every custom character program that I've seen uses this technique – POKE 56,28:CLR.

Both the screen RAM and custom character area must be located below page 30 in RAM to work. Unfortunately, with the 8K expansion (and without 3K) there is no space available below page 30. The obvious thing to do is to relocate the start of BASIC text from page 18 to page 30, since with 8K the top of RAM is page 64. This appears possible by changing locations 44, 46, 48, and 50 from 18 to 30. Unfortunately, it doesn't work. The RUN command results in a Syntax Error and GO TO yields Syntax Error in statement 0. (LISTS do work though.)

This wordy prelude leads to my question: Is it possible to have custom characters (and alternate screens) with the 8K expansion (and without the 3K)? If yes, how?

Dick Gough

Several readers have inquired about this. Unfortunately, there is no known solution at this time. Several programmers are currently working on this problem and we'll publish the answer as soon as it is solved.

A Time-saving Tip

I'd like to share a trick I learned from the programmers who did our business software.

Make the first line in your program a REMark statement containing your SAVE command. For example:

```
10 REM SAVE 6, "INVENTORY",D80
```

or

```
1 REM SAVE "INVENTORY",1.1
```

Then, whenever you make a change you don't have to remember where and how to save your program. Just LIST the first line, blank out everything before the SAVE command, and execute. It works equally well with tape or diskette.

Linda Johnson

Atari's Right Cartridge

Aside from "Monkey Wrench" [a programmer's aid package from Eastern House Software], does anyone know of any cartridge that goes into the right slot on the Atari 800? Does Atari have any

We've Got More Than A Fond Attachment For Your ATARI

We've Got A Disk Drive For $488.

Percom Data Corporation believes your Atari* home computer is more than just fun and games. We believe you should be able to get a single-density, floppy-disk-system for your Atari 400 or 800 at a price that will take you into the future without knocking you into the next galaxy.

Percom Data has been manufacturing disk-drive systems, and other accessories for personal computers since the mid-1970's and is the industry standard to follow when it comes to data separation and system compatibility.

The Percom Data AT-88 combines Percom Data quality and reliability at a price that is not a budget-buster.

The Percom Data AT-88 offers 88 Kbytes (formatted) in single-density, with plug-in ease of attachment to your Atari. The AT-88 has integral power supply, "no-patch" to Atari DOS and critical constant speed regulation.

Take advantage of this low introductory price of $488 by calling Percom Data now to get more information, or the name of an authorized dealer nearby. Call toll-free

1-800-527-1222

PERCOM DATA

CORPORATION

Expanding Your Peripheral Vision

DRIVES • NETWORKS • SOFTWARE

11220 Pagemill Road Dallas, Texas 75243 (214) 340-7081
1-800-527-1222

*Atari 400/800 is a trademark of ATARI, INC.

plans to ever utilize this slot, or was it an idea that didn't pan out?

Bill Lukeroth

The problem with the right slot is that the Atari 400 doesn't have one. Atari has evidently abandoned applications for the right slot since such an application could not be used on the 400. Atari is committed to supporting both machines equally. Also, because a full 16K can be put on the left cartridge, it alone is enough space for most applications.

Commodore Time

I have a VIC-20. On several occasions I have attempted to tally the amount of computer time utilized via the TI and TI$ functions.

Could you explain first the purpose of TI? Can it be modified manually – set to zero? What does a printout of 1429292 mean? What is its relationship to realtime and TI$?

Secondly, is TI$ supported to count realtime? If so, can you explain why 6 hrs. 31 mins. would show up on TI$ as 063639?

Lastly, is there any way to maintain a cumulative tally on the system of aggregate "on-time"?

D. L. Branam

You can tell Commodore computer's special time variable, TI$, what time it is (or set it to zero so it can keep track of how long the system has been on) by treating it like an ordinary string variable. For example:

TI$ = "033000"

would let the computer know that it was 3:30. This can be done either from within a program or by just typing it in from the keyboard in "direct mode." The string is arranged "HHMMSS" for the positions of hours, minutes, and seconds. It will take anything up to 240000 hours and must include all six numbers, even if a zero is in the first position as in the example above.

You can print out the time in a variety of ways. Here's one:

?LEFT$ (TI$,2) "/" MID$ (TI$,3,2) "/" RIGHT$ (TI$,2

or just:

? TI$

TI is the numeric clock variable. It is set to zero when power is first turned on or when you reset the clock: TI$ = "000000". The number in TI is counting time in 1/60ths of a second. To see TI:

10 ?"[HOME]" TI: GOTO 10

Six hours 31 minutes should be 063100 when TI$ is printed out.

Atari Memory Expansion Problems?

I own an Atari 800 with 16K memory and am careful about the quality of products I buy for my computer. There seems to be a rumor going around about the 32K RAM memory board made to fit the 800 model. I've heard that adding this board can cause errors to occur in the computer's performance. Please tell me if it's true and, if so, how or why it happens.

Allen Levy

We have heard of no problems related to the use of a 32K board on an Atari 800. These expansion boards are not manufactured by Atari, but rather by third party vendors. The requirements of an Atari board are fairly specific: they must not use too much power and they have to be fast enough (200 ns. or better). We haven't heard, though, of problems relating to expanding memory with these products.

INPUT That Puts Anything In

Here's an interesting Commodore input routine that I'd like to share with you. Ever notice that when you INPUT a string which has a comma or colon that the computer only takes in the part before the punctuation and then prints EXTRA IGNORED?

Here's how to get around it. Say you want to INPUT C$:

```
10 GOSUB1000:C$=B$
20 PRINTB$:END
1000 B$=""
1010 GETA$:IFA$=""THEN1010
1020 PRINTA$;
1030 IFA$=CHR$(13)THEN RETURN: REM 13 MEANS THE
     RETURN KEY WAS TYPED
1040 B$=B$+A$
1050 GOTO 1010
```

When you run this, you don't get the normal question mark. Now you can put in anything you want, but don't use A$ or B$ any place except in this subroutine. Whenever you want to put in a string, GOSUB to 1000 and, when you come back with RETURN, just let the string you're looking for (C$ in this example) be equal to B$.

George Trepal

This INPUT routine is great for people who will want to use a computer program, but don't know about avoiding commas, etc. Whatever they type, the program will take it in without stopping and going to an error message and then saying READY. It's also possible to use delete and insert to correct errors (but these "characters" will be included in the final string).

To prove it, run this and type TEST the first time. Then, when the program ends, type: ?LEN(B$) to see how long B$ is. You will get four as the answer. Now run it again and type TESX and then use the delete key to change the X back to a T. ?LEN(B$) will now give you a six because B$ still contains the X and a character for a delete. You don't notice these extra characters, though, because when B$ is PRINTed, it puts the X on the screen and then deletes X, replacing it with T. It's too quick to see.

HOW TO MAKE YOUR VIC-20

AWESOME

Right out of the box, your VIC-20® from Commodore is one great little computer. And it gets even better with DATA 20's easy to buy, easy to install, and easy to use enhancements. Here are four ways you can get new power and more sophisticated capabilities from your VIC...

VIDEO PAK gives you a computer that outperforms systems costing twice as much. Plug our cartridge into your expansion port, and your display instantly goes to the industry-standard 24 lines, with a choice of 40 or 80 upper and lower-case characters. At the same time, you up your memory from 5K to 20K or 70K, so you can handle more sophisticated functions—including most 8032 software. Our package also includes a terminal emulator and a screen print feature. VIDEO PAK is a must for word processing—and your key to increased performance on everything from games to spread sheets. Suggested retail is just $299.95 including 16K—or $399.95 for 64K.

PRINTER INTERFACE is simple, yet sophisticated. Flexibility, continuous visual monitoring of the data transfer functions, and easy installation make this an exceptionally smart buy! A glance at the status lights, and you know if the printer is hooked up, if the data buffer is full, and if data is being transmitted. Easily configured DIP switches match your VIC-20 to most popular printers. The DATA 20 interface comes with cable and connector, needs no assembly, and virtually troubleshoots installation for you. Suggested retail: $69.95

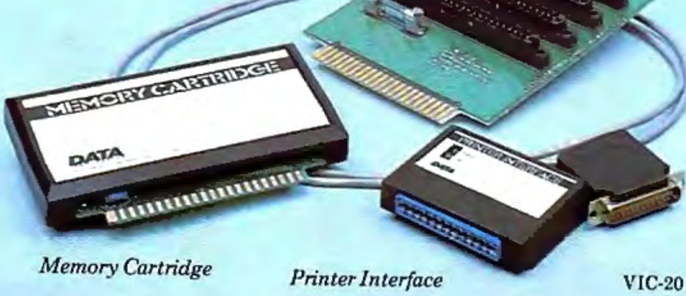
EXPANSION CHASSIS lets you use 4 cartridges at once. Run a series of compatible memory, software, or game cartridges of any make. Just pop in any cartridge with the standard 22-pin edge connector. And don't worry about your VIC-20 power supply—our chassis protects it with a built-in 500ma fuse. Suggested retail: $64.95

MEMORY CARTRIDGE boosts your brainpower to 20K. Here's an ideal first add-on for your VIC-20. And when DATA 20 gives you more memory, you can forget about headaches. Our cartridge is housed in a rugged plastic case and features 200ns RAM's for reliability. Suggested retail: $99.95

Check out our AWESOME peripherals. Ask your computer dealer for a first-hand look at our extensive capabilities, high quality, and very reasonable prices. Or send $3.50 for a copy of our current catalog on VIC-20 compatibles.

DATA 20 CORPORATION
23011 Moulton Parkway, Suite B10, Laguna Hills, California 92653.

Expansion Chassis



Memory Cartridge *Printer Interface*

DATA 20 CORPORATION

Price/Performance Peripherals

VIC-20 is a registered trademark of Commodore Computer Systems.

Questions Beginners Ask

Tom R. Halfhill, Features Editor

Are you thinking about buying a computer for the first time, but don't know anything about computers? Or maybe you just purchased a computer and are still baffled by what personal computing is all about. Starting this month, **COMPUTE!** *will tackle some questions which we are most frequently asked by beginners.*

Q: Which is the best computer to buy?

A: The best one for your needs.

Seriously, we're not trying to duck the question. People ask us this all the time, in letters, telephone calls, and at computer shows we attend. We get the feeling they are never really satisfied with our answers, since what they really want to hear is something like, "Buy the Atapple ZX-20, it's definitely the best one." Unfortunately, we cannot give such an answer. For one thing, since **COMPUTE!** covers many machines, the magazine must maintain objectivity. But more importantly, there is no one right answer. All the computers have their own strengths and weaknesses, and all computer buyers have – or should have – their own ideas of what they need in a computer. We think nearly anybody who buys one of the major brands with a clear idea of his or her needs will be satisified with the purchase.

The key is to identify your needs and desires. If game-playing will be a major use of your computer, then color graphics and sound will be important features. Someone primarily interested in word processing may well have no need for either feature.

If you've looked hard and long at the various computers in a certain price range and still can't decide between them, then perhaps the differences are too slight to matter anyway. Or maybe you should base your decision not on the hardware, but on the available software. If the computer will be used primarily for educational purposes, and you're attracted by a particular line of educational programs, you may lean toward the computer that those programs are designed to work on. The programs may not be compatible with or available for another machine.

If you still think we are sidestepping the whole question, then consider this: If one brand of computer were clearly superior, and if we at **COMPUTE!** were in a position to know about it, then it stands to reason that all of our editors would own that computer. But in fact, both at work and at home, we own and use many different computers. 'Nuff said?

Q: What are PEEK and POKE?

A: PEEK and POKE are words (instructions to the computer to do something for you) in a computer programming language known as BASIC (Beginner's All-purpose Symbolic Instruction Code). BASIC is the standard language on home/personal computers. PEEK and POKE allow you, as a programmer, to work directly with the computer's memory.

PEEK allows you to examine the contents of a single memory location (known as a "byte"). Each memory location in a computer has a numbered address, sort of like houses in a city. In turn, each memory location *stores* a number which usually has something to do with the operation of the computer or a computer program. If you type PRINT PEEK (8502), the computer will PRINT on the screen the number stored in that address. Therefore, PEEK is often used in programs to determine if a certain number is stored at a particular location, usually as a prelude to changing the number to achieve some desired result.

POKE is the word that allows you to make those changes, to change numbers stored in locations in Random Access Memory (RAM) – that part of a computer's "user memory" which can be changed by the programmer. For example, if you type POKE 82,0, the number 0 will be stored at memory location 82. POKEs can often change some facet of the computer's behavior. Since each model's memory is arranged differently, PEEKs and POKEs will not achieve the same results on different computers (in the above example, POKE 82,0 will make the left screen margin zero on an Atari).

Q: What is a CONTROL key?

A: A CONTROL key (often abbreviated CTRL) is

a special key found on many computer keyboards. In effect, it works something like a SHIFT key. Just as a SHIFT key adds a function to a regular key – i.e., changes a lowercase letter to uppercase, or changes the "4" key to a dollar sign – the CONTROL key also is used in combination with another key to select an additional function or symbol.

These functions and symbols vary among different models of computers. For example, holding down the CONTROL and "C" keys on an Apple II will usually stop (or "break") a BASIC program which is running. CONTROL-C on an Atari will print on the screen a small graphics character resembling the lower right corner of a box. On a Commodore VIC-20, CONTROL-C has no effect; another special key is used instead to print graphics symbols. The manual which comes with every computer explains the functions of its special keys. ©

Computers And Society

David D. Thornburg, Associate Editor

The Computer As A Tool For Discovery

The notion that the development of low-cost computers is "revolutionary" is not a new one, but the word revolutionary is used so much that one is likely to dismiss it as pure advertising hype along with words like "new" and "improved." And yet those of us who have been involved with this industry since its inception are aware that the development of the personal computer is not, by itself, revolutionary just because it may bring computer technology into people's homes.

"Revolutionary" is a special word – it implies that a technology or tool causes far-reaching changes in many aspects of our lives.

The development of the steam engine was revolutionary; the development of steam cleaning for carpets was not. The development of the telephone was revolutionary; the development of the answering machine was not. The development of the airplane was revolutionary; the development of in-flight entertainment was not.

Computer technology has had an impact that reaches far beyond the world of the computer itself. Computer users in industry and academia have known this for many years. Now that the power of the computer has reached the home, can we expect that people will start thinking about their world differently?

I think so.

The computer will help people to explore ideas that they wouldn't begin to explore if the computer hadn't given them the leverage to start thinking about them.

Beauty And Practicality

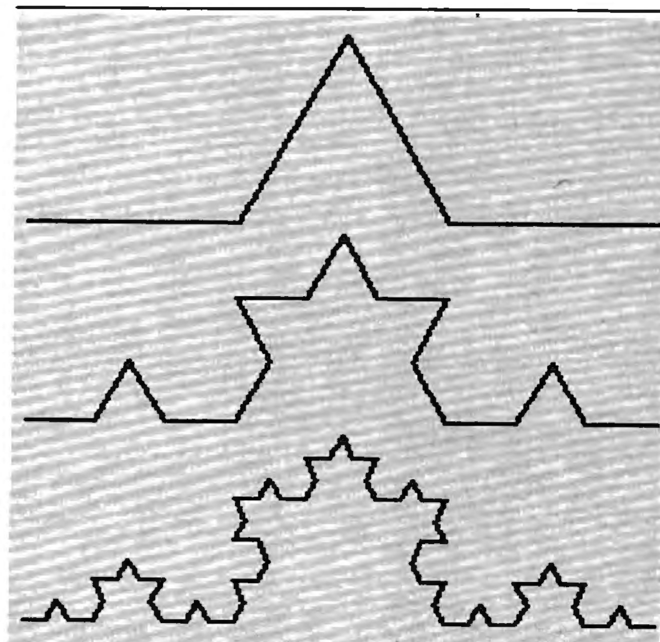
As an example of this, let's explore the development of a new field of mathematics called "fractal geometry." I have touched on this branch of mathematics in the "Friends of the Turtle" column a few times. I am intrigued by it because it deals with topics of considerable beauty and practical interest. Its seeds were planted a hundred years ago, but it was only after the development of the computer that anyone was able to begin to advance this field beyond the crudest level.

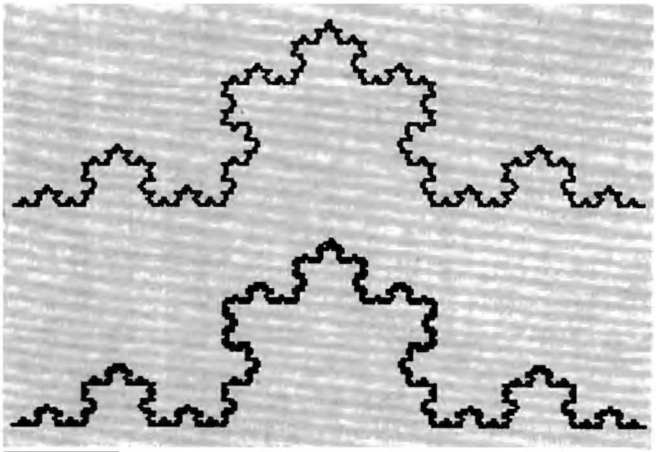
I realize the risk of illustrating a computer application based on mathemetics, since it tends to reinforce the erroneous concept that computers are primarily mathematicians' tools. The only reason for pursuing this example is because it is an interesting story in its own right.

In the late 1800s mathematicians were exploring some questions that went to the very foundations of geometry. One question of interest was if one could construct a curve that would fill a plane. At first thought, the idea of filling a two-dimensional surface with a curve made from a one-dimensional line is as absurd as asking for a roll of optically flat steel, or asking how many angels can dance on the head of a pin.

To the Italian mathematician Guiseppe Peano, this was a most intriguing question. In 1890 he published a proof that space-filling curves were, in fact, possible – that one could construct a curve that has the dimension of a surface. While this proof attracted the attention of several other mathematicians, the bulk of the academic community abhorred the thought of such "ill-behaved" curves.

In 1904 Helge von Koch continued the pursuit of strange types of functions by publishing the discovery of the "snowflake" curve. This curve is created by preparing successive generations from a simple motif. The rule to be followed is that each new generation is made by replacing each straight line in the previous generation with a copy of the motif itself.





If this process is carried on to infinity, one gets a very strange curve indeed. First, the curve is everywhere bumpy – there are no smooth regions. Second, even though the curve has clearly defined boundaries, it has infinite length. Third, the curve has a "dimension" that is intermediate between that of a line and a surface. To mathematicians of the early twentieth century, this curve was monstrous. To the contemporary mathematician Benoit Mandelbrot, it represented the need for a new field of mathematics, to be called fractal geometry.

The history and development of this field is beautifully illustrated in Mandelbrot's new book, *The Fractal Geometry of Nature* (W. H. Freeman, San Francisco). Through the pages of this richly illustrated volume, the reader is treated to a new way of thinking about geometry and nature.

For example, if you want to model a coastline, you are far better off to use a fractal curve than a smooth approximation, simply because coastlines are not smooth. Coastal lengths depend on the ability of the measuring stick to follow the nooks and crannies along the way. A coarse measuring stick gives a result corresponding to an early generation of a fractal curve. As the length of the measuring stick gets smaller, the total measured length of a coastline grows ever larger. This is also true for fractal curves.

Where does the computer fit in all of this? The notion of defining a curve in terms of itself may challenge the imagination, but it has a simple implementation in computer programming called recursion. Furthermore, the speed and accuracy with which computer-driven plotters can graph the various stages of curves free the mathematician to study their properties without being bogged down in drafting.

Computer graphics plays another pivotal role in the practical application of fractal geometry as well, since it is the tool that allows the creation of the simulated landscapes seen in movies such as *Star Trek II*. This practical application of a branch of mathematics would not have been possible were it not for the computer.

Those of you who read "Friends of the Turtle" know that fractal curves can be created on home computer systems using turtle graphics. Their expression in languages such as Logo is quite simple, and Mandelbrot's book provides hundreds of challenges for the interested programmer.

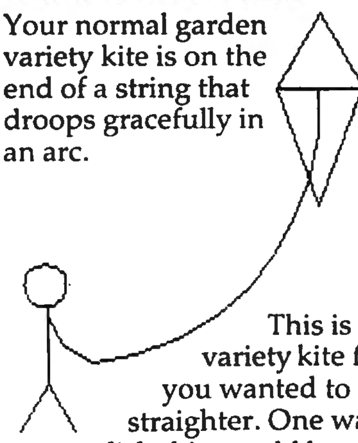
It is important to keep the role of the computer in perspective. The reason that these curves were not explored in depth in the early 1900s is that there was no appropriate tool to aid in their exploration. Now that the computer has made the study of fractals accessible to millions of people, one can expect the field to advance rapidly.

I Call It Kring

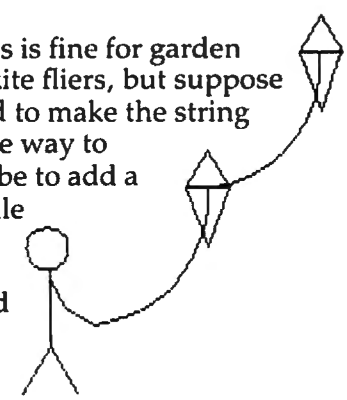
I saw a T-shirt that carried the message: "Recursion is a way of expressing the infinite in finite guise." My friend Sam Savage (the computer scientist/mathematician that invented the jigsaw puzzle called "Shmuzzles") likes to play with the infinite recursively. While I have used Logo to tinker with the latest of his ideas, you may wish to implement them mechanically.

Consider The Kite

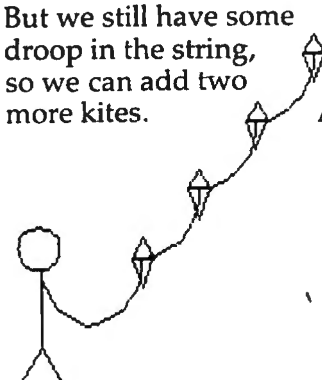
Your normal garden variety kite is on the end of a string that droops gracefully in an arc.



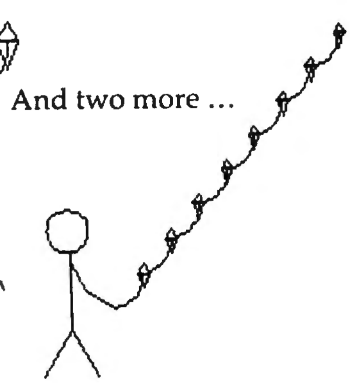
This is fine for garden variety kite fliers, but suppose you wanted to make the string straighter. One way to accomplish this would be to add a second kite in the middle of the string. Because each kite would carry less weight, they would each be smaller.

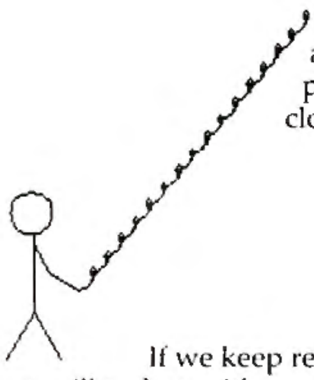


But we still have some droop in the string, so we can add two more kites.



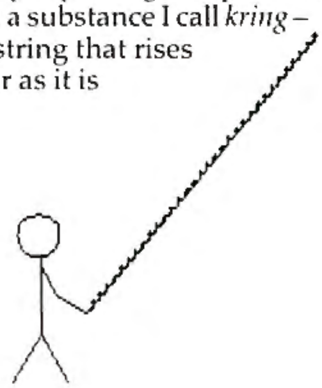
And two more ...





And so on, reducing the area of each kite and placing them closer and closer together.

If we keep repeating this process, we will end up with a substance I call *kring* – a combination kite/string that rises straight up in the air as it is unfurled.

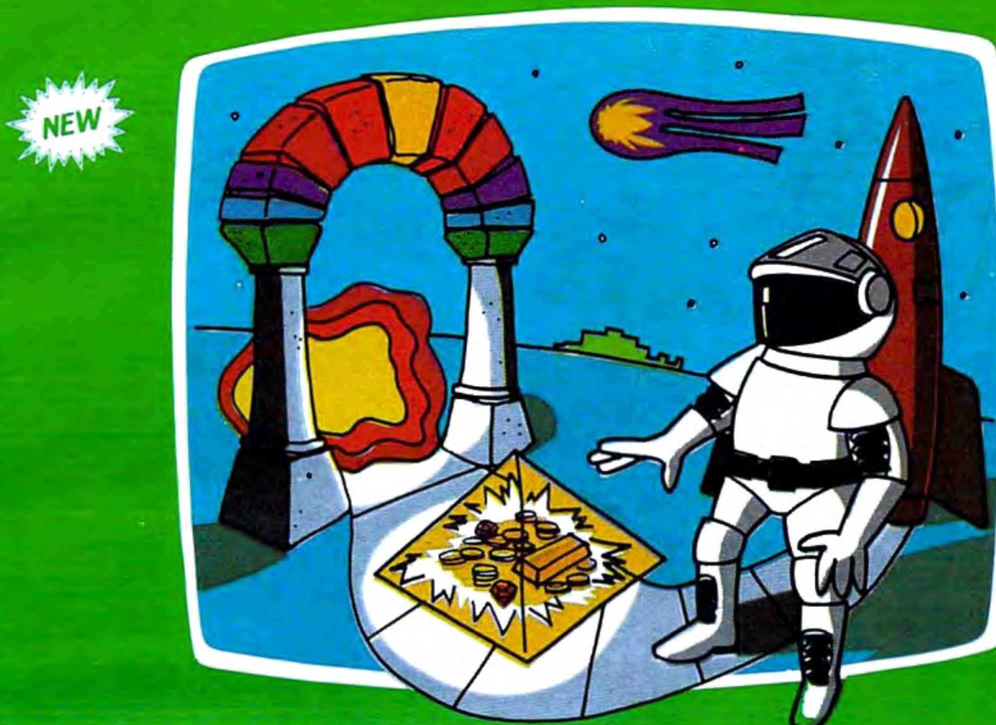


Now that's revolutionary!

©

JOURNEY TO THE PLANETS™

A space, adventure, and arcade action game for your ATARI® 400/800™ personal computer.



What is your favorite type of game; space, arcade, or adventure? "Journey To The Planets" presents an intriguing combination of all three as you find yourself on a strange planet in a strange universe. Luckily, the local gods are friendly and supply you with energy, a spaceship, and weaponry. In turn, you agree to search the universe for treasures for the gods. Board your ship, take off, accelerate through the upper atmosphere and out into space. Your flight should take you past many other inviting planets. With a slow approach and skilled maneuvering, you drop down through the planet's sky to a soft landing on its surface. Disembark and wander through several TV screens full of mystery and excitement. A different adventure awaits you on each planet.

Planetary adventures are designed to exercise your puzzle solving intellect, with arcade action thrown in to enhance the excitement. Although you are given as much time as necessary to solve each adventure, your score, which is based on many factors, favors those who are speedy.

After you capture all the treasures this universe has to offer, return to your adopted planet. Who knows, the gods may be so happy that you can convince them to send you back to your real home. You can't get there without their help!

Available from your local Atari retailer or send $29.95 in check or money order (California residents add 6½% sales tax) to JV Software Inc.

A 32K assembly language program written for your ATARI® 400/800™ computer.

Other products by JV Software include Action Quest and Ghost Encounters, both 16K real time adventure games. Available on cassette or diskette for $29.95.

JV SOFTWARE, INC

3090 MARK AVE. SANTA CLARA, CA 95051

Sound Synthesis

Tom R. Halfhill, Features Editor

Synthesized computer music is a recent development, but inventors have been working on "synthesizers" for decades. Today's home computers and microchips are now starting to open a new world of music and sound for everyone.

Hal Chamberlin, a leading authority on computerized music, remembers the days when adventuresome programmers used transistor radios and even line printers to squeeze music from their early computers.

"People used to tune a little AM radio to an open frequency and hold it next to their spacebars and listen to the sound of [program] loops," recalls Chamberlin, vice president of engineering for Micro Technology Unlimited in Raleigh, North Carolina.

The method worked because pulses flowing through the computer's logic circuits would emit radio frequencies which "leaked" from the computer into the radio's receiver. The programmers on these early IBMs – fiddling around when the boss wasn't looking – soon learned they could play different notes and tones by writing little machine language programs with carefully timed loops.

"They even used to make music by 'playing' the printer," says Chamberlin. "They found out they could control the little hammers in the printhead with a machine language program. So they wrote programs to fire the printhead hammers in a certain pattern to create rhythms.

"Of course," he adds, "it wasn't so great on the printheads."

Such experiments seem crude, even quaint, in this day of computerized music synthesis and home computers with built-in, multiple-voice sound synthesizers on a chip. But these early efforts illustrate that today's "modern" sound devices are really the result of years of research, inventing, and just plain fooling around.

In fact, people have been working on sound synthesizers since the 19th century. And although today's computerized synthesizers seem incredibly advanced in comparison, the leading experimenters in the field believe electronic music is only starting to make itself heard.

Telharmoniums, Theremins, And Rhythmicons

The first music "synthesizer" was built between 1896 and 1906 by American inventor Thaddeus Cahill. He called it a "Telharmonium." The Telharmonium is to modern synthesizers what ENIAC is to modern computers. The Telharmonium weighed more than 200 tons, and moving it to New York from Cahill's lab required several railroad flatcars.

Since the Telharmonium was a pre-electronic instrument, it functioned by means of electric drive motors, pulleys, belts, and gears. Yet it was similar in basic concept to today's synthesizers. It was *polyphonic* (as opposed to *monophonic*), meaning it could play more than one note at a time and thus create chords. It was equipped with a standard music keyboard, but the controls were so complicated that it took two people to play the thing.

The loudspeakers worked mechanically, and the machinery required to generate enough current to drive the speakers was so noisy that part of the Telharmonium had to be housed separately from the listening room. Unfortunately, after ten years of Cahill's work, the Telharmonium was a commercial failure.

For one thing, it was obsolete soon after it was finished. The diode tube was invented in 1904, followed by the triode tube in 1915, which made electronic amplifiers possible. It wasn't long before tube-powered electronic instruments began appearing.

The most successful of these was an instrument invented between 1920 and 1924 by Leon Theremin, originally called an "Etherophone" or "Thereminovox" but now known simply as a "Theremin." This odd instrument was played without being touched – the musician passed his or her hands through the air near two antennas which controlled the pitch and volume. To say the least, this made a Theremin very hard to play, since there were no pre-defined notes like the keys on a piano or the frets on a guitar. Still, Theremins became popular in the late 1920s.

Leon Theremin invented another electronic instrument in 1931 – the "Rhythmicon," the first electronic rhythm instrument. The Rhythmicon was quite sophisticated with features which have appeared on rhythm synthesizers only recently.

A SPEECH SYNTHESIZER ON A DISK!

That's why Apples and Ataris are saying: "Talk Is Cheap"

IT'S CALLED THE SOFTWARE AUTOMATIC MOUTH, S.A.M. FOR SHORT

It's a high quality speech synthesizer created entirely in software. You use it as a software utility, load it into RAM, and then use your machine as usual, except now you can make your programs talk. It generates the speech sounds on demand, so there is no limit to what it can say.

When you hear S.A.M., you'll probably agree that it sounds better than all the hardware speech synthesizers for Apple or Atari computers. And, it has a truly remarkable price.

YOU CONTROL INFLECTION, PITCH AND SPEED

With its user-variable inflection, S.A.M. can accent words on the right syllable and emphasize the important words in a sentence.

You can also make S.A.M.'s speech higher or lower, and faster or slower, over a wide range of settings.

USE EASY PHONETIC INPUT OR PLAIN ENGLISH TEXT

S.A.M. understands a simple phonetic spelling system, not a mysterious alpha-numeric code. S.A.M. helps you learn phonetic spelling by showing you your mistakes, and the owner's manual gets you started with an English-to-phonetics dictionary of 1500 words. So it's easy to make S.A.M. produce exactly the sounds you want.

But suppose you want to type ordinary English, or you want your machine to read a word processor file aloud. The S.A.M. disk comes with RECITER, an English text-to-speech conversion program that lets S.A.M. speak from plain English text.

ADD SPEECH TO YOUR PROGRAMS WITH EASE

In a BASIC program, you add speech with just a couple of commands. In a machine language program, it's just as easy. S.A.M. comes with four demonstration programs to show off its distinctive features and help to write your own talking programs. Write adventure games with talking characters, educational programs that explain aloud, or utilities with spoken prompts — put your imagination to work.

S.A.M.

S.A.M. programmed by Mark Barton.

You can order S.A.M. directly from DON'T ASK. Add $2.00 for shipping and handling to your check or money order (or order C.O.D.)

S.A.M. for the Apple II/II+

Includes an 8-bit digital-to-analog converter and audio amplifier on a board. Only **$124.95**

Requires 48K, disk. (S.A.M. takes up to 9K; RECITER 6K.) You will also need a speaker.

S.A.M. for the Atari 400/800

S.A.M. talks through your television speaker. No additional hardware is required. Only **$59.95**

Requires 32K, disk. (S.A.M. takes up 9K; RECITER 6K.) Note: to produce the highest quality speech, S.A.M. automatically blanks the screen during vocal output; the display is preserved. S.A.M. can talk with the screen on, but the speech quality is reduced.

DON'T ASK INC
COMPUTER SOFTWARE

2265 Westwood Boulevard, Suite B-150
Los Angeles, California 90064
Telephone: (213) 397-8811

Hear S.A.M. at your favorite computer store today!
Dealer inquiries welcome.

THE 40/80 VIDE

O CARTRIDGES

Quantum Data, Inc. (QDI) produces two (2) 40/80 Video cartridges, the 40/80 Video Combo Cartridge with 16K basic user memory, and the 40/80 Video Cartridge which does not contain memory.

The 40/80 Video Cartridge and Video Combo Cartridge is the means to upgrade the VIC-20 computer to a 40 x 24 or an 80 x 24 character display which provides a wealth of new uses for the VIC-20. With the appropriate software, you can now accomplish quality word processing and various business functions that previously were very difficult to achieve with only the VIC's standard 22 character video display.

- *Features a high quality 8 x 8 dot matrix*
- *Character-by-character reverse video attributes allowing adjacent characters to have different attributes.*
- *Black & White composite video (6545 controller does not support color). The Black & White composite video output has the same connector as the VIC video output, 5 pin DIN jack.*
- *Includes two character sets: The ANSI standard 7-bit character set and the Commodore character set. The ASCII character set features all of the standard lower-case and upper-case letters, symbols and numbers. The character set is contained in a standard 2716 EPROM. It is used most often to allow the 40/80 column cartridge to display all of the Block Graphics supported by the VIC-20.*
- *Cannot be used with abbreviated commands nor does it support multiple line wrap around.*
- *Operates in VIC-20 Block Graphic mode.*
- *Features lower-case dot descenders.*
- *All features are accessible through BASIC using POKE commands (screen memory resides at $B800 through $BFFF).*
- *Contains 2K of CMOS internal video RAM; no system RAM is used by the Video Cartridge.*
- *Plugs directly into the VIC-20 memory expansion port or a Mini-Mother or Maxi-Mother board.*
- *Is small in size: 6½ x 4½ inches encased.*

The 40 character mode may be easily viewed on most standard T.V. sets but a monitor is required for the 80 column mode to provide the necessary additional resolution.

Video Combo $319.95

Video Cartridge $219.95

QUANTUM DATA, INC.

Suite A, Box 285, 14252 Culver Dr. • Irvine, CA 92714 • 714/966-6553

Visa & MasterCard are accepted. Above prices retail in U.S. dollars. Shipping and handling not included.

The most popular electronic instrument of the past half-century was invented in 1935 by Laurens Hammond – the Hammond organ, still widely used.

But although these devices were electronic *instruments*, music historians trace the origin of electronic *music* back to Paris in 1947-48. Acoustical engineer Pierre Schaeffer and composer Pierre Henry began experimenting with new sounds by using electronic filtering, speed changes on tape recorders, and other manipulation tricks done in studios. Their technique became known as *musique concrète*, and was quickly picked up by tinkerers elsewhere. By 1952, the first concert of electronic music was sponsored by Columbia University at the New York Museum of Modern Art.

The problem with these techniques was that it took many hours of tedious tape splicing and other tricks to produce only a few brief minutes of sound. And musicians couldn't even hear the results until they were done. That's why there was a lot of interest during the late '50s and early '60s in instruments which could produce electronic music directly. Even the old Theremins from the '20s – updated with transistors – were resurrected.

Toward A New Form Of Music

Robert A. Moog – whose name is virtually synonymous with sound synthesis – was selling kits for transistorized Theremins in the early '60s when he was inspired to invent his own electronic instrument. The result was the Moog Synthesizer, first built in the summer of 1964.

Although recognized by electronic musicians as an important development, the Moog Synthesizer was practically unknown to the general public until a few years later, when it was featured on a record album entitled *Switched-On Bach*. The album was a collection of Bach compositions performed entirely on a Moog Synthesizer by musician Walter Carlos. Almost instantly, *Switched-On Bach* catapulted up the charts like a pop record, and became the biggest-selling classical record of all time. It was especially popular with teen-agers, who astounded their parents by playing electronic Bach along with their Beatles and Rolling Stones records.

However, a few classical music devotees, stunned by the album's popularity, dismissed the electronic interpretations as "artificial." Some critics, although they are decreasing in number, argue that music which is synthesized by purely electronic means is somehow artificial or unnatural when compared to conventional instruments.

Today, Moog counters these arguments with: "The fact is, you don't find musical instruments in nature. The only 'natural' musical instrument is a human voice. The fact that a synthesizer produces its music by electronic means doesn't mean it's 'artificial' in any sense. It's no more artificial than taking a bunch of wood and gluing it together into a box and stretching some strings over it to produce sounds."

Electronic musicians, of course, never had any doubt that their instruments deserved equal billing with violins and woodwinds. In fact, years ago they recognized synthesizers as a rare historical opportunity to open a new world in music. Although synthesizers are often used to mimic "conventional" instruments, the most exciting electronic music takes advantage of the synthesizer's power to create totally new sounds. This provides the possibility of entirely new forms of music.

For example, would rock 'n' roll have happened without electric guitars? Did the invention of a musical instrument with a totally fresh sound spur the rise of a new genre of music? For the members of a whole generation, rock has become the dominant musical style. Synthesizers are now used in virtually every form of music, but even Moog isn't sure if they will "liberate" themselves and spark a new form which could replace rock. "Musicians are moving in so many different directions these days that it's hard to say if a new musical form will emerge."

It may be too early yet for the birth of a dominant musical form based on synthesizers, since the instruments themselves are changing so rapidly. Not only are they advancing technologically almost day by day, but the rising use of microchips is just beginning to make them affordable for everyone. To return to the rock 'n' roll analogy, it would have been difficult for the teen-age groups of the '50s and '60s to arise if electric guitars had cost thousands of dollars. Or if radical new advances were constantly rendering three-year-old guitars obsolete.

Synthesizers, on the other hand, are still passing through important phases in their development. Moog foresees a trend away from analog sound synthesis to digital, or at least to digitally controlled analog instruments. "There's so much more you can do with digital sound synthesis, especially in small keyboard instruments like the little Casios or Yamahas you can buy very inexpensively."

About a year ago, Moog set up a new company – Big Briar, Inc. – and relocated to a small town in rural North Carolina to work on such developments. Among his frequently used tools, he says, is an Apple II microcomputer. Mindful of the baffling array of controls on modern synthesizers, he's experimenting with new types of control devices aimed at making synthesizers easier to play. But he warns that the complex instruments will never be a cinch.

"Musical instruments will never be easy to

Make your fun and games computer get serious

For only $79.95

VIC-20® The Wonder Computer of the 1980's. Less than $300. One heckuva lot of fun. But if you're using your VIC-20 strictly for recreation and fun, we at Micro-Systems think you're not getting your money's worth. Because we've designed the Micro-Systems IEEE-488 Cartridge which allows you to interface VIC and CBM 64 to all existing Commodore IEEE peripherals (8050, 4040, 2031, and printers). In addition, we've designed the RS-232 Interface board to allow the VIC-20 and CBM 64 to communicate with various serial devices such as printers, modems, and other systems—**NOT JUST A BUFFER DRIVER.** Also available is Micro-Systems Modem for the VIC-20 and CBM 64 which allows your computer to utilize auto-dial and auto-answer control over the communication process. If you need to access more than one cartridge in your VIC-20 expansion port, we have the solution. The V-Expander is now available with 3 or 6 additional expansion ports. The 6-slot V-Expander is switch selectable.

CIE Cartridge [IEEE-488]	$ 99.95
VIE Cartridge [IEEE-488]	$ 79.95
C-232 RS-232 C64 Interface	$ 45.00
V-232 RS-232 VIC Interface	$ 45.00
V-Modem	$ 129.00
V-Expander [3-slot]	$ 49.95
V-Expander [6-slot]	$ 85.00
VIC-1210 3K Expansion	$ 35.00
V8K RAM Expansion	$ 49.95
V16K RAM Expansion	$ 95.00
V24K RAM Expansion	$ 149.00

Available from Micro-Systems

VIC-20 Color Computer	$ 195.00
VIC-1541 Single Disk Drive	$ 475.00
VIC-1525 Graphic Printer	$345.00
VAC-Audio Cassette Interface	$ 29.95
VMC/CMC-VIC & C64 Monitor Cables	$ 19.95
Joy Stick [Arcade Quality]	$ 29.95

VTE/CTE-Terminal Communicator program for VIC and C64	
Cassette	$ 10.00
Diskette	$ 15.00
VT 40-VIC 40 Col Terminal Communicator Cartridge	$ 65.00
Super Expander	$ 65.00
Programmers Aid Cartridge	$ 55.00
VICMON Machine Language Monitor	$ 55.00
Intro to Basic Programming	$ 24.95

IMMEDIATE DELIVERY

Yes, Please send me:

QTY.	MODEL#	NAME	PRICE
TOTAL	(In Texas, add 5% sales tax)		

Your Name ____________

Address ____________

City ____________ State ____________ Zip ____________

payment method: Check ☐ Card-Exp. Date ____________

Master Card # ____________

Visa # ____________

American Express # ____________

Signature ____________

Micro-Systems • 2554 Southwell • Dallas, Texas 75229 • (214) 484-7836

®Registered Trademark of Commodore Int.

play," he says. "If it's too easy to play, most musicians would say it's not a musical instrument, because you usually can't do much with an instrument that's 'easy to play.' It has too many limitations."

Still, as synthesizers get easier to manage and less expensive, they become more accessible to the average musician – and thereby more widely heard and appreciated by music listeners. Pop music historians may recognize this trend as the same sort of breeding ground for rock created by 45 rpm records in the '50s.

SID: Synthesizer On A Chip

One important way in which synthesizers are becoming accessible to people is within home computers. Virtually every new model introduced in recent years has featured more sophisticated sound capabilities.

Unfortunately, up to now, the sound capabilities have attracted less attention than the often more glamorous feature: graphics.

"Well, in terms of the human senses, sound inherently takes a backseat to sight," notes Frank Covitz, a New Jersey research scientist whose sideline is computer music. "Sight is the more important sense, so computer graphics naturally gets more attention."

For instance, very little has been written on the Atari computer's sound capabilities, although the built-in four-voice sound chip has represented the state-of-the-art in home computers for the last couple of years. Almost all the attention has been focused on the Atari's graphics. Perhaps this will change now that a computer with even more advanced sound has appeared on the market – the Commodore 64 with its SID (Sound Interface Device) chip.

The new SID chip is generating lots of interest among computer music enthusiasts. It may well be a herald of the sound capabilities of tomorrow's home computers. "I think machines of that class [home computers such as the 64] in the future will be expected to have sound chips, just as they are expected to have the BASIC built into them now," says Chamberlin, the MTU engineer. "For one thing, the sound chips are relatively cheap in large quantities, so there's no real reason not to."

SID is a hybrid digital/analog device with programmable attack, decay, sustain, and release for each of its three voices, a master volume control, a choice of four waveforms, 16-bit frequency resolution over a nine-octave range, and programmable high-, low-, band-, and notch-pass filters.

"The SID chip is basically a synthesizer on a chip," says its designer, Bob Yannes. "I played with synthesizers for years, so I'm quite familiar with them. I tried to put it all on a chip with the SID chip."

Yannes designed SID while an engineer for MOS Technology, which is owned by Commodore. He recently left Commodore to form his own company, Peripheral Visions, Inc. Although he won't say for sure what new products his company will introduce, it seems likely that computerized sound devices will be among them. He says chips such as SID are the key.

"There's no reason we can't take music systems being sold now for $4000 and bring them out for consumers for around $400 or $500 – a ten to one cost reduction. I consider the [Commodore] 64 to be only the first step. In the future I'd like to see something totally digital. I think that's the way to go.... I pretty much got the features that I wanted out of the SID chip in the 64, but not the performance I wanted. But now that I've done it once, I think I have a better idea about how to go about it next time."

Yannes says he was given specifications by Commodore only to develop a "sound chip," and then he decided to make it as much like a synthesizer as possible. But he had to work within the limitations of marketing considerations. For example, although SID allows each voice's envelope to be individually programmed, all three voices share the same volume control.

"I had to put separate envelope controls for each oscillator [voice] into the SID chip in order to satisfy the video game/sound effects marketing demands. If I had my way, the three oscillators would work in unison to create one voice. Anyway, that's why there're separate envelope controls for each oscillator but only one peak amplitude [volume] control – it was designed to function as one voice. You could vary the attack of the different oscillators, for example, to get a brassy sort of sound that way."

But Yannes bestowed SID with yet another feature to compensate for this limitation – an input line. It's possible to feed an outside sound source into a computer equipped with SID, process it through the chip's filters and volume controls, and output the extra source as a "fourth voice" in accompaniment with SID's regular three voices. In the case of the Commodore 64, for instance, the outside source would be routed through the RF modulator to the TV speaker – or a stereo system.

What kind of outside sources can be fed into SID? "You name it," says Yannes. "Tape recorders, radios, electric guitars, even another SID chip."

Note that last item: another SID chip. "One thing I thought you might be able to do is chain a bunch of SID chips together to get even better sound, without having to use external hardware," explains Yannes. "I designed the SID chip as a

standard 6502 peripheral chip with all the proper bus signals. You could put some SID chips in a cartridge and plug it into the 64, or the VIC-20, or the Atari – any 6502, 6809, or even 68000 system, even the Radio Shack Color Computer. It only requires 32 address locations, and the chips are pretty cheap, so there's not much to stop you."

It's an exciting prospect, but Commodore controls the SID chip, not Yannes. And for now, Commodore needs virtually all the SID chips it can make to meet demand for the new 64, plus the upcoming Max Machine, P Series, B Series, and BX Series computers soon to hit the market.

Still, a few SID chips have reached private hands, and the results are fulfilling their creator's hopes.

The Synthesizers Of Tomorrow

Chamberlin, the MTU engineer, got four SID chips from his friend, Yannes. Chamberlin used them to make a prototype sound board for the MTU-130, a high-end personal computer for which he designed most of the circuitry. He then passed the board and SID chips along to another friend, Frank Covitz, the New Jersey research scientist. Covitz added four more SID chips to the board, for a total of 24 individually programmable voices. The board is plugged into an MTU-130 equipped with an organ keyboard which, in turn, is controlled by its own 6502 microprocessor.

The instrument made its first public appearance recently when Covitz's son, Philip, gave a performance at the Personal Computer in the Arts Festival in Philadelphia.

Ironically, Covitz says he didn't play his own invention at the festival because he's not a good enough musician. But he's working on software which not only will exploit the instrument's souped-up capabilities, but which also will make it playable by mediocre musicians. This is called *non-realtime* playing.

Musical instruments are usually played in what's known as *realtime:* the music is heard instantaneously as the musician plays the instrument. When an instrument is played in non-realtime, the keying of notes is a separate event from the playing of the music. Notes are entered (the computer instrument is programmed), and then played back (the program is run).

An example of this on home computers is the Atari *Music Composer* cartridge. Essentially, it does for music composition what word processing does for writing. Notes are entered on the computer keyboard and plotted on staffs drawn on the screen. The notes, measures, and phrases can be edited and arranged at will, then played back at the touch of a key. Similar composition programs are available for other personal computers.

"One of the things that computers can do is change music from a physical endeavor to a programming endeavor," says Chamberlin. "That's one of the reasons why I got into computer music – my total lack of dexterity. Even if you're a total butterfingers like me, you can experiment with computer music."

Covitz is striving to push the concept even further. He's added four special keys to his prototype board: Record, Play, Fast Forward, and Rewind. But don't mistake it for a conventional tape recorder – the keys are similar in function, but not in method.

When the Record button is pressed, the computer will "remember" whatever music is played. But no recording tape is involved. Instead, each keypress on the organ keyboard generates information coded in four bytes: which key was pressed, the velocity (how hard the note was played), and the exact moment the key was pressed, accurate to a split-second. Another four bytes of information are generated when the key is released, for a total of eight bytes per note. All this information is stored in memory so the music can be reconstructed later.

After a musical part is "recorded," the Play button can be used to play it back – in accompaniment with a matching musical part played by the musician on the organ keyboard. And this duet, in turn, can be "recorded" in memory by a second unit. Using just two of these "memory recorders," the process can be repeated again and again, layering sound upon sound.

While the same thing can be done with conventional tape recorders, the sound would deteriorate with each generation of re-recording. Tape hiss and other defects would soon overcome the music. But since Covitz's instrument "records" the sound digitally, there is no degradation whatsoever. Beyond that, the music can be "edited." If a note is missed, the musician can correct it by rewriting the correct values into memory.

"This is what I see as the ultimate system," says Covitz. "Right now, this software doesn't exist anywhere except in my mind. I'm in the process of working on this now, and it's all being done in machine language."

The brief history of home computing – and indeed, home computing itself – indicates that advanced technology eventually works its way down to the personal level. It's not hard to envision the day when plug-in organ keyboards and cartridges with add-on synthesizer chips will transform home computers into the kind of instruments Covitz is experimenting with now.

"Seeing what the SID chip can do, and do digitally, I expect you'll see an explosion of that sort of thing," says Covitz. "There has to be. It doesn't require very much hardware. There definitely will be an explosion in complexity." ©


PETSPEED
Now for COMMODORE 64!
FAST ENOUGH FOR THE HUMAN RACE
Our alien won't hang around for slow software. He wants crisp responses and really fast processing.
For the human race too, slow PET BASIC is not good enough. When we run a program, whatever it is, we want fast efficient action.
PETSPEED, the compiler recommended by Commodore, is now available for the 64 and CBM 2. It can make any BASIC program run many times faster. It even speeds up disk handling. We guarantee that PETSPEED is easier to use and generates faster code than any other BASIC compiler for Commodore Systems.
Using PETSPEED is simple. Just type in the name of the program, wait a few minutes and then watch your software run up to 40 times faster.
Petspeed is not simply a compiler, it contains a powerful OPTIMISER. While PETSPEED is compiling, it breaks your program down into tiny fragments and reassembles it removing the unnecessary and simplifying the complex. Dazzling graphics. Lightning sorts. With PETSPEED anything is possible.
Also available INTEGER BASIC COMPILER — 150 to 200 times the speed of Basic. Integer Basic is for those applications where the speed of machine code is required without the inconvenience of assembly level programming. Ideal for scientific and educational users. Compatible with Petspeed.
PETSPEED (Commodore 64) $150
PETSPEED (8000 or 4000 series) $300
INTEGER BASIC (8000 or 4000 series) $175
SPECIAL OFFER: Petspeed PLUS Integer Basic $435
SSE (415) 964-8201
SMALL SYSTEMS ENGINEERING
1056 Elwell Court • Palo Alto, CA 94303

Part I:

Writing Transportable BASIC

Edward T Ordman
Department of Mathematica Sciences
Memphis State University

If you think your programs might ever be used on another computer with a different dialect of BASIC – the suggestions in this two-part article can go a long way towards easing the transition. This month the author covers documentation, vocabulary, and readability. The article concludes next month with an overview of highly machine-sensitive issues such as input-output and graphics.

So you finally got your own computer. Unfortunately, it is not the same model you had at school. Or you've arrived at high school or college and the computer there is not the same one that your junior high school or high school had. What are you going to do with all the programs you have accumulated? My own school has just bought several of the new IBM Personal Computers – but most of the programs we have on hand were written for a mainframe or for our OSI microcomputers. Come to think of it, we are changing mainframes next semester, too!

Of course, all of these machines have a version of BASIC. (Some of them, in fact, have several versions of BASIC.) But, as is clear to anyone who has read a program written in Apple BASIC and wished he could run it on his Atari (or PET or TRS-80 or ...), all BASIC interpreters are not the same.

What is the solution? There is no ideal solution, for all cases. Some published programs are difficult to convert from one dialect to another. We can, however, in writing programs for ourselves, for friends, and perhaps even for publication, try to make our programs *transportable*. That is, we can write the programs so that they can be adapted to another machine with a minimum of difficulty.

Self-documenting

A program is easily transportable from one machine to another if it can be entered and run in the second machine with no substantial rewriting – certainly no changes in the underlying logic or algorithms – and a minimum of minor changes. The program should be self-explanatory so that it can be rewritten without knowledge of the original machine – a knowledge of the machine we are rewriting it for should be enough.

I have one fairly complex simulation program that was first written about 12 years ago for a PDP-8. It has since been rewritten, by me or by others, for S-100 bus machines in CBASIC, Apple, TRS-80, IBM Personal Computers and IBM 370's, Xerox Sigma 9, PDP-11, and enough other machines that I have lost count. I suspect that it would have been forgotten after the second or third transportation to a new machine, if it had not been written so that it was usually just a matter of typing it in again.

I should warn you at the outset that all this article considers is how to write the BASIC program. It does not address the problems of getting a program from one machine to another without having to key it in again. Increasingly, it is possible to connect the two computers over a phone line, directly or via one of the dial-up timesharing services, and move the program as a text file to avoid retyping. Nevertheless, the focus of this article is transportable *programming* techniques.

What can you do, when writing a program, to make it easily transportable? We will divide the strategy into five main parts: 1) minimal vocabulary; 2) in-program readability; 3) formal structuring; 4) careful attention to input-output; and 5) limited graphics.

Minimal Vocabulary

First, let's consider the question of vocabulary – what features of BASIC we should use. Apparently, whenever a company produces a new computer or a new version of BASIC, it feels compelled to add features not found in anyone else's BASIC. Often these features are convenient and may make programming for that machine easier. However, they make transporting a program much harder. If at all possible, such features should be avoided when writing with transportability in mind.

MEET THE NEW KID ON THE BLOCK.

Catch-up with the newest, wackiest video game you've ever played! Kid Grid! But don't get caught with your guard down because the mischievous bullies Squashface, Thuggy, Muggy and Moose are after you in hot pursuit.

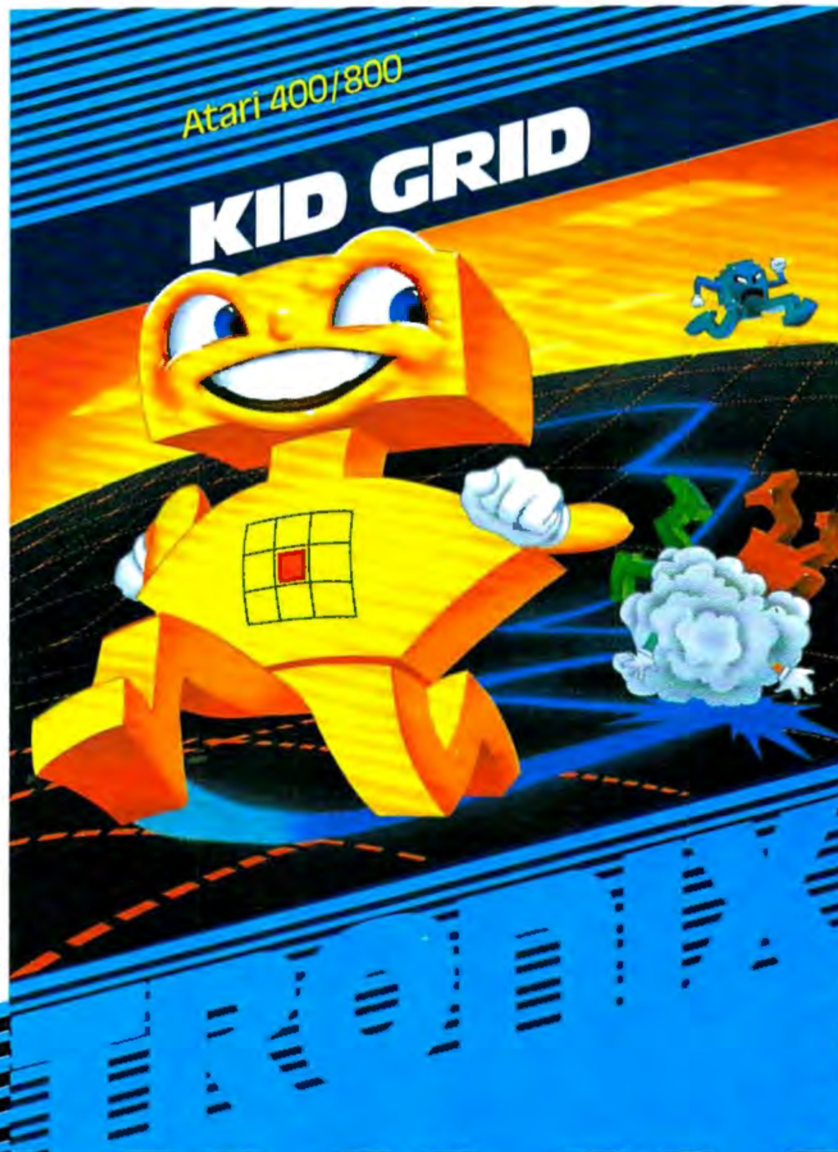
It's the wildest, fastest chase you've ever been in. You've got to connect all the dots on your grid: keep on the look-out for the mysterious bouncing question mark and don't slow down at corners! Squashface, Thuggy, Muggy and Moose are always on your shirttails. Your secret weapon — the joystick button. Press it and the bullies get zapped! When they turn white and freeze, make your escape. If the bullies finally catch you the results are explosive!

With Kid Grid you get all the sights, sounds and colors of arcade games. The better you get the faster and wackier Kid Grid becomes. Soon you'll be racing around at hyper-space speeds!

Kid Grid by Tronix. A hi-resolution video game written in 100% machine code. Designed for the Atari 400 & 800 home computer. Available now at your dealer for $29.95 (suggested retail price).

So meet the new Kid on the block. He'll be your pal forever.

Dealer inquiries invited.
Tronix Publishing, Inc., 701 W. Manchester Blvd. Inglewood, CA 90301 (213) 671-8440



TRONIX

If we must use special features, they should be isolated in a subroutine near the end of the program and clearly labelled. The main program should stick to features found in virtually all versions of BASIC. This does not mean that string handling must be restricted to the limitations of Radio Shack Level 1 BASIC, which is an extreme example; nor are there universal rules as to what constructions are allowed. Some textbooks define "minimal BASIC" or restrict themselves in a similar way.

Educational institutions often belong to groups (consortia) which promote standards for exchanging programs; CONDUIT is one such educational group with a nice pamphlet on standard BASIC. If you have worked with several versions of BASIC, sticking to common features is a good guide for what will be transportable between them. For informal use, however, or for the individual who has just worked on one machine, here are the standards I have found useful in working with perhaps a dozen different machines, large and small.

Variables And Commands

Figure 1 suggests some guidelines for variable names, numbers, line numbers, DIM statements. Clearly, the list could be made much longer. For instance, how big can a real number be and not overflow? How small can a positive number be and still be distinguished from zero? Most BASIC programs do not depend critically on these figures, which may differ dramatically from one system to another.

Figure 1: Variables and Numbers

Line numbers: 1 to 9999

Variable names: One letter or one letter and one digit. Strings, one letter and $. Examples:

A B2 C9 F$ Z$

Dimensions: Always declared if needed; execute the DIM statement once, before using the variables. If possible, stick to one subscript for strings, two for numbers. Do not use variable sizes or reuse letters. DIM C$(50),D(20,10) is good; DIM B(N), A(50), A$(20) is bad.

If your program does depend on them, you should probably make this explicit (and include a REMark giving the limits on your system). For instance, if your program has a variable X that gets closer and closer to zero as you go around a loop, and you exit the loop by testing IF X=0 THEN ... , the program may behave very differently or even fail on another computer. Changing this to

```
500 IF ABS(X)<1E-50 THEN ... : REM USE A SMALL
    NON-ZERO NUMBER
```

will make the program transportable: the person converting it can check to see if the new computer will accept 1E-50. If it will not, he can substitute an acceptable number, e.g., 1E-30.

Figure 2: The most common statements

DATA	LET
DIM	NEXT
END	ON...GOTO
FOR...TO...STEP	PRINT
GOSUB	READ
GOTO	RESTORE
IF...THEN...	RETURN
INPUT	

Figure 2 shows a limited list of BASIC commands – a very limited list. While almost every BASIC accepts more commands than these, they differ on which statements those are. For each command not on this list, there is some computer around that will not accept it. To make matters worse, computers differ substantially in how they interpret some of these commands. Some, for instance, do strange things on a STOP but allow END only as the last line of a program. The cure: place 9999 END as the last line of the program, and terminate anywhere else by GOTO 9999.

GOTO and GOSUB should be followed just by a line number. GOTO 500 is fine; avoid GOTO A even if your computer likes it. In the statement FOR X=A TO B STEP C, it is best to restrict A, B, and C to integers (or expressions evaluating to integers) and to avoid changing them inside the loop. NEXT must name just one variable for the corresponding FOR, e.g., NEXT X.

IF...THEN statements require special attention, since so many computers have so many different extensions. A few computers accept only statements such as IF Y>=Z THEN 830, prohibiting calculations, logical operations, and not allowing anything but a line number after THEN. I am not seriously suggesting that you keep things this simple: the extensions are extremely helpful. However, it *is* a good idea to keep things simple enough so that your statements can be translated into this form. This will be discussed further in the section on structure, next month.

Numeric And String Functions

Figure 3 shows the most commonly implemented numeric functions. Either most BASICs have these functions, or the programmer using the machine will be prepared to fake them somehow. Two deserve special mention: RND and TAB.

RND is implemented differently on almost every computer. Some use X=RND, some use

Figure 3: The most common numeric functions

ABS	COS	INT
RND	SIN	TAB
ATN	EXP	LOG
SGN	SQR	TAN

X = RND(0), some use RND(1), some use RANDOMIZE to start (seed) the random number generator and some do not. You should assume that every line containing RND will have to be rewritten. You should make this as easy as possible, by minimizing the number of lines involved and making your intention clear. If you need a random number in 20 different places in your program, do not have RND appear in 20 places; place it in a subroutine. That is, incorporate in your program

```
9000 REM *** GET RANDOM NUMBER, CHANG
     E FOR OTHER COMPUTERS ***
9010 X=RND(1):REM  RANDOM,0<X<1,NEW S
     EQUENCE EACH RUN
9020 RETURN
```

and then place GOSUB 9000 wherever needed in your program. Here is a more typical use, near the start of a game program:

```
150 N = INT(100*RND) + 1 :REM RANDOM INTEGER
    1 TO 100 ***************
```

Here the string of asterisks warns you, when transporting the program, that the line is likely to change. The remark tells what is wanted and will save a lot of time if the new computer achieves this by N = RND(100).

Turning briefly to TAB : there are computers that like TAB(N) (go to column N), those that like SPC(N) (print N spaces), those that like both, and those that like neither. Most people know how to juggle spacing on their own machine, so making your intention clear (by remarks or a sample printout) is probably more important than the exact way you write your PRINT statements. There will be more on this in the discussion of input-output, next month.

Figure 4: The most common string functions

ASC(X$)	LEFT$(A$,N)
CHR$(N)	RIGHT$(A$,N)
VAL(X$)	MID$(A$,I,J)
STR$(X)	

The functions given in Figure 4 are now remarkably widespread in *microcomputers*. It is probably safe to use all of them freely in that context. That is, if the person rewriting the program does not have LEFT$, he probably has a reasonbly direct substitute. You cannot count on the format produced by STR$ being the same from one machine to another – some pad with blanks on the left, some on the right, some not at all. Functions that match a substring are present on many machines, but absent on many others. Many systems will crash if you call LEFT$(A$,N) and A$ has less than N characters, so you should always test for this before you call LEFT$ even if your system does not insist on it.

Large computers differ substantially in how they handle strings, and are often *more* restrictive than small computers. ASC and CHR$ are frequently absent; many large computers do not even use the ASCII character set. Avoid extensive string manipulations, or at least place them in a subroutine, if your program may have to run on a large mainframe next year.

Readability

Next, if our program is to be readily transportable to another version of BASIC, it must be readable. First, can the reader understand our individual lines, and translate them for the new system? Second, can the reader understand our general strategy or procedure (our *algorithm*) well enough to debug the program if errors creep in, or if his BASIC interprets some command very differently than expected?

The most important consideration, for the second of these, is to make the program sufficiently modular and to provide appropriate REMarks for each module; this is addressed more in the discussion of structure, later. There are a number of "tricks of the trade" that make individual lines easier to read, however. Here are a few principles:

1. Leave plenty of space between line numbers. Even if you have only one command per line, some one-line commands on your system may become multiple commands on another. If you use several commands per line, the situation gets far worse. This is not to condemn all multiple-command-per-line statements, since they can add to the clarity of the program. Just remember that while your computer may allow:

```
500 INPUT "WHAT IS YOUR NAME?"; N$
```

someone else's may require

```
500 PRINT "WHAT IS YOUR NAME";
501 INPUT N$
```

This is an easy change if you left a line number available. It is quite possible for a complex one-line statement on one system to require six or eight lines on another.

2. Leave plenty of blanks in your commands, where appropriate. You may have no trouble understanding 250PRINTT5 or 300FORI5 = PTOM

PET/CBM

PaperClip

Professional Word Processor at a Breakthrough Price

PaperClip™ performs all the advanced features found in Word Processors costing much more. . .

1) Full screen editing. 2) Copy/Transfer sentences and paragraphs. 3) Insert/Delete sentences and paragraphs. 4) Headers/Footers/Automatic page numbering. 5) Justification/Centering. 6) User defineable keyphrases. 7) Supports both cassette and disk. 8) Variable data – Form letters. 9) Horizontal scrolling up to 126 characters. 10) Insert/transfer/erase columns of numbers. 11) Add/subtract columns of numbers. 12) Supports most dot matrix and letter quality printers. In fact, a printer set-up routine is supplied to take the best advantage of the printer at hand. 13) French and Math technical character sets available.

Also available for Commodore 64

Requires
Basic 4.0, 32K memory.

$125.00 U.S.

Dealer enquiries welcome

BATTERIES INCLUDED

71 McCaul Street
Toronto, Ontario
Canada M5T 2X1
(416) 596-1405

but a reader will find 250 PRINT T5 and 300 FOR I5 = P TO M much easier to copy or edit. Many BASICs *do* insist on the spaces; the new IBM Personal Computer is one that does. Your computer may allow a larger program or run faster if you delete spaces and remarks, but you make the program much harder to transport when you delete them. It may be worth keeping two programs, a transportable copy and a condensed, quick-run copy.

3. Avoid unprintable characters. Where a few are necessary, find a way to make their presence visible. For instance, a disk read in Applesoft requires that you PRINT a CONTROL-D followed by a string. You can make this readable by

```
200 D$ = CHR$(4) :REM CONTROL-D
...
540 PRINT D$;"OPEN FILENAME" :REM DOS
    COMMAND STARTS CTRL-D
```

It is a good idea to indicate what other CHR$ characters are when they are created, too – for instance when CHR$ is used to put a quote mark into a string, or manipulate carriage returns or line feeds.

4. Identify specific features you depend on. This happens most often in connection with PRINT and INPUT statements. Most of us can guess what someone else's PRINT statements are supposed to do, but the INPUTs are another matter.

Some systems input a sentence like "TODAY IT RAINS" by INPUT A$ and the response ?TODAY IT RAINS; others by INPUT A$ and response ?"TODAY IT RAINS"; others by INPUT LINE A$ or by LINPUT A$ or even by INPUT (FIELD 40) A$. You can make this clear to the reader – so that he can try to do the appropriate thing on his system – by remarks, but clear user instructions within the program are probably even better. For example,

```
110 PRINT "TYPE IN A SENTENCE SURROUNDED
    BY QUOTE MARKS"
120 INPUT A$ :REM SAMPLE "HELLO, JOE,
    WADDAYA KNOW."
```

5. Make cues to the user extremely clear. Remember that you won't be around to show people how to use it; in fact, *no* expert on the program will be around. Give sample answers whenever possible, and protect against invalid answers.

```
130 PRINT"DO YOU WANT TO PLAY AGAIN (
    Y/N)";
140 INPUT A$
150 IF A$="N" THEN 9999
160 IF A$<>"Y" THEN 130
```

Note that invalid answers will cause the question to be asked again.

Next month, examples of portable program structure, input-output, and graphics programming. ©

Mattel's New Home Computer

Tom R. Halfhill, Features Editor

Judging from the inquiries we've been receiving at **COMPUTE!**, people are having a tough time choosing between the current crop of low-end home computers: the Atari 400, Commodore VIC-20, Radio Shack Color Computer, Sinclair/Timex, and Texas Instruments TI-99/4A.

Well, it's about to get even tougher.

Mattel Electronics has announced a home computer aimed squarely at the low-end market. It is *not* to be confused with the long-delayed Intellivision keyboard attachment – which has been redesigned again, incidentally. Mattel says the new computer, dubbed the Aquarius, is due "very early in 1983."

The Aquarius will sell for under $200 retail. The price is expected to vary because the machine will be sold through mass consumer outlets. This means it will be available at a wide range of TV-electronics stores, audio/video shops, department stores, discount stores, and catalog showrooms. As we've seen with the other low-end home computers being marketed this way, prices are greatly discounted because competition is so fierce.

Expands To 52K RAM And CP/M

The Aquarius will come with 4K of Random Access Memory (RAM), expandable to 52K RAM in 4K and 16K steps with plug-in cartridges. Microsoft BASIC is built-in. There is one voice for sound effects or music, and an expansion option (described below) provides three voices. The maximum graphics resolution is 320 by 200 pixels (screen dots) in 16 colors.

The Aquarius can display 256 characters. This includes a 128-character ASCII set with upper- and lowercase, and 128 user-programmable characters, similar to the redefinable character sets on the Atari, Commodore 64, VIC-20, and TI-99/4A computers.

For the Central Processing Unit, the central "brain" of the computer, Mattel chose the Z-80A, an eight-bit microprocessor chip. The use of this chip allowed Mattel to give the Aquarius CP/M capability. CP/M (Control Program for Microcomputers) is an operating system primarily used for business applications. The Aquarius can run CP/M with the addition of a disk drive and disk controller card. It is very unusual that a low-end home computer would have CP/M capability, but a Mattel official explained, "Some home users will be professionals who will prefer to work at home." With CP/M, a huge library of existing business programs will work on the Aquarius.

The keyboard has 49 keys. It's more than the membrane keyboard found on the Atari 400, but is not quite a full-stroke typewriter keyboard like the VIC-20's. The keystroke travel is 1.5 to 2 millimeters, and the keys are made of a rubber-like material instead of hard plastic, similar to the keys on Sinclair's new ZX Spectrum (see **COMPUTE!**, August 1982). The keyboard accepts overlays for special applications. For example, an overlay for BASIC programming allows one-key entry of BASIC commands.

While all of these features are standard in the under-$200 Aquarius, Mattel says it will offer a complete system "in the $500 range" which will include a data recorder (tape drive), a printer, and the Aquarius Mini-Expander. The Mini-Expander is an attachment which adds three-voice sound, two game controllers on eight-foot cords, and two slots for plug-in cartridges. One slot is for memory expansion and the other accepts cartridge programs.

All of the software initially released for the Aquarius will be on cartridges. Mattel promises that eight to ten cartridges will be available when the Aquarius is introduced. This will include education, home management, personal improvement, and entertainment software.

One cartridge will be a low-cost Logo with turtle graphics, the acclaimed learning language for children. Mattel says its Logo is designed to work on a minimum system without extra accessories. Another cartridge will be *FileForm*, a word processor. Mattel says more cartridges will be released monthly after the computer is introduced, and that additional software may be available on cassette in the future. All the software has been developed at Mattel, although the company is now talking to outside developers.

Other expected add-ons include a disk drive and a 40-column thermal printer. No prices have been disclosed for these extras, but a Mattel official did say that the memory expansion boards would be "extremely price-competitive."

Apparently, the Aquarius will lack special function keys, sprite graphics, and full-screen editing. Reportedly, it will have 8K of ROM (Read Only Memory), a 40 by 24 screen display, dimensions of 13 by 6 by 2 inches, and weigh four pounds. It will come with simplified instruction cards so beginners can get the machine working without reading the manual. ©

Richard Mansfield, Senior Editor

Myths About Programming

Whenever someone says that they are "not the type" or they "weren't good in math" as a reason for not learning to program in BASIC, I like to compare learning BASIC to learning how to drive a car. A few people never learn to drive, but most of us do. What's more, it takes a few weeks at most to catch on to either driving or BASIC. You improve over the years, but the essentials can be grasped pretty quickly. And in the coming Age of Information, not knowing how to program could well be as inconvenient as not having a driver's license is today.

After all, there are only some 50 BASIC words to learn. Several of them are very rarely needed (you can look through hundreds of programs and never find WAIT, POS, or TAN). Also, many BASIC words mean exactly what they say: STOP stops a program, RUN runs, END ends.

If someone is still doubtful, the most convincing argument is a demonstration. You can say: "Let's try something complex. How about printing your name 1,000 times on the screen?" Then type: 10 ? "Alan";: GOTO 10.

Could It Explode?

Another factor which causes hesitation about learning programming is a fear of the unknown. It's not hard to see where this nervousness comes from. There is a category in movies which changes each generation, but could be called the Frankenstein Slot. A scientist gets too big for his britches, tampers with unknown forces, and his creation runs amok while the villagers, in their ancient wisdom, ineffectually stone the laboratory.

During the fifties there were dozens of films in which atomic radiation filled this slot. It caused moths and ants to grow to enormous size, rampaging through cities. When people learned that radiation, dangerous as it can be, could not cause insect giantism, popular entertainment found a new monster. Dozens of movies in the sixties (some of them excellent, like *2001*, *The Forbin Project*, and more recently, *The Demon Seed*) portrayed the computer as Frankenstein.

When people buy their first personal computer, they are not generally worried that it might destroy their house, but they often worry about the computer getting out of their control and damaging itself. They sense, correctly, that a computer is a powerful machine.

You'll see this hesitancy when people look up, their first time in front of the keyboard, and ask, "What should I be careful of? Can I hurt it?" We get letters from beginners wanting to know if they should use POKE, the BASIC word that changes what's in the computer's memory. They have a perfectly understandable fear that, as one New Yorker recently wrote, "I might damage the BASIC ROM chips."

Your computer watches out for most kinds of errors. It simply won't allow you to POKE into BASIC ROM chips. If you try to send a POKE to an address that's in ROM (Read Only Memory), it will just bounce off. Nothing happens. These addresses can only be read, not written (POKEd) to.

We always used to say that nothing you could type into the computer could hurt it in any way. If it doesn't understand what you write, or can't carry out your instructions, it will stop and tell you where it stopped and, in general terms, why. The worst that could happen would be an "endless loop," and you would have to turn power off and back on to get control. None of this, however, would start the computer smoldering.

This advice, unfortunately, is not 100 percent correct. It was discovered that there was one POKE to PETs with Original or Upgrade BASIC versions which would make the video display about six times faster. It's POKE 59458,62. PET/CBMs with the most recent 4.0 BASIC chips, however, *can* be damaged by this POKE if left running without turning off the power. Since there are about 17 million combinations of POKEs you can make into your computer, the odds are clearly against your accidentally making this error. Nevertheless, it does make it impossible to tell someone that nothing you type into any computer could hurt it.

NOT EVERYONE CAN TEACH THEIR ATARI™ NEW TRICKS...

WE MAKE USING AND LEARNING ABOUT COMPUTERS FUN!

PROGRAMMING GUIDE FOR BEGINNERS OR EXPERTS — MASTER MEMORY MAP.™ A 32 page book with hundreds of hints on how to use your computer. Over 500 memory locations! $6.95.

LEARN SOUND AND GRAPHICS with our exciting lessons called **TRICKY TUTORIALS.™** Each comes with a tape or disk full of examples, and a 12 to 64 page manual written in an easy to understand manner. **#1 DISPLAY LISTS** — Put several graphics modes on your screen at once. **#2 SCROLLING** — Move text or graphics smoothly up, down, sideways, or diagonally. **#3 PAGE FLIPPING** — Change TV screens as quickly as flipping pages in a book. **#4 BASICS OF ANIMATION** — A beginner's lesson in animation using PLOT, PRINT, and a surprise game. **#5 PLAYER MISSILE GRAPHICS** — Learn the basics of writing your own arcade games. **#6 SOUND & MUSIC** — Simple methods to play complete songs, with graphics. Includes PLAYER PIANO free! **#7 DISK UTILITIES** — 7 programs to help you use your disk drive. 32K. **#8 CHARACTER GRAPHICS** — The best editor available with examples using special characters YOU CREATE and ANIMATE. **#9 GTIA, GRAPHICS 9 to 11** — New tricks you can do with these 16 color modes. **#10 SOUND EFFECTS** — Many examples, from rainfall to laser blasts, with ample explanation. **#11 MEMORY MAP TUTORIAL** — 30 colorful examples of tricks your computer can do.

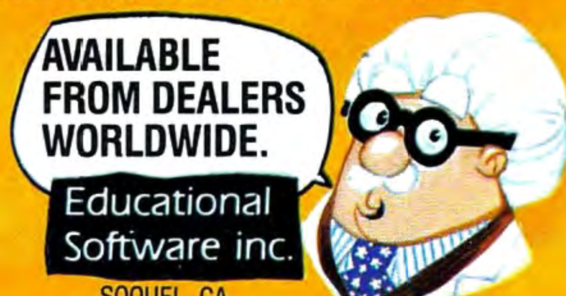
TUTORIALS cost only $19.95 each, except 5 and 7 which cost $29.95 each. 16K Tape/24K Disk required.

SPECIAL: Tutorials 1 through 6 in a binder for $99.95. SAVE $30.00!

USER SUBMITTED PROGRAMS

We sell many fine programs written by dedicated computer owners, such as **INSTEDIT, SPACE GAMES, MINI WORDPROCESSOR, DATA-BASE DIALER, PROTO'S ADVENTURES,** and many more.

OUR GUARANTEE: Your money back if unsatisfied!



Educational Software inc.

SOQUEL, CA

NO LOCAL DEALER? CALL FOR A FREE CATALOG, OR ORDER DIRECT (CHARGE OR COD): 800-692-9520 OR (408) 476-4901

The Math Myth

When autos were first becoming popular, there were doubtless many people who refused to try driving, saying, "That's one thing that I will never try to learn. I could never control our horse." A weakness in math is often given as the reason for not trying programming. In fact, the two activities are hardly related. Mathematical words are available in BASIC, but programming does not need to involve much math beyond simple arithmetic unless you choose to solve mathematical problems. You will need to search a long time to find any use of the word SIN in the dozens of BASIC programs published each month in **COMPUTE!**.

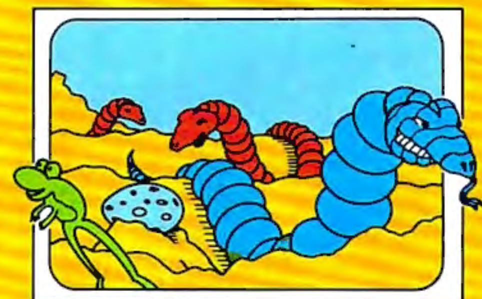
Personal computers are general-purpose tools. They *can* be used to solve complex equations, but to call programming "mathematical" would be too narrow it down to only one of its countless applications. And it would also mislead people into thinking that they need a special talent in math to become competent programmers.

Several years ago one of America's largest corporations undertook a study to find out what its best programmers had studied in college. To nearly everyone's surprise, English and music were the most common majors among the top computerists. Perhaps this is because these disciplines stress creativity and attention to detail. Perhaps they combine logical thinking with imagination. No one has yet given a satisfactory explanation.

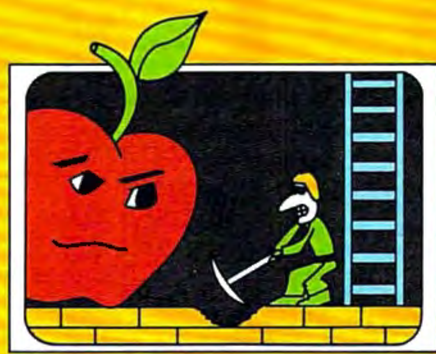
In any case, you don't need to become an ace programmer any more than you need to drive at the Indianapolis 500. To get where you want to go, ordinary driving or programming knowledge will suffice. There are very few people who can't learn the necessary skills.

If there is a topic that you would like to see discussed in this column, send a card or letter to: The Beginner's Page, **COMPUTE!** *Magazine, P.O. Box 5406, Greensboro, NC 27403.* ©

VIC-20® ?

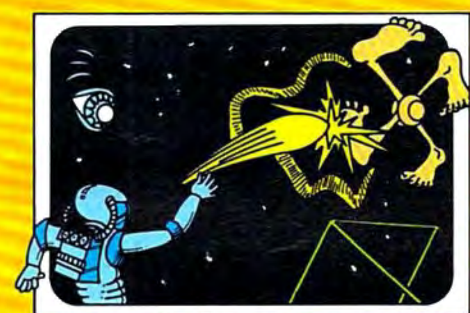


SERPENTINE



APPLE PANIC

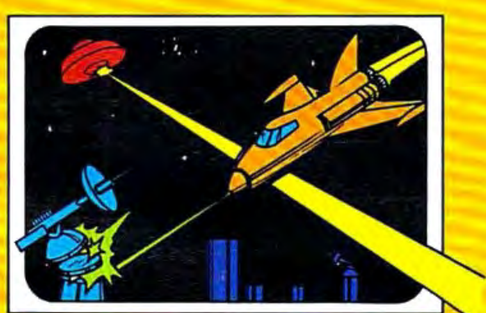
CHOPLIFTER



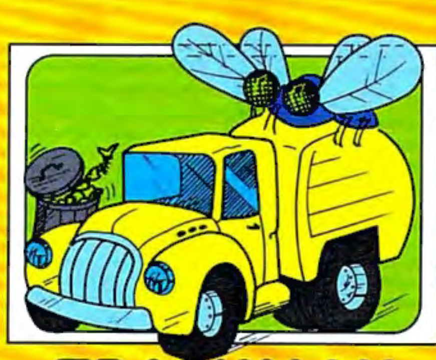
VIDEO MANIA

and these are just the games!

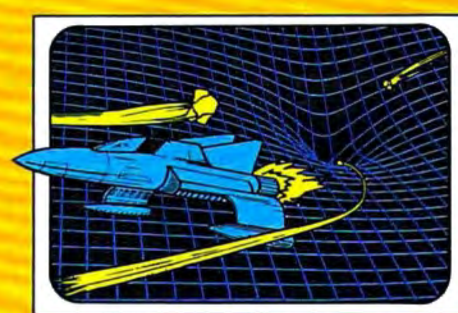
TERRAGUARD



ASTROBLITZ



TRASHMAN



BLACK HOLE

YOUR FULL LINE VIC-20® SOFTWARE COMPANY

CREATIVE SOFTWARE

201 San Antonio Circle • Mountain View, CA 94040

Atari's Sound System

John Scarborough, Novato, CA

If you're interested in the improvements to Atari sound and music possible via machine language – this will get you started. These simple demonstrations might convince you to abandon the SOUND command entirely.

Many programmers who decide to make the jump from BASIC programming to machine language programming find frustration in their attempt to print to the screen or produce sound. The problem is that there are now no print or sound statements. Machine language deals entirely with retrieving, manipulating, and storing data.

But even after the programmer discovers this it won't do him much good unless he knows where and what to store to produce results. Furthermore, this information is often not provided in the manuals that come with the computer. So how does he obtain this information? He must turn to other methods. Four common ones are:

1. The trial and error method (very inefficient, but sometimes necessary).

2. Advanced user's manuals.

3. Information obtained from a human source, such as from friends or teachers.

4. A magazine.

This article is in category four. After studying this article, you will have more control over the four voices provided by the Atari. The article will also briefly cover the built-in speaker.

Sound Commands

Look at the following sound command:

```
SOUND 0,121,X,X
```

This instructs the computer to store a value of 121, which will produce a middle C note, into Audio Frequency Control register 0. This register is located at memory location 53760 ($D200 hex). Thus, the following two commands will function identically:

```
SOUND 0,121,X,X
POKE 53760,121
```

The three remaining Audio Frequency Control registers are located at 53762 ($D202), 53764 ($D204), and 53766 ($D206). A POKE 53764,128 would store a value of 128 (a B note) into the Voice 2 Audio Frequency Control register. SOUND 2,128,X,X will do the same. (See Figure 1 for a clearer representation of the four Audio Frequency Control registers.)

Now you can store a given frequency (note) into any of the four Audio Frequency Control registers. But what about distortion and volume? Look at the following sound command:

```
SOUND 0,X,10,12
```

This tells the computer to produce a pure tone (10) and a volume level of 12. Upon execution, the computer will convert the number 10 to 160 (160 is the actual pure tone code. See Figure 2 to find the corresponding distortion codes for the eight additional distortion levels), add 12 to it, and then store the result into Audio Control register 0. This register is located at memory location 53761 ($D201 hex). Thus, the next two commands will perform the same task:

```
SOUND 0,X,10,12
POKE 53761,160+12
```

The three remaining Audio Control registers are located at 53763 ($D203), 53765 ($D205), and 53767 ($D207). A POKE 53767,160+7 would store a pure tone and volume level of 7 into Audio Control register 3. SOUND 3,X,160,7 will do the same. (See Figure 1 for a clearer representation of the four Audio Control registers.)

You should now know how to store any given note, tone and volume level into any of the four voices provided by the Atari (and without using sound statements). The following two BASIC programs function identically:

```
10 SOUND 0,121,10,12
20 GOTO 20
```

```
10 POKE 53760,121
20 POKE 53761,160+12
30 GOTO 30
```

The Built-in Speaker

That is an overview of the sound that is channeled to the television speaker, but what about the built-

in speaker? The built-in speaker is controlled via location 53279 ($D01F).

Program 1 will make the built-in speaker randomly click. (Not a very spectacular sound effect, to be sure, but that's not to be expected from BASIC statements.) Program 2A and 2B (which function identically) will also make the built-in speaker randomly click, but they do not use BASIC statements to produce the sound and will therefore click the speaker much faster than will Program 1. compare the two for yourself. The comparison will give you some idea of the speed available to you from machine language programming.

If you compared Programs 1 and 2, you might have thought: "If machine language can do that much for that little built-in speaker...." Yes, by using machine language you can greatly increase the quality of the sound that comes out of your television speaker.

I leave you with a simple machine language program that will demonstrate this increase in quality. The program is written in both assembly language (3A) and BASIC (3B). This is a simple program; it would have been half as long without the delay routine. However, the sound would not be audible if there were no delay built into the machine language program!

Figure 1.

The Four Voices Provided By The Atari

VOICE 0	
Audio Frequency Control Register	- 53760 ($D200)
Audio Control Register	- 53761 ($D201)
VOICE 1	
Audio Frequency Control Register	- 53762 ($D202)
Audio Control Register	- 53763 ($D203)
VOICE 2	
Audio Frequency Control Register	- 53764 ($D204)
Audio Control Register	- 53765 ($D205)
VOICE 3	
Audio Frequency Control Register	- 53766 ($D206)
Audio Control Register	- 53767 ($D207)

Figure 2.

Examples Of The Eight Distortion Levels Using Random Voices

```
SOUND 0,X,0,V  = POKE 53761,0+V
SOUND 0,X,2,V  = POKE 53761,32+V
SOUND 3,X,4,V  = POKE 53767,64+V
SOUND 1,X,6,V  = POKE 53763,96+V
SOUND 0,X,8,V  = POKE 53761,128+V
SOUND 1,X,10,V = POKE 53763,160+V
SOUND 2,X,12,V = POKE 53765,192+V
SOUND 0,X,14,V = POKE 53761,224+V
```

Program 1.

```
5 REM -THIS PROGRAM UTILIZES THE BUIL
  T-IN SPEAKER USING BASIC STATEMENTS
10 A=INT(256*RND(1)):REM -LOAD A WITH
    A RANDOM NUMBER FROM 0 TO 255
20 POKE 53279,A:REM -STORE A AT 53279
30 GOTO 10:REM -START OVER
```

Program 2a.

```
5 ;THIS PROGRAM UTILIZES THE BUILT-IN
   SPEAKER FROM MACHINE LANGUAGE USIN
  G THE ASSEMBLER-EDITOR CARTRIDGE
10 *=$600
20 LOOP LDA $D20A ;LOAD A WITH A RAND
   OM NUMBER FROM 0 TO 255
30 STA 53279 ;STORE A AT 53279
40 JMP LOOP ;START OVER
```

Program 2b.

```
5 REM -THIS PROGRAM UTILIZES THE BUIL
  T-IN SPEAKER FROM MACHINE LANGUAGE
  USING THE BASIC CARTRIDGE
10 FOR LOOP=1536 TO 1544
20 READ DATA
30 POKE LOOP,DATA
40 NEXT LOOP
50 X=USR(1536)
60 DATA 173,10,210,141,31,208,76,0,6
```

Program 3a.

```
5 ;THIS PROGRAM UTILIZES THE TELEVISI
  ON SPEAKER FROM MACHINE LANGUAGE US
  ING THE ASSEMBLER-EDITOR CARTRIDGE
10 *=$600
20 FREQ=$660
30 ;STORE A PURE TONE (160) AND A VOL
   UME LEVEL OF 15 (160+15) INTO VOIC
   E 0
40 LDA #175
50 STA $D201
60 ;STORE CURRENT FREQUENCY INTO VOIC
   E 0
70 START LDX FREQ
80 STX $D200
90 INX ;INCREMENT FREQUENCY LEVEL
100 STX FREQ
110 ;DELAY PROGRAM EXECUTION
120 DELAY LDX #15
130 LOOP1 LDY #15
140 LOOP2 DEY
150   BNE LOOP2
160   DEX
170   BNE LOOP1
180   JMP START ;CONTINUE
```

Program 3b.

```
5 REM -THIS PROGRAM UTILIZES THE TELE
  VISION SPEAKER FROM MACHINE LANGUAG
  E USING THE BASIC CARTRIDGE
10 FOR LOOP=1536 TO 1563
20 READ DATA
30 POKE LOOP,DATA
40 NEXT LOOP
50 X=USR(1536)
60 DATA 169,175,141,1,210,174,96,6,14
   2,0,210,232,142,96,6,162,15,160,15
   ,136,208,253,202,208,248,76,5,6
```

VIC Sound Generator

Robert Lee, Vancouver, BC

Adding sounds to VIC can significantly slow down a BASIC program. The action stops and waits for the sound to finish. This could be especially annoying when you want a game to run as fast as possible. With this sound generator, you can add sounds in BASIC easily and without a speed penalty.

Among the novel features of the VIC-20 are its sound capabilities. These give it an advantage over the PET, bringing a new dimension to game programs. However, one of the problems I and undoubtedly other VIC owners have encountered is that, while manipulating the sound generators in a BASIC program, it is not possible to do anything else.

This is especially a problem in game programs written in BASIC and using extensive graphics. Either you have to write such programs without complex sound effects, or you have to settle for slow motion.

Faster Sound

Faced with this problem, I decided to write a machine language (ML) program for the VIC which adds speed to its sound generation capabilities. Most of the sound effects we use in game programs are sounds with increasing or decreasing tones. For example, a simple way to simulate the sound of a laser with the VIC is:

```
FOR K=250 TO 240 STEP -1 :POKE36876,K:NEXT
```

The ML program works along these lines, except that it is necessary to use only one POKE command. It generates sounds with increasing or decreasing frequency to make almost any kind of sound effect possible.

The program "VIC Sound" places a machine language program in the cassette buffer of the VIC. This means, of course, that you cannot transfer data using the cassette player while you are running the program. By changing the contents of memory locations 788-789 (decimal), the interrupt system of the computer is used to run the ML program.

As you know, the VIC has four "speakers" to make music and noise. The first and second speakers, activated by POKEing memory locations 36874 and 36875, are used for sounds with increasing tones. The third speaker (36876) is used for sounds with decreasing tones. The fourth speaker, activated by memory location 36877, is used mainly for explosions.

The ML program stores a starting number into the appropriate location and increases or decreases it for the period specified by the user. The interrupt of the computer will run through the program 60 times a second, which means that the starting number or tone will increase or decrease 60 times in one second.

Sound Duration

To make this a little clearer, let me explain that four memory locations have been assigned in the ML program to activate the four speakers, and four others to control the duration of the sounds.

Speaker	To activate	Duration
1st	846	858
2nd	847	888
3rd	848	918
4th	849	948

The number POKEd into locations 846-849 is the starting number which is stored in location 853 (dec); the initial value is 222, but this may be changed for the kind of sound you require. Locations 858, 888, 918, and 948 control the duration of the sounds. The program will generate the sounds for the number of jiffies (the 1/60th of a second interval used to measure time in Commodore machines) specified in these locations.

For a demonstration, RUN the program and then type SYS828; this will trap the interrupt. It will also set the volume control (location 36878) to maximum. Now POKE 846,222.

Location 858 contains 10 (dec), so the sound you heard was for ten jiffies. What the program has done is store 222 in location 36874 (first speaker), incremented it by one every 60th of a second until ten jiffies elapsed, then stored 0 into the memory location to switch off the speaker. To change the duration of the sound to, say, 20 jiffies, POKE 858,20. Now POKE 846,222.

The same method can be used for the other speakers. POKE 858,10. To change the starting number (i.e., to get a tone which starts higher or lower), simply POKE into memory location 853. For example, POKE 853,240. Now POKE 846,240.

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

Explosion Simulation

It is necessary to POKE the starting number into locations 846-849; any other number will give only silence. Try POKE 847,240 (second speaker); it gives a sound of increasing frequency like the first. Now POKE 853,222:POKE 848,222. You notice this gives a sound that decreases in frequency. POKE 849,222 will simulate an explosion. By manipulating the durations and starting number, you can get almost any kind of sound from the first three speakers and explosions from the fourth. However, when you are changing the duration of the sounds, make sure it is not too long; e.g., if you POKE 853,50:POKE 846,222 the program will store 222 in location 36874 and increment by one every jiffy for 50 jiffies. But in this case the contents of 36874 would increase to 255 and then cycle back to zero. You would hear a note for only 33 jiffies, since a number less than 128 in the sound generators of the VIC produces silence.

When using this program, you cannot generate sounds the normal way. To do so, you must first reset the interrupt vector by SYS996. This will also set the volume control to zero. To use the ML program, add the subroutine starting at line 8900 to your own BASIC program; and you can create sound effects using just one POKE, which would otherwise require a series of POKEs.

In a BASIC program with lines 8900-9240 added, you would first have a line like this in the main program to enter the ML into memory:

10 GOSUB 8900 : REM SOUND GENERATOR

```
10 PRINT"{CLEAR}"
20 PRINT"{03 DOWN}{08 RIGHT}{REV}V
    IC20{OFF}"
30 PRINT"{02 DOWN}{06 RIGHT}VIC SO
    UND"
800 GOSUB8900
900 END
8900 FORJ=828TO1019:READF:POKEJ,F:NE
    XT
9000 DATA169,15,141,14,144,120,169,8
    2
9010 DATA141,20,3,169,3,141,21,3
9020 DATA88,96,10,15,16,64,160,0
9030 DATA162,222,173,78,3,201,10,176

9040 DATA9,238,78,3,238,10,144,76
9050 DATA116,3,140,10,144,236,78,3
9060 DATA208,6,140,78,3,142,10,144
9070 DATA173,79,3,201,25,176,9,238
9080 DATA79,3,238,11,144,76,146,3
9090 DATA140,11,144,236,79,3,208,6
9100 DATA140,79,3,142,11,144,173,80
9110 DATA3,201,16,176,9,238,80,3
9120 DATA206,12,144,76,176,3,140,12
9130 DATA144,236,80,3,208,6,140,80
9140 DATA3,142,12,144,173,81,3,201
9150 DATA64,176,28,238,81,3,173,81
9160 DATA3,201,22,208,7,169,176,141
9170 DATA13,144,240,25,201,43,208,21
9180 DATA169,160,141,13,144,240,14,1
    40
9190 DATA13,144,236,81,3,208,6,140
9200 DATA81,3,142,13,144,76,191,234
9210 DATA169,0,141,14,144,120,169,19
    1
9220 DATA141,20,3,169,234,141,21,3
9230 DATA88,96,0,0,0,0,0,0
9240 RETURN
```

©

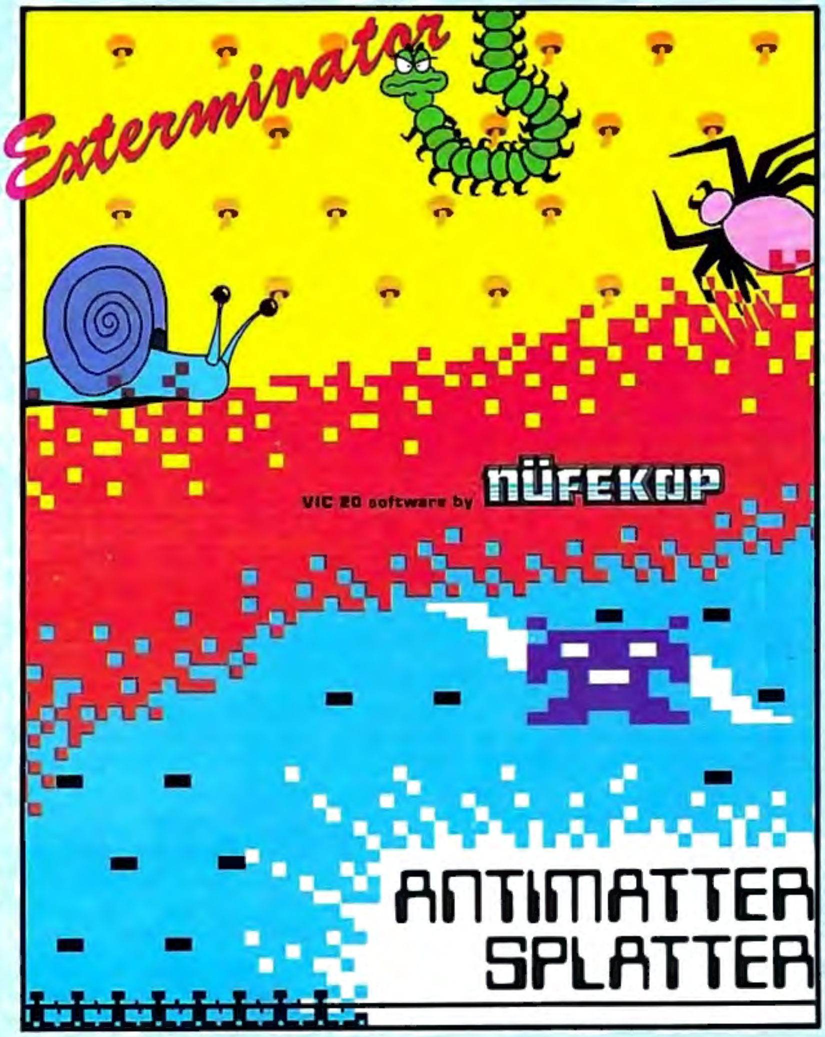
Exterminator by Ken Grant

First the bad news...this game is literally full of bugs. The good news? We guarantee hours of exciting entertainment trying to remove them. Some bugs you are likely to come up against are spiders, snails, fleas and centipedes in this rapidfire, 100% machine language, exceptional quality game. *Exterminator* runs in standard 5K VIC. **$24.95**

Antimatter Splatter! A more dastardly alien could scarcely be found than one who would wipe out an entire civilization by dropping antimatter anti-canisters, right? If your opinion of this alien troublemaker is the same as ours, probably your first thought was, get some matter! We say calm down! All is not lost. A mobile rapid splatter cannon capable of both breaking through his standard alien moving force fields and laying waste to the ever-increasing number of anti-canisters is even now hovering above us. If only our cannoneer hadn't called in sick...say, what are you doing today? *Anti-Matter Splatter* is 100% machine language and runs in standard 5K VIC. **$24.95**

3-D Man The exact maze from probably the most popular arcade game ever with perspective altered from overhead to eye level. The dots...the monsters...the power dots...the side exits. New on-screen radar. This game is *amazing*. *3-D Man* requires at least 3K memory expander but will run with any memory add-ons (8K, 16K, 24K, etc.) that we have come across. **$19.95**

Defender on Tri As pilot of the experimental Defender-style ship "Skyes Limited," you are the only hope for an advance party of scientists trapped in ancient alien sphere which suddenly (heat from collision course with sun presumably—G.E.) came to life. Four screens worth of unique defenses, on-off shields, fuel deposits, alien treasures, running timer, energy, score and very nice graphics display make this one that does not quickly wax old. *Defender on TRI* requires at least 3K memory expander, but will run with any memory add-on (8K, 16K, 24K, etc.) we have come across. **$19.95**



★ ★ SPECIAL ★ ★

Another NUFEKOP Big Deal to close this year out **RIGHT!** We at NUFEKOP want to celebrate by giving you a gift. Our catalog...FREE. No, seriously, if you order both *Antimatter Splatter* and *Exterminator*, or three of the lesser-priced games, you may pick **ONE FREE GAME** from any of the above (except *Exterminator*.) **Happy New Year!!**

Alien Panic Standard 5K VIC 20/combination stick & keyboard. This arcade-type game pits you against time and an alien on a six level construction sight with ladders and pitfalls, but *not to worry!* You have a shovel. **$12.95**

Krazy Kong The crazy gorilla has taken three fair maidens up to the top of the giant stairway. And you (the valiant hero) will attempt to rescue them at the risk of your own life. Your timing must be totally accurate as you jump the barrels that Kong is rolling down at you. **$12.95**

Rescue From Nufon *Adventure* **$12.95**
Collide *Crunch* **$12.95**
Vikman *Classic* **$12.95**
Search *Challenging* **$12.95**

SPECIAL OFFER! With any order of *Exterminator* and *Anti-Matter Splatter* we'll send you a *FREE* 17" X 22" poster like the one shown above. Also available separately for $3.50.

NÜFEKOP

P.O. Box 156, Shady Cove, Oregon 97539-0156
C.O.D. Orders...call (503) 878-2113
Mastercard and Visa cards accepted
Ask for our new FREE catalog!
Games will be on tape unless you request disk.

VIC is a trademark of Commodore Business Machines, Inc.

Easy Apple Disk Space Messages

Beirne L. Konarski, Kent, OH

As diskettes fill up, it is useful to know how much room is left on them. Two methods exist for finding this number. One is to get a calculator or pencil and paper and add the figures. This seems pointless, since the computer is supposed to do those tasks. The alternative is to run the FID program from the system master. This is a nuisance, though, because you often change diskettes.

Since many people incorporate a CATALOG command into their greeting program, this would be the most useful place for a sector-counting subroutine. The *DOS Manual* contains two sections which help to solve this problem. The first is the RWTS (Read or Write a Track and Sector) machine language program (p. 94). The second is the listing of sector allocation (p. 129). The RWTS subroutine can be used to read the sectors containing the catalog and tally the amount of sectors used.

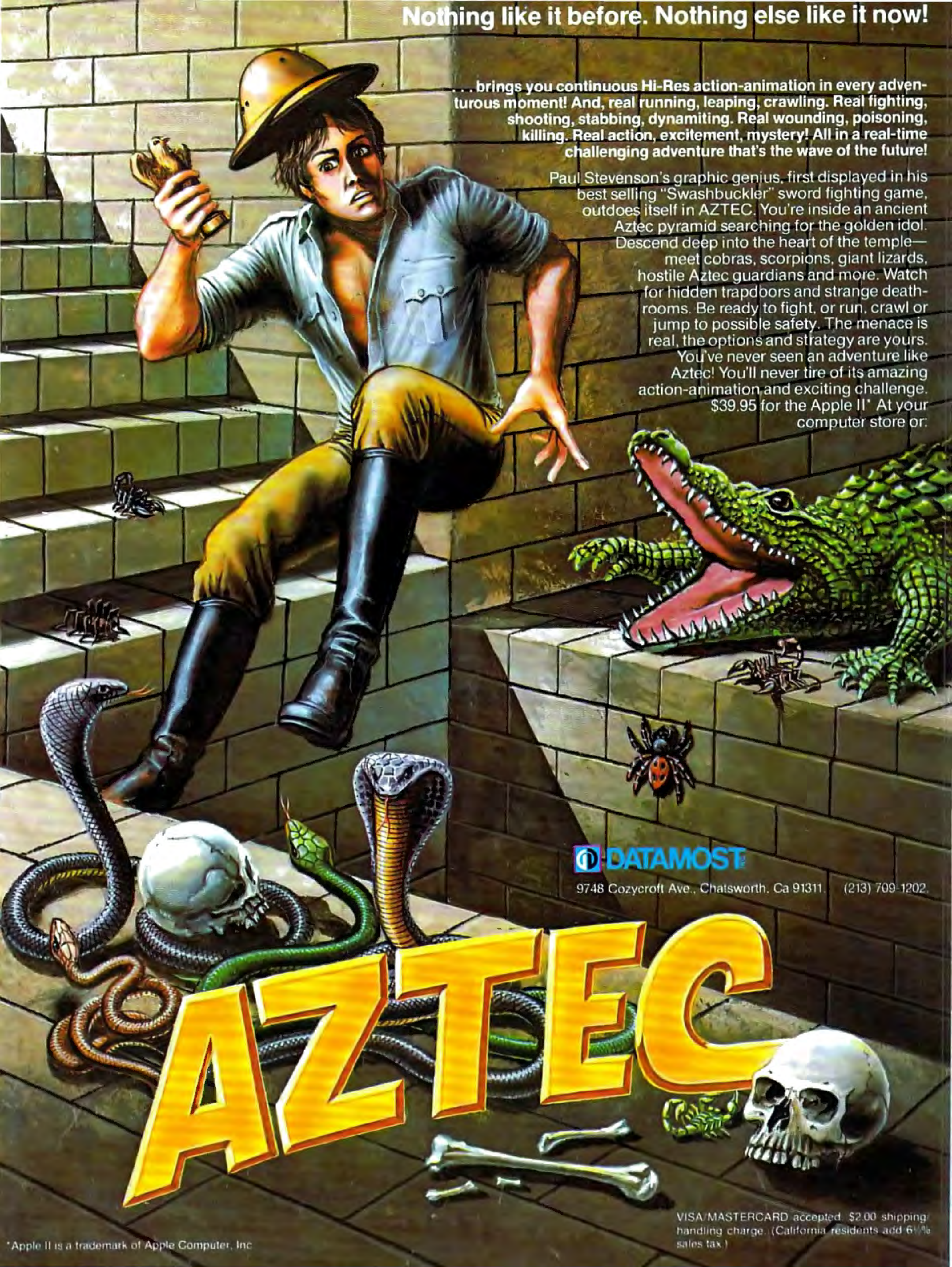
The diskette directory is located in track $11. It contains 15 sectors of catalog information, each holding the names and sizes of seven programs. The program reads one sector at a time beginning with sector $F and places it into the memory range beginning at $2000. The sector is then checked before the next one is read from the disk.

The seven program titles are checked for two things. The first is to see if the program is current. When a program is deleted, its catalog entry is not erased, but instead the first byte of the listing becomes $FF. If the program is current, then the bit containing the length is added to the running total. The Least Significant Byte of the total is stored in location $6074, or 24692. If there is a carry, it is plaçed in $6075. When all seven listings have been checked, the next sector is loaded, and the process continues until all fifteen sectors are read. The computer then returns to BASIC.

The rest of the BASIC program PEEKs the two locations containing the sum of the sectors used. The Most Significant Byte is multiplied by 256 and added to the LSB. This total is displayed and then subtracted from 496 to give the figure for the space remaining.

The program reads the DATA lines and enters the numbers beginning at $6000. It displays the CATALOG, then gives the results of the count. It can be substituted for your HELLO program, with your own personal touches like name and date placed before or after line 20, or after line 110.

```
10 D$ =  CHR$ (4)
20  PRINT D$;"CATALOG"
30  FOR K = 24576 TO 24694
40  READ A
50  POKE K,A
60  NEXT
70  CALL 24576
80 X =  PEEK (24692):Y =  PEEK (24693)
90 Z = X + 256 * Y
100  PRINT : PRINT "SECTORS USED: ";Z
110  PRINT "SECTORS REMAINING: ";496 - Z
120  END
130  DATA   169, 96, 160, 76, 32, 217, 3, 17
     3, 11, 32
140  DATA   201, 255, 240, 3, 32, 99, 96, 23
     8, 118, 96
150  DATA   173, 118, 96, 201, 7, 208, 28, 1
     69, 0, 141
160  DATA   118, 96, 169, 44, 141, 104, 96,
     169, 11, 141
170  DATA   8, 96, 173, 2, 32, 201, 0, 240,
     25, 141
180  DATA   81, 96, 76, 0, 96, 173, 104, 96,
      105, 35
190  DATA   141, 104, 96, 173, 8, 96, 105, 3
     5, 141, 8
200  DATA   96, 76, 7, 96, 96, 0, 1, 96, 1,
     0
210  DATA   17, 15, 94, 96, 0, 32, 0, 0, 1,
     0
220  DATA   0, 96, 1, 0, 0, 1, 239, 216, 0,
     24
230  DATA   173, 116, 96, 109, 44, 32, 141,
     116, 96, 144
240  DATA    3, 238, 117, 96, 96, 0, 0, 0, 0
```



Nothing like it before. Nothing else like it now!
. . . brings you continuous Hi-Res action-animation in every adventurous moment! And, real running, leaping, crawling. Real fighting, shooting, stabbing, dynamiting. Real wounding, poisoning, killing. Real action, excitement, mystery! All in a real-time challenging adventure that's the wave of the future!
Paul Stevenson's graphic genius, first displayed in his best selling "Swashbuckler" sword fighting game, outdoes itself in AZTEC. You're inside an ancient Aztec pyramid searching for the golden idol. Descend deep into the heart of the temple—meet cobras, scorpions, giant lizards, hostile Aztec guardians and more. Watch for hidden trapdoors and strange death-rooms. Be ready to fight, or run, crawl or jump to possible safety. The menace is real, the options and strategy are yours. You've never seen an adventure like Aztec! You'll never tire of its amazing action-animation and exciting challenge. $39.95 for the Apple II* At your computer store or:
DATAMOST
9748 Cozycroft Ave., Chatsworth, Ca 91311. (213) 709-1202.
AZTEC
VISA/MASTERCARD accepted. $2.00 shipping/handling charge. (California residents add 6½% sales tax.)
*Apple II is a trademark of Apple Computer, Inc

Juggler

Doug Ferguson, Elida, OH

For Atari and VIC (with or without expansion), this game will challenge anyone's dexterity. If you can score 50 with two balls or 500 with three, there is an extra surprise.

"Juggler" is a fast-action game I wrote in September 1981 when the idea of programming BASIC was very new to me. It has undergone countless revisions since then, but the main loop (lines 860-970) remains what it was the first night I struggled.

The purpose of Juggler is to use the joystick to control the arms of a cartoon juggler in his attempt to keep two or three balls in the air. The juggler's hands move in tandem and can be positioned to catch the three colored balls (inside, middle, and outside). Since this game is not for the timid, it requires a joystick because only game addicts know how to work them.

The balls are as random as I could make them and, contrary to the opinion of novice players, cannot land at the catching stage at the same time. The most important playing strategy is to make sure you hear the ball being caught before you move the hand toward another catch. The game can be slowed down merely by adding a few extra GOSUB 150 statements in the main loop, preferably at lines 860, 900, and 940.

The game will run on either the unexpanded or expanded VIC-20. I would rather not explain how it all works, mainly because it often follows inconsistent logic and layout. One technical footnote for those curious enough to investigate the program: instead of using zeros as values, I used a period (as in Y = .) because I read somewhere that this is faster and still zeros out the variable.

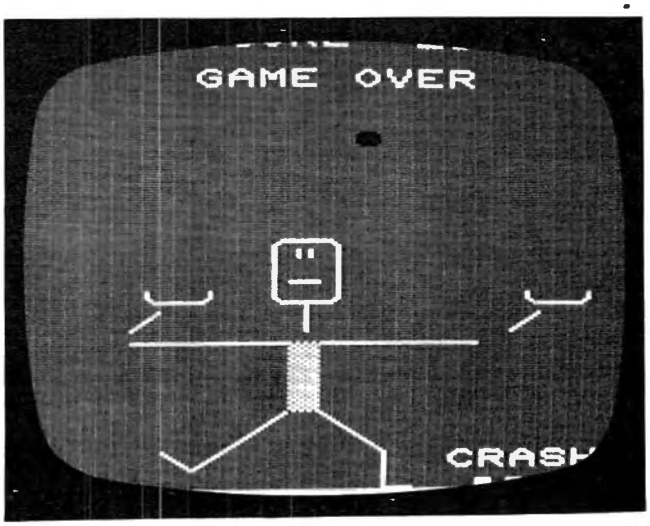
To start a game, push the joystick up or left for the three-ball or two-ball games, respectively. To repeat the same game, press the fire-button, or move the stick to change games. If you want to quit, hit Q.

Program 1: VIC-20 Version

```
100 POKE36879,75:X=RND(-TI):Y=2:GOT
    O670
110 POKEV,15:POKEV-2,N:FORT=1TO10:N
    EXT:POKEV-2,0
120 O=O+1+ABS(C)*9
130 PRINTTAB(7)"{WHT}SCORE "O"{HOME
    HOME}";:RETURN
140 DEFFNJ(X)=-((PEEK(37151)ANDX)=.
    ):RETURN
150 IFFNJ(16)ANDU=1THENGOSUB360
160 IFFNJ(16)ANDU=2THENGOSUB420
170 POKEG+2,127:Q=PEEK(G):POKEG+2,2
    55
180 IFQ=119ANDU=1THENGOSUB480
190 IFQ=119ANDU=.THENGOSUB420
200 RETURN
210 IFPEEK(H)<64THEN330
220 RETURN
230 IFPEEK(H+2)<64THEN340
240 RETURN
250 IFPEEK(H+4)<64THEN350
260 RETURN
270 IFPEEK(H+16)<64THEN330
280 RETURN
290 IFPEEK(H+14)<64THEN340
300 RETURN
310 IFPEEK(H+12)<64THEN350
320 RETURN
330 POKEJ%(K-A),32:D=K-1:GOSUB630:G
    OTO540
340 POKEU%(L-B),32:D=L:GOTO540
350 POKEG%(M-C),32:D=M+1:GOSUB640:G
    OTO540
360 POKEH+2,32:POKEH+3,32:POKEH-1,7
    4:POKEH,64:POKEH+1,75
370 POKEH+4,32:POKEH+5,32:POKEH+24,
    32:POKEH+23,77:POKEH+25,32
```

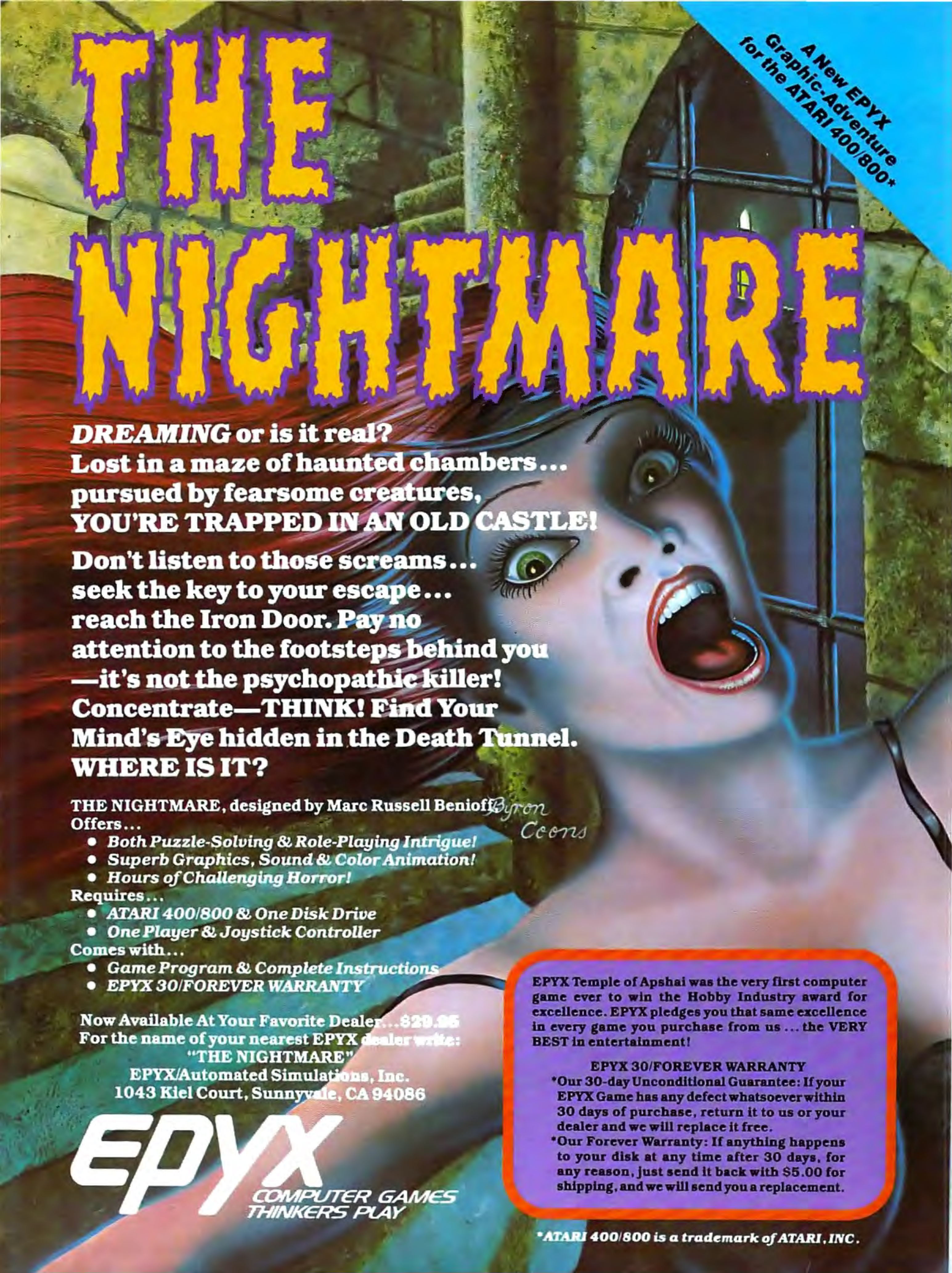

SUNDAY DRIVER!
NEW!
FOR THE ATARI 400/800

```
380 POKEH+14,32:POKEH+15,32:POKEH+1
    1,74:POKEH+12,64:POKEH+13,
    75
390 POKEH+16,32:POKEH+17,32:POKEH+3
    7,32:POKEH+35,77:POKEH+36,
    32
400 POKEH+187,77:POKEH+188,78:POKEH
    +209,32:POKEH+181,103:POKE
    H+203,122:POKEH+180,32:U=.
410 RETURN
420 POKEH+1,74:POKEH+2,64:POKEH+3,7
    5:POKEH,32:POKEH-1,32
430 POKEH+4,32:POKEH+5,32:POKEH+25,
    32:POKEH+23,32:POKEH+24,72
440 POKEH+13,74:POKEH+14,64:POKEH+1
    5,75:POKEH+12,32:POKEH+11,
    32
450 POKEH+16,32:POKEH+17,32:POKEH+3
    7,32:POKEH+35,32:POKEH+36,
    72
460 POKEH+181,103:POKEH+203,122:POK
    EH+187,101:POKEH+209,76:PO
    KEH+188,32:POKEH+180,32:U=
    1
470 RETURN
```



A fumbled ball crashes in "Juggler," VIC-20 version.

```
480 POKEH+3,74:POKEH+4,64:POKEH+5,7
    5:POKEH+2,32:POKEH+1,32
490 POKEH,32:POKEH-1,32:POKEH+25,78
    :POKEH+23,32:POKEH+24,32
500 POKEH+15,74:POKEH+16,64:POKEH+1
    7,75:POKEH+14,32:POKEH+13,
    32
510 POKEH+12,32:POKEH+11,32:POKEH+3
    7,78:POKEH+35,32:POKEH+36,
    32
520 POKEH+181,78:POKEH+180,77:POKEH
    +203,32:POKEH+187,101:POKE
    H+209,76:POKEH+188,32:U=2
530 RETURN
540 GOSUB650:PRINTTAB(D)"{21 DOWN}{
    WHT}CRASH{HOME}";:POKEV-1,
    N:POKE7954-SC,15:POKEH-36,
    34
550 FORT=15TO0STEP-1:POKEV,T:POKEV+
    1,PEEK(V+1)AND248ORT
560 FORW=1TO100:NEXT:NEXT:POKEV-1,.
    :POKE36879,75
570 PRINTTAB(7)"{YEL}{02 DOWN}GAME
    OVER{DOWN}":POKEH-14,64:GO
    TO590
580 PRINT"     {BLK}↑{YEL} 3 BALLS
    DOWN}":PRINT"     {BLK}←
    YEL} 2 BALLS{DOWN}":PRINT"
        PRESS {REV}Q{OFF} TO Q
    UIT":GOSUB140
590 GETA$:IFA$="Q"THENSYS65234
595 IFFNJ(32)ANDY<>2THENRESTORE:E=0
    :O=0:GOTO710
600 IFFNJ(16)THENCLR:Y=1:GOTO700
610 IFFNJ(4)THENCLR:GOTO700
620 GOTO590
630 FORT=38884TOT+6:POKET-CO,7:NEXT
    :FORT=38900TOT+6:POKET-CO,
    7:NEXT:RETURN
640 FORT=38884TOT+6:POKET-CO,6:NEXT
    :FORT=38899TOT+6:POKET-CO,
    6:NEXT:RETURN
650 FORT=8165+DTOT+2:POKET-SC,123:N
    EXT
660 RETURN
670 PRINT"{CLEAR}{DOWN}        {YEL}V
    IC JUGGLER{WHT}"
680 PRINT"{03 DOWN}{03 RIGHT}USE JO
    YSTICK ONLY{04 DOWN}"
690 PRINT"{YEL}{04 RIGHT}CHOOSE GAM
    E{DOWN}":GOTO580
700 V=36878:H=7968:G=37152:DIMJ%(18
    ),U%(16),G%(13)
710 PRINT"{CLEAR}";:IFPEEK(36869)=1
    92THENSC=3584:CO=512
720 H=7968-SC:GOSUB140
730 FORX=8015TO8025:POKEX-SC,64:NEX
    T:POKEX-6-SC,104
740 POKEH-37,93:POKEH-35,93:POKEH+7
    ,74:POKEH+9,75:POKEH+30,93
    :POKEH+118,102
750 POKEH-15,93:POKEH-13,93:POKEH-5
    9,85:POKEH-57,73
760 POKEH-36,114:POKEH-14,82:POKEH+
    8,114:POKEH-58,64:POKEH+74
    ,102
770 POKEH+139,78:POKEH+141,77:POKEH
    +160,78:POKEH+164,77:POKEH
    +96,102
780 GOSUB360
790 FORK=2TO16:READJ%(K):J%(K)=J%(K
    )-SC:POKEJ%(K)+30720+SC-CO
```


A New EPYX Graphic-Adventure for the ATARI 400/800*
THE NIGHTMARE
DREAMING or is it real?
Lost in a maze of haunted chambers...
pursued by fearsome creatures,
YOU'RE TRAPPED IN AN OLD CASTLE!
Don't listen to those screams...
seek the key to your escape...
reach the Iron Door. Pay no
attention to the footsteps behind you
—it's not the psychopathic killer!
Concentrate—THINK! Find Your
Mind's Eye hidden in the Death Tunnel.
WHERE IS IT?
THE NIGHTMARE, designed by Marc Russell Benioff
Byron Coons
Offers...
• Both Puzzle-Solving & Role-Playing Intrigue!
• Superb Graphics, Sound & Color Animation!
• Hours of Challenging Horror!
Requires...
• ATARI 400/800 & One Disk Drive
• One Player & Joystick Controller
Comes with...
• Game Program & Complete Instructions
• EPYX 30/FOREVER WARRANTY
Now Available At Your Favorite Dealer...$29.95
For the name of your nearest EPYX dealer write:
"THE NIGHTMARE"
EPYX/Automated Simulations, Inc.
1043 Kiel Court, Sunnyvale, CA 94086
EPYX
COMPUTER GAMES THINKERS PLAY
EPYX Temple of Apshai was the very first computer game ever to win the Hobby Industry award for excellence. EPYX pledges you that same excellence in every game you purchase from us... the VERY BEST in entertainment!
EPYX 30/FOREVER WARRANTY
*Our 30-day Unconditional Guarantee: If your EPYX Game has any defect whatsoever within 30 days of purchase, return it to us or your dealer and we will replace it free.
*Our Forever Warranty: If anything happens to your disk at any time after 30 days, for any reason, just send it back with $5.00 for shipping, and we will send you a replacement.
*ATARI 400/800 is a trademark of ATARI, INC.

```
     ,7:NEXT
800 FORK=2TO14:READU%(K):U%(K)=U%(K
     )-SC:NEXT
810 FORK=2TO12:READG%(K):G%(K)=G%(K
     )-SC:POKEG%(K)+3Ø72Ø+SC-CO
     ,6:NEXT
820 K=INT(RND(1)*8)+2:A=1
830 L=INT(RND(1)*7)+2:B=1
840 IFY=ØTHENM=INT(RND(1)*6)+2:C=1
850 IFO>49+45Ø*ABS(C)THENE=1:Y=.
860 GOSUB15Ø
870 N=22Ø:IFK=17THENGOSUB27Ø:A=-1:G
     OSUB11Ø:K=15-E
880 IFK=1THENGOSUB21Ø:A=1:GOSUB11Ø:
     K=3+E
890 POKEJ%(K-A),32:POKEJ%(K),81:K=K
     +A
900 GOSUB15Ø
910 IFL=15THENGOSUB29Ø:B=-1:GOSUB11
     Ø:L=13-E
920 IFL=1THENGOSUB23Ø:B=1:GOSUB11Ø:
     L=3+E
930 POKEU%(L-B),32:POKEU%(L),81:L=L
     +B
940 GOSUB15Ø
950 IFM=13THENGOSUB31Ø:C=-1:GOSUB11
     Ø:M=11-E
960 IFM=1THENGOSUB25Ø:C=1:GOSUB11Ø:
     M=3+E
970 POKEG%(M-C),32:POKEG%(M),81:M=M
     +C:IFE=.THEN85Ø
980 POKEJ%(K-3*A),32:POKEJ%(K-2*A),
     81
990 POKEU%(L-3*B),32:POKEU%(L-2*B),
     81
1010 POKEG%(M-3*C),32:POKEG%(M-2*C),
     81:GOTO86Ø
1020 DATA 7946,79Ø2,7858,7815,7772,7
     73Ø,771Ø,7712,7714,7738,77
     84,7829,7874,7918,7962
1030 DATA7948,79Ø4,786Ø,7817,7774,77
     54,7756,7758,7782,7827,787
     2,7916,796Ø
1040 DATA795Ø,79Ø6,7862,7819,7798,77
     78,78Ø2,7825,787Ø,7914,7958
```

Program 2: Atari Version

```
100 REM  JUGGLER - ATARI VERSION
110 GRAPHICS 18
120 GOSUB 660:GRAPHICS 18:POKE 756,CH
    SET/256:SETCOLOR 1,12,6:SETCOLOR
    3,4,8:SETCOLOR 2,0,10
130 SCR=PEEK(88)+256*PEEK(89)
140 POSITION 7,6:? #6;"  !"
150 POSITION 7,7:? #6;"<{F}{E}{D}>"
160 POSITION 7,8:? #6;"  {C}"
170 POSITION 7,9:? #6;" $ %"
180 BALLS=2:DIM BALL$(3):BALL$="G{E}
    {G}":POSITION 6,0:? #6;M$
190 GOSUB 320:POSITION 1,1:? #6;"ball
    S: ";BALLS
200 POSITION 10,1:? #6;"{8 SPACES}":I
    F PRACTICE THEN POSITION 10,1:? #
    6;"PRACTICE"
210 K=PEEK(53279):IF K=7 THEN 210
220 IF PEEK(53279)=K THEN 220
230 IF K=5 THEN BALLS=5-BALLS
240 IF K=3 THEN PRACTICE=1-PRACTICE
```

Atari Notes

Use your joystick to move the Juggler's arms to any of three positions (far left, center, or far right) to catch and deflect the balls. You can select a game with either two balls or three balls by pressing SELECT when the game is RUN. If you press OPTION, you can play a "practice" game. In a practice game, you cannot make any points, but you can't lose, either. It's a good way to learn how to play without becoming frustrated.

A successful catch is greeted with a "bleep," but a miss gets you a raspberry. If you're not playing a practice game, it's all over when you miss a ball. The rest come tumbling down!

A note on strategy: let go of the joystick after each move. This will return the Juggler to the center position, where it is easiest to quickly react.

```
250 IF K<>6 THEN 190
260 POSITION 1,1:? #6;"{9 SPACES}"
270 GOSUB 1190
280 REM  MAIN LOOP
290 GOSUB 320:GOSUB 320
300 GOSUB 440:GOSUB 320
310 GOTO 290
320 REM  MOVE ARMS
330 IF PEEK(53279)=6 THEN RUN
340 POS=2-(1-PTRIG(0))+(1-PTRIG(1))
350 ON POS GOSUB 370,390,410
360 RETURN
370 POSITION 6,7:? #6;"<{F}{D}{E}>  "
    :POKE 77,0
380 POSITION 8,9:? #6;"$ +":RETURN
390 POSITION 6,7:? #6;" <{F}{E}{D}> "
400 POSITION 8,9:? #6;"$ %":RETURN
410 POSITION 6,7:? #6;"  <{E}{D}{F}>"
    :POKE 77,0
420 POSITION 8,9:? #6;"* %":RETURN
430 REM  MOVE BALL
440 INDEX=(INDEX+1)*(INDEX<BALLS-1)
450 BPOS=BPOS(INDEX)
460 WHICH=WHICH(INDEX):BDIR=BDIR(INDE
    X):BIN=BIN(INDEX)
470 POKE BPOS,0
480 BPOS=BPOS+PB(WHICH,BIN)*BDIR
490 POKE BPOS,ASC(BALL$(INDEX+1))
500 BIN=BIN+BDIR:IF BIN=0 OR BIN>PB(W
    HICH,0) THEN 530
```

For the Atari 400/800 Home Computer

You are Sentinel I, the latest in highly maneuverable strike aircraft, and you have a mission, to protect the metropolis, but the alien attack will stop at nothing to destroy your very last lines of defense. Your senses are tuned for battle and the attack begins.

Aliens will block your path, destroy your ship, deplete your fuel and sacrifice their lives to stop your mission. You must destroy the aliens with your rapid fire lasers before they home in and destroy you. There is no escape — you must destroy them all for they will stop at nothing.

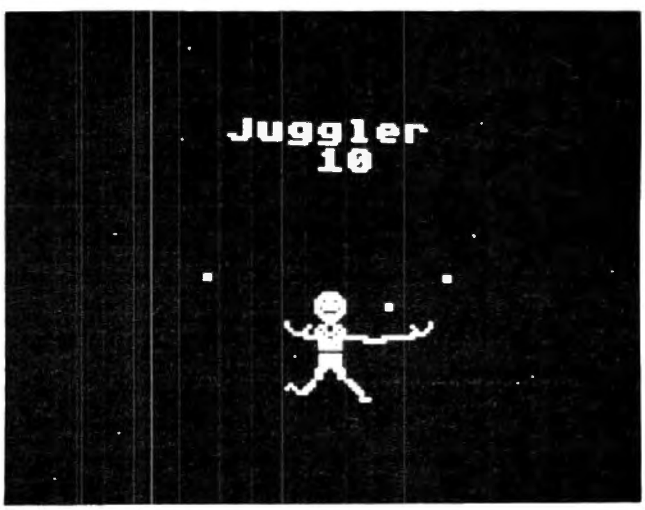
- 100% machine language
- 1 or 2 player option
- joy stick controls
- lateral scrolling screen
- superb graphics
- extensive color
- finest sound utilization
- available in 16K tape $29.95 U.S. funds
- 24K disc $34.95 U.S. funds
- call your local dealer for more information

INHOME SOFTWARE

Inhome Software Incorporated 2485 Dunwin Drive, Mississauga, Ontario L5L 1T1. (416) 828-0775.

Atari is a registered trade mark of Atari Inc. Made in Canada

```
510 BIN(INDEX)=BIN
520 BPOS(INDEX)=BPOS:RETURN
530 REM END OF TRAVEL...MISSED?
540 P=PEEK(BPOS+20):IF P<>28 AND P<>3
    0 THEN 580
550 FOR W=14 TO 0 STEP -2:SOUND 0,50,
    10,W:SOUND 1,60,10,W:NEXT W
560 IF  NOT PRACTICE THEN SCORE=SCORE
    +1+9*(BALLS=3):POSITION 10-LEN(ST
    R$(SCORE))/2,1:? #6;SCORE;" "
570 POKE BPOS,0:BDIR(INDEX)=-BDIR(IND
    EX):GOSUB 1260:RETURN
580 SOUND 0,100,12,8:FOR W=1 TO 50:NE
    XT W:SOUND 0,0,0,0:IF PRACTICE TH
    EN 570
590 FOR I=1 TO 10:FOR J=0 TO BALLS-1
600 POKE BPOS(J),ASC(BALL$(J+1))
610 BPOS(J)=BPOS(J)+20-BDIR(J):SOUND
    0,I*BALLS+J,12,10-I:NEXT J:NEXT I
620 POSITION 9,6:? #6;"[illegible]"
630 POSITION 5,0:? #6;"press START"
640 IF PEEK(53279)<>6 THEN 640
650 RUN
660 REM INITIALIZATION
```



Reaching with an outstretched arm in the Atari version of "Juggler."

```
670 GRAPHICS 2+16:DIM M$(10)
680 M$="JUggLEr"
690 FOR I=1 TO LEN(M$):A=ASC(M$(I)):P
    OKE 712,INT(A/16)*16+14
700 FOR J=1 TO 11:COLOR 32:PLOT I+J-1
    ,J-1:COLOR A:PLOT I+J,J:SOUND 0,J
    +I*10,10,8:NEXT J:COLOR 32:PLOT I
    +J-1,J-1
710 FOR J=10 TO 6 STEP -1:COLOR 32:PL
    OT I+J/2+3.5,J+1:COLOR A:PLOT I+3
    +J/2,J:SOUND 0,J+I*10,10,8:NEXT J
720 NEXT I
730 FOR I=0 TO 240 STEP 10:POKE 712,0
740 A=PEEK(708):POKE 708,PEEK(709):PO
    KE 709,PEEK(710):POKE 710,PEEK(71
    1):POKE 711,A
750 POKE 711,PEEK(53770):POKE 53279,0
760 SOUND 0,I,10,4:SOUND 1,I+10,10,4:
    NEXT I
770 SOUND 0,0,0,0:SOUND 1,0,0,0
780 GRAPHICS 2+16:POSITION 7,6:? #6;M
    $
790 CHSET=(PEEK(106)-8)*256:FOR I=0 T
    O 7:POKE CHSET+I,0:NEXT I
800 RESTORE 840:IF PEEK(CHSET+8)=60 T
    HEN 1030
810 FOR I=128 TO 207:POKE CHSET+I,PEE
    K(57344+I):SOUND 0,I,10,8:POKE 53
    274,I:NEXT I
820 FOR I=1 TO 14:READ A,B:FOR J=0 TO
     7:POKE CHSET+A*8+J,PEEK(57344+B*
    8+J):SOUND 0,A,10,8:SOUND 1,B,10,
    8
830 POKE 53274,A:NEXT J:NEXT I:SOUND
    0,0,0,0:SOUND 1,0,0,0
840 DATA 34,34,33,97,44,108,51,115,26
    ,26,42,42,53,117,39,103,37,101,50
    ,114,48,48,35,99,52,116,41,105
850 READ A:IF A=-1 THEN SOUND 0,0,0,0
    :GOTO 1030
860 FOR J=0 TO 7:READ B:POKE CHSET+A*
    8+J,B:SOUND 0,B,10,8:POKE 53274,B
    :NEXT J
870 GOTO 850
880 DATA 1,60,126,219,255,189,195,126
    ,60
890 DATA 2,24,102,165,219,231,126,122
    ,126
900 DATA 3,0,126,126,126,126,231,195,
    195
910 DATA 4,3,6,12,24,16,112,0,0
920 DATA 5,192,96,48,24,8,14,0,0
930 DATA 6,60,90,219,255,231,231,126,
    60
940 DATA 7,0,0,0,24,24,0,0,0
950 DATA 8,0,0,0,0,63,224,0,0
960 DATA 9,0,0,0,0,252,7,0,0
970 DATA 10,1,67,166,28,0,0,0,0
980 DATA 11,128,194,101,56,0,0,0,0
990 DATA 28,128,130,68,60,7,0,0,0
1000 DATA 30,1,33,18,60,224,0,0,0
1010 DATA 127,16,24,28,30,30,28,24,16
1020 DATA -1
1030 REM MORE INITALIZATION
1040 REM Read in parabolas
1050 DIM PB(4,15):RESTORE 1060
1060 DATA 0,-19,-20,-21,-21,-1,19,19,
     20,20,0,-99
1070 DATA 0,-20,-21,-21,-21,-1,-1,19,
     20,21,21,0,-99
1080 DATA 0,-19,-20,-21,-21,-1,-1,19,
     19,20,21,0,-99
1090 DATA 0,-19,-20,-20,-21,-1,19,19,
     19,20,0,-99
1100 DATA 0,-20,-21,-21,19,19,20,0,-9
     9
1110 FOR I=0 TO 4
1120 FOR J=1 TO 15
1130 READ A:IF A<>-99 THEN PB(I,J)=A:
     NEXT J
1140 PB(I,0)=J-1
1150 NEXT I
1160 DIM BPOS(2),BPEEK(2),WHICH(2)
1170 DIM BDIR(2),BIN(2):RETURN
1180 REM INITIALIZE BALLS
1190 FOR I=0 TO 2:WHICH(I)=-1:NEXT I
1200 FOR I=0 TO BALLS-1
1210 BDIR(I)=-1+2*(RND(1)>0.5)
1220 INDEX=I:GOSUB 1260
1230 NEXT I
1240 RETURN
1250 REM INITIALIZE ONE BALL
1260 WHICH=INT(5*RND(0)):WHICH(INDEX)
     =-1
1270 BPOS(INDEX)=SCR+130+INT(3*RND(0)
     )-4*(BDIR(INDEX)=-1)
1280 FOR J=0 TO BALLS-1
1290 IF WHICH(J)=WHICH THEN J=BALLS:N
     EXT J:GOTO 1260
1300 NEXT J:WHICH(INDEX)=WHICH
1310 BIN(INDEX)=1+(PB(WHICH,0)-1)*(BD
     IR(INDEX)=-1)
1320 RETURN
```

KMMM Pascal for PET/CBM $85

A subset of standard Pascal with extensions.
- Machine language Pascal Source Editor with cursor oriented window mode
- Machine Language P-Code Compiler
- P-Code to machine language translator for optimized object code
- Run-time package
- Floating point capability
- User manual and sample programs

Requires 32K Please specify configuration.

EARL for PET (disk file based) $65

Editor, Assembler, Relocator, Linker

Generates relocatable object code using MOS Technology mnemonics. Disk file input (can edit files larger than memory). Links ,multiple object programs as one memory load. Listing output to screen or printer. Enhanced editor operates in both command mode and cursor oriented "window" mode.

RAM/ROM for PET/CBM

4K or 8K bytes of soft ROM with optional battery backup.

RAM/ROM is compatible with any large keyboard machine. Plugs into one of the ROM sockets above screen memory to give you switch selected write protectable RAM.

Use RAM/ROM as a software development tool to store data or machine code beyond the normal BASIC range. Use RAM/ROM TO LOAD A ROM image where you have possible conflicts with more than one ROM requiring the same socket. Possible applications include machine language sort (such as SUPERSORT), universal wedge, Extramon, etc.

RAM/ROM – 4K	$75
RAM/ROM – 8K	90
Battery Backup Option	20

SUBSORT by James Strasma $35

Subsort is an excellent general purpose machine language sort routine for PET/CBM computers. Sorts both one and two dimensioned arrays at lightning speed in either ascending or descending order. Other fields can be subsorted when a match is found, and fields need not be in any special order. Sort arrays may be specified by name, and fields are random length. Allows sorting by bit to provide 8 categories per byte. The routine works with all PET BASICs, adjusts to any memory size, and can co-exist with other programs in high memory.

SuperGraphics 2.0

NEW Version with TURTLE GRAPHICS

SuperGraphics, by John Fluharty, provides a 4k machine language extension which adds 35 full featured commands to Commodore BASIC to allow fast and easy plotting and manipulation of graphics on the PET/CBM video display, as well as SOUND Commands. Animations which previously were too slow or impossible without machine language subroutines now can be programmed directly in BASIC. Move blocks (or rocketships, etc.), or entire areas of the screen with a single, easy to use BASIC command Scroll any portion of the screen up, down, left, or right. Turn on or off any of the 4000 (8000 on 8032) screen pixels with a single BASIC command. In high resolution mode, draw vertical, horizontal, and diagonal lines. Draw a box, fill a box, and move it around on the screen with easy to use BASIC commands Plot curves using either rectangular or polar co-ordinates (great for Algebra, Geometry and Trig classes.)

The SOUND commands allow you to initiate a note or series of notes (or even several songs) from BASIC, and then play them in the background mode without interfering with your BASIC program. This allows your program to run at full speed with simultaneous graphics and music.

Seven new TURTLE commands open up a whole new dimension in graphics. Place the TURTLE anywhere on the screen, set his DIRECTION, turn him LEFT or RIGHT, move him FORWARD, raise or lower his plotting pen, even flip the pen over to erase. Turtle commands use angles measured in degrees, not radians, so even elementary school children can create fantastic graphic displays.

Specify machine model (and size), ROM type (BASIC 3 or 4)

SuperGraphics in ROM $45

Volume discounts available on ROM version for schools.

FLEX-FILE NEW VERSION II

for PET/CBM Computers

FLEX-FILE is a set of flexible, friendly programs to allow you to set up and maintain a data base. Includes versatile Report Writer and Mail Label routines, and documentation for programmers to use Data Base routines as part of other programs.

RANDOM ACCESS DATA BASE

Record size limit is 256 characters. The number of records per disk is limited only by record size and free space on the disk. File maintenance lets you step forward or backward through a file, add, delete, or change a record, go to a numbered record, or find a record by specified field (or partial field). Field lengths may vary to allow maximum information packing. Both subtotals and sorting may be nested up to 5 fields deep. Any field may be specified as a key. Sequential file input and output, as well as file output in WordPro and PaperMate format is supported. Record size, fields per record, and order of fields may be changed easily.

MAILING LABELS

Typical mail records may be packed 3000 per disk on 8050 (1400 on 4040). Labels may be printed any number wide, and may begin in any column position. There is no limit on the number or order of fields on a label, and complete record selection via type code or field condition is supported.

REPORT WRITER

Flexible printing format, including field placement, decimal justification and rounding. Define any column as a series of math or trig functions performed on other columns, and pass results such as running total from row to row. Totals, nested subtotals, and averages supported. Complete record selection, including field within range, pattern match, and logical functions can be specified.

FLEX-FILE II by Michael Riley $110

Please specify equipment configuration when ordering.

DISK I.C.U. $40

Intensive Care Unit by L.C. Cargile

COMPLETE DISK RECOVERY SYSTEM FOR CBM DRIVES
- edit disk blocks with ease
- duplicate disks, skipping over bad blocks
- complete diagnostic facilities
- un-scratch scratched files
- check and correct scrambled files
- recover improperly closed files
- extensive treatment of relative files
- optional output to IEEE488 printer
- comprehensive user manual (an excellent tutorial on disk operation and theory).

Furnished on copy-protected disk with manual.
Backup disk available, $10 additional.

PROGRAM YOUR OWN EPROMS $75

Branding Iron EPROM Programmer for PET/CBM software for all ROM versions. Includes all hardware and software to program or copy 2716 and 2532 EPROMs.

PORTMAKER DUAL RS232 SERIAL PORT $63

Two ports with full bipolar RS232 buffering. Baud rates from 300 to 4800. For PET/CBM, AIM, SYM

CBM Software

TCL Pascal Version 1.6	135
Petspeed BASIC Compiler	225
Integer BASIC Compiler	110
CMAR Record Handler	110
UCSD Pascal (without board)	135
Wordcraft 80	300
BPI Accounting Modules	300
Professional Tax Prep Sys.	600
Intelligent Terminal Emulator	25
ASERT Data Base	375
Personal Tax Calc	55
Dow Jones Portfolio Mgmt.	110
Assembler Development	80
Legal Time Accounting	445

FORTH for PET

BY L. C. Cargile and Michael Riley $50

Features include:

full FIG FORTH model.
all FORTH 79 STANDARD extensions.
structured 6502 Assembler with nested decision making macros.
full screen editing (same as when programming in BASIC).
auto repeat key.
sample programs.
standard size screens (16 lines by 64 characters).
150 screens per diskette on 4040, 480 screens on 8050.
ability to read and write BASIC sequential files.
introductory manual.
reference manual.

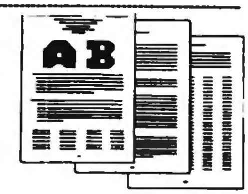
Runs on any 16K or 32K PET/CBM (including 8032) with ROM 3 or 4, and CBM disk drive. Please specify configuration when ordering.

Metacompiler for FORTH $30

simple metacompiler for creating compacted object code which can be executed independently (without the FORTH system).

PaperMate 60 COMMAND WORD PROCESSOR

by Michael Riley



Paper-Mate is a full-featured word processor for CBM/PET by Michael Riley. Paper-Mate incorporates 60 commands to give you full screen editing with graphics for all 16K or 32K machines (including 8032), all printers, and disk or tape drives. Many additional features are available (including most capabilities of Professional Software's WordPro 3).

For writing text, Paper-Mate has a definable keyboard so you can use with either Business or Graphics machines. Shift lock on letters only, or use keyboard shift lock. All keys repeat.

Paper-Mate text editing includes floating cursor, scroll up or down, page forward or back, and repeating insert and delete keys. Text block handling includes transfer, delete, append, save, load, and insert.

All formatting commands are imbedded in text for complete control. Commands include margin control and release, column adjust, 9 tab settings, variable line spacing, justify text, center text, and auto print form letter (variable block). Files can be linked so that one command prints an entire manuscript. Auto page, page headers, page numbers, pause at end of page, and hyphenation pauses are included.

Unlike most word processors, CBM graphics as well as text can be used. Paper-Mate can send any ASCII code over any secondary address to any printer.

Paper-Mate functions with all CBM/PET machines with at least 16K, with any type of printer, and with either cassette or disk.

To order Paper-Mate, please specify machine and ROM type.

Paper-Mate (disk or tape) for PET, CBM, VIC, C64 **$40**

SM-KIT for PET/CBM $40

Enhanced ROM based utilities for BASIC 4. Includes both programming aids and disk handling commands

BASIC INTERPRETER for CBM 8096 $200

A full interpreter implementation to automatically take advantage of the extra memory available with 8096.

PEDISK II Systems from cgrs Microtech available.
FILEX IBM 3741/2 Data Exchange Software available.

JINSAM Data Base Management System for CBM.

Comprehensive version available for most configurations.

COPY-WRITER Word Processor for PET/CBM. $159

Works like expensive word processors, plus has added features like double column printing and shorthand generator.

CASH MANAGEMENT SYSTEM $45

Easy to use disk system. Keeps track of cash disbursements, cash receipts, cash transfers, expenses for up to 50 categories.

252 Bethlehem Pike
Colmar, PA 18915

215-822-7727

A B Computers

WRITE FOR CATALOG

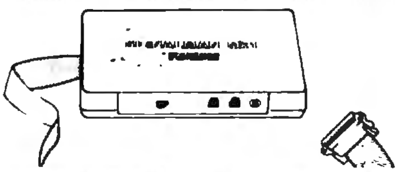
Add $1.25 per order for shipping. We pay balance of UPS surface charges on all prepaid orders. Prices listed are on cash discount basis. Regular prices slightly higher. Prices subject to change.

WE STOCK EAGLE COMPUTERS **EAGLE COMPUTER**

USI Video Monitors—Green or AMBER 20 MHz hi-res. Dealer and OEM inquiries invited

SPECIALS on INTREGATED CIRCUITS

6502	7.45	10/ 6.95	50/ 6.55	100/ 6.15
6502A/6512A	8.40	10/ 7.95	50/ 7.35	100/ 6.90
6520 PIA	5.15	10/ 4.90	50/ 4.45	100/ 4.15
6522 VIA	6.45	10/ 6.10	50/ 5.75	100/ 5.45
6532	7.90	10/ 7.40	50/ 7.00	100/ 6.60
2114-L200		2.45	25/ 2.30	100/ 2.15
2716 EPROM		4.90	5/ 4.50	10/ 4.00
2532 EPROM		8.90	5/ 8.45	10/ 7.90
6116 2K×8 CMOS RAM		8.90	5/ 8.45	10/ 7.90
4116 RAM				8 for 14
Zero Insertion Force 24 pin Socket (Scanbe)				2.00



Anchor Automation Signalman Modems

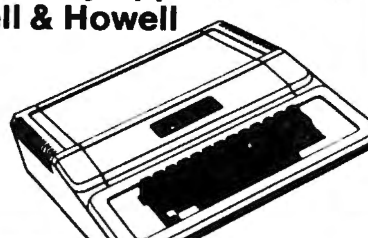
FREE SOURCE MEMBERSHIP WITH SIGNALMAN

All Signalman Modems are Direct Connect, and include cables to connect to your computer and to the telephone. Signalman Modems provide the best price-performance values, and start at less than $100. **Dealer and OEM Inquiries invited**

Mark I RS232
Mark II for Atari 850
Mark IV for CBM/PET with software
Mark V for Osborne (software available)
Mark VI for IBM Personal Computer
Mark VII Auto Dial/Auto Answer
Mark VIII Bell 212 Auto Dial/Answer

DC HAYES Smartmodem	**229**
RS232 MODEM — Accoustic	**119**
RS232 MODEM — CCITT frequencies	**175**

We carry Apple II+ from Bell & Howell



16K RAM Card for Apple	**65**
Apple LOGO	150
Video Recorder Interface	545
Super Serial Card	149
Thunderclock Plus	119
Z80 Softcard and CP/M	295
Parallel Printer Interface/Cable	80
Grappler Interface	139
TG Products Joystick for Apple	**48**
TG Paddles	32
DC Hayes Micromodem II	299
Videx 80 Column Card	259
fullFORTH+ for Apple (fig-Forth)	85
Silentype Printer and Card	310
Graphics Tablet and Card	645
Apple PASCAL Language	195
Apple FORTRAN	160
We stock EDUWARE Software	
GENIS I Courseware Development System	145
Unicom Grade Reporting or School Inventory	250
Executive Briefing System with fonts	225
Apple Dumpling (Microtek) Printer Interface	**115**
Apple Dumpling with 16K Buffer	**160**
PIE Writer Word Processor	**120**

commodore

See us for Personal, Business, and Educational requirements. Educational Discounts available.

PETSCAN $245 base price

Allows you to connect up to 35 CBM/PET Computers to shared disk drives and printers. Completely transparent to the user. Perfect for schools or multiple word processing configurations. Base configuration supports 2 computers. Additional computer hookups $100 each.

Commodore COMMUNICATES!

COMPACK $129

Intelligent Terminal Package includes:
ACIA hardware based interface; DB25 Cable and STCP Software with remote telemetry, transfer to/from disk, printer output, XON-XOFF control, user program control, and status line.

VE-2 IEEE to Parallel Interface 119

Includes case, power supply, full 8-bit transmission, and switch selectable character conversion to ASCII.

VIC 20	189	VIC Sargon II Chess	32
VIC Printer	335	VIC GORF	32
VIC 3K RAM	32	Meteor Run (UMI)	39
VIC 8K RAM	53	VIC Radar Ratrace	24
VIC 16K RAM	99	Amok (UMI)	20
VIC Disk Drive	395	Snakman	15
VIC Pinball	32	Rubik's Cube	13
VIC Omega Race	32	Programmers Reference	15
Spiders of Mars (UMI)	39	Renaissance (UMI)	39
VIC Draw Poker	24	VIC Superslot	23

VICTORY Software for VIC

Street Sweepers	12	Maze in 3-D	12
Night Rider	11	Cosmic Debris	12
Treasures of Bat Cave	12	Grave Robbers Advent.	11
Games Pack I	12	Games Pack II	12
Victory Casino	8	Adventure Park I	12
Adventure Pack II	12	Trek	11

TNW 488/103 with DAA	450
Compute!'s First Book of PET/CBM	11
POWER ROM Utilities for PET/CBM	78
WordPro 3+ - 32K CBM, disk, printer	195
WordPro 4+ - 8032, disk, printer	300
SPELLMASTER spelling checker for WordPro	**170**
COPY-WRITER Professional Word Processor	159
VISICALC for PET, ATARI, or Apple	190
PETRAX PET to Epson Graphics Software	**35**
SM-KIT enhanced PET/CBM ROM Utilities	**40**
Programmers Toolkit - PET ROM Utilities	35
PET Spacemaker II ROM Switch	36
2 Meter PET to IEEE or IEEE to IEEE Cable	40
Dust Cover for PET, CBM, 4040, or 8050	8
VIC or C64 Parallel Printer Interface	85
CmC IEEE-RS232 Printer Interface — PET	120
SADI Intelligent IEEE-RS232 or parallel	235
Library of PET Subroutines	12
Programming the PET/CBM (Compute!) — R. West	**20**
Compute! First Book of VIC	11
Whole PET Catalog (Midnight Gazette)	**8**
Color Chart Video Board for PET	125
PET Fun and Games (Cursor)	11

REVERSAL (Spracklen) Apple or Atari	25
SARGON II — Apple or TRS-80	26
Apple II User's Guide (Osborne)	12
Introduction to Pascal (Sybex)	13
Pascal Handbook (Sybex)	16
Musical Applications of Micros (Chamberlin)	20
Starting FORTH	14
Discover FORTH	12
User Guide to the Unix System	13
6502 Assembly Language Subroutines	11
PET Fun and Games	9
KAMIKAZE (Hayden Software-Apple)	28

DISK SPECIALS

SCOTCH (3M) 5"	10/ 2.30	50/ 2.10	100/ 2.05
SCOTCH (3M) 8"	10/ 2.45	50/ 2.20	100/ 2.15

We stock VERBATIM DISKS
Write for Dealer and OEM prices.

BASF 5" or 8"	10/ 2.00	20/ 1.95	100/ 1.85
Wabash 5"	10/ 1.80	50/ 1.75	100/ 1.70
Wabash 8"	10/ 2.25	50/ 2.20	100/ 2.10

We stock MAXELL DISKS
Write for dealer and OEM prices.

Disk Storage Pages 10 for $5 Hub Rings 50 for $6
Disk Library Cases 8"—3.00 5"—2.25
Head Cleaning Kits 11

CASSETTES—AGFA PE-611 PREMIUM
High output, low noise, 5 screw housings.

C-10	10/ .61	50/ .58	100/ .50
C-30	10/ .85	50/ .82	100/ .70

SPECIALS

Timex/Sinclair Computer	95
Zenith ZVM-121 Green Phosphor Monitor	109
INTEX Talker Text to Speech System	**265**
Epson, Okidata, Prowriter printers available	
Brother Daisy Wheel Printer	880
STARWRITER Daisy Wheel Printer F10	1445
We Stock AMDEK Monitors	
Watanabe Intelligent Plotter 995 6-pen	1295
Staticide anti-static spray	6
dBASE II	445

ALL BOOK and SOFTWARE PRICES DISCOUNTED
A P Products 15% OFF

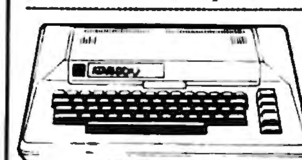
Synertek SYM-1 Microcomputer	**SALE 189**
KTM-2/80 Synertek Video and Keyboard	349
KTM-3/80 Synertek Tubeless Terminal	385

Alspa Computer, Inc.

The price-performance leader. Includes Z80A, 1 or 2 full 8" drives (double density, double sided), 3 serial and 1 parallel port, and winchester port. Prices start at less than $2000. DEALER and OEM inquiries invited.

ZENITH data systems

Z90-80 64K	1995
Z90-82 64K, 1 double dens. drive	2245
Z37 1.3 Megabyte Dual Drive	1355
Z19 Video Terminal (VT-52 compatible)	695
ZT-1 Intelligent Communications Terminal	**550**
Z100 16-bit/8-bit System	**CALL**



ATARI SPECIALS

800 Computer	649	Microsoft BASIC	72
400—16K	269	**MISSILE COMMAND**	**29**
810 Disk Drive	440	**ASTEROIDS**	**29**
825 Printer	625	**STAR RAIDERS**	**34**
850 Interface	170	Space Invaders	29
Inside Atari DOS	18	Music Composer	35
Joysticks or Paddles	19	Caverns of Mars	33
16K RAM (Microtek)	69	**PAC-MAN**	**36**
32K RAM (Microtek)	99	**CENTIPEDE**	**36**
Pilot	65	First Book of Atari	11
Super Breakout	29	**Anchor Modem—Atari**	**85**
APX Software	Call	Other Atari products	Call

252 Bethlehem Pike
Colmar, PA 18915 215-822-7727 **A B Computers**

WRITE FOR CATALOG
Add $1.25 per order for shipping. We pay balance of UPS surface charges on all prepaid orders. Prices listed are on cash discount basis. Regular prices slightly higher. Prices subject to change.

SOUND On The Sinclair/Timex

Arthur B. Hunkins
School of Music, UNCG, Greensboro, NC

Sound on the Sinclair/Timex? Beethoven symphonies, no; simple melodies, yes! All you need is one of the programs below, and a high-gain amplifier/speaker connected to the computer's mike output. (Radio Shack's battery-operated mini- or telephone-amplifier/speakers at $10-12 work fine.)

The software secret is a short Z-80 machine language (ML) routine buried in a REM statement at the beginning of a BASIC program. It is important that the routine be the FIRST program statement; if it isn't, all the USR and POKE addresses that follow must be changed. We'll make our opening REMark statement number 10, and leave it!

After typing 10 REM, press the following sequence of CHARACTERS (ignore commas, periods, and spaces): NEXT, A, /, I, =, :, COPY, INKEY$, PEEK, COPY, (, RETURN, INKEY$, <=, RETURN, (, RETURN, H, 4, LET, 9, 4, GOTO, TAN. Remember, these are *Characters – single keystrokes*. Spaces will appear in the display, but you don't type any in.

Some of the characters are FUNCTIONS; to register them, you first hit the function key. Others are KEYWORDS, a bit trickier. First hit THEN, which causes the K cursor to appear; then press the keyword; finally go back and delete THEN. Presto, a keyword in a REM statement! When you are finished entering characters, be sure to hit ENTER.

Now we'll add a second statement: 30 LET A=USR 16514. Attach the amplifier to the mike output, turn up the volume, and we're ready for a test. (You might want to save the program first – machine language crashes make you start over.) Be sure you are in FAST mode; sound doesn't work in SLOW. (In your own applications, you may switch back and forth from SLOW to FAST whenever you wish.)

RUN the short program. If all is well, you should hear a slightly low B above middle C for approximately one second. The screen display goes berserk during the note, showing horizontal black streaks similar to a LOAD. When the sound is finished, the previous display returns.

Note that the computer hum, which is quite audible except during the note itself, is caused by the screen display. When the screen is "off," including black during computer calculations, there is no hum. This program produces one note per subroutine call – a square wave, limited in range to pitches from B above middle C on up. (More sophisticated routines that extend the range into the bass register, and permit a variety of tone colors, are discussed later.)

Frequency And Duration

Let's construct a simple BASIC program to play our choice and length of note:

```
10 REM (as above)
12 LET B=255
14 LET D=5
17 POKE 16520,B
22 POKE 16518,D*1000/B
30 LET A=USR 16514
```

B is the frequency value, POKEd into location 16520; D is the duration value, which – converted to be constant for varying frequencies – is placed in 16518. B must be between 1 and 255, and D must be greater than 0. For higher frequencies, it is possible to ask for too long a duration; one may receive a report code of B in statement 22 here (an attempt to POKE a number larger than 255 into location 16518).

For longer tones, adjust tempo location 16516, which normally contains value 24. POKEing a smaller value speeds up the tempo, while a larger value makes everything last longer. Note that these three memory locations – for frequency, duration, and tempo – are the same for all programs in this article. So is the USR subroutine call address.

The single-byte frequency values for pitches from the B above middle C up through two octaves are given in the table. These are the values with zero as the high byte (ignore the zero and use only the low byte). For this particular program, *add three* to each value listed; thus a low B (the first pitch with high byte = 0) should be 250 instead of 247. Experiment with different frequency and duration values.

Now let's explore a routine that permits a full range of pitches, down to two octaves below middle C or lower, if you wish. Here we require a two-byte frequency value. The machine language is more extensive. Again begin with 10 REM, adding the following sequence of character strokes: NEXT, A, /, I, =, upper left quarter square graphic, upper left quarter square graphic, COPY, VAL, PEEK, COPY, lower left quarter square graphic, left half square graphic, C, upper right quarter square graphic, (, RETURN, £, $, C, upper half square graphic, $, /, RUN, AT, <=, RETURN, lower left quarter square graphic, left half square graphic, C, upper right quarter square graphic, (, RETURN, £, $, C, upper half square graphic, $, /, RUN, H, 4, AND, 9, 4, STR$, TAN.

Whew! Again conclude by hitting ENTER, and double or triple-check the entry. Be sure to identify the correct graphics characters, preceding and following them by pressing the GRAPHICS key to obtain or cancel the G cursor. To get the graphics character rather than a reverse field letter, you must use the shift key.

This routine may be quick-checked also by adding the statement: 30 LET A = USR 16514. This time you should hear a slightly low B *below* middle C, for about two seconds. The complete BASIC program is listed below.

```
10 REM (as above)
12 LET B=1
13 LET C=255
14 LET D=10
15 LET X=B*256+C
17 POKE 16520,B
18 POKE 16521,C
22 POKE 16518,D*1000/X
23 IF PEEK 16518=0 THEN POKE 16518,1
30 LET A=USR 16514
```

Here B and C are the two-byte frequency values (high-low order). X sums the two values for purposes of calculating the duration to be POKEd into 16518. Statement 23 may seem problematic. It is needed because very low notes (high B and C values) may cause 16518 to contain zero, which will produce a *very long* tone in combination with a small D value. Statement 23 protects against this possibility. To obtain accurate rhythms on low short tones, speed up the tempo (POKE a value less than 24 into 16516) so that you can work with larger values of D. Again, try different frequencies and durations, using the frequency table at the end of the article. B is the high byte, C, the low. Use the values straight out of the table (do *not* add 3).

Varying Tone

Next we have a modification of the above ML routine that permits variation in tone color. It is more complex and difficult to use, but permits any width of pulse (rectangular) wave by changing a single variable. While requiring considerable additional overhead, only one more variable is specified (pulse width). Frequency values are the same as in the previous program. It will be easiest to enter this routine as a modification of the previous one.

Start, as usual, with 10 REM then: NEXT, A, /, I, =, upper left quarter square graphic, upper left quarter square graphic, COPY, PEEK, COPY, lower left quarter square graphic, left half square graphic, C, upper right quarter square graphic, (, RETURN, £, $, C, upper half square graphic, $, /, RUN, upper left quarter square graphic, upper left quarter square graphic, COPY, <=, RETURN, lower left quarter square graphic, left half square graphic, C, upper right quarter square graphic, (, RETURN, £, $, C, upper half square graphic, $, /, RUN, H, 4, OR, 9, 4, USR, TAN. Conclude with ENTER. Adding the statement 30 LET A = USR 16514 should result in the same tone as previously, since a square wave (50 percent pulse width) is specified.

As the BASIC overhead for this routine is fairly extensive, it may not run on the 1K ZX-81. All other programs should.

```
11 LET P=50
12 LET B=1
13 LET C=255
14 LET D=10
15 LET X=B*256+C
16 LET Y=P*.02*X
17 POKE 16520,INT (Y/256)
18 POKE 16521,Y-PEEK 16520*256
19 LET Y=(100-P)*.02*X
20 POKE 16538, INT (Y/256)
21 POKE 16539,Y-PEEK 16538*256
22 POKE 16518,D*1000/X
23 IF PEEK 16518=0 THEN POKE 16518,1
30 LET A=USR 16514
```

P is the new variable, representing pulse width expressed as a percentage (greater than zero and less than 100). The other statements are needed to calculate, from a single two-byte frequency number, the two sets of timing loop values for the top and bottom portions of the pulse wave. Suffice it to say that the two pairs of loop values go into locations 16520-16521 and 16538-16539, and that if the two pairs are identical, you get a square wave; otherwise, a variable-width pulse results. Incidentally, these variable pulse widths may be monitored on the screen, where the thicker streaks of white represent greater positive pulse widths at the same frequency. Note, too, that spacing of the streaks is proportional to frequency.

A Short Melody

Finally, let's return to our first sound routine – the one with one-byte frequency values, square waves, and high pitches only – and attempt a short melodic phrase. To do this, we define a series of frequency and duration variables in arrays, inserting them in order during a FOR/NEXT loop that calls the notes one at a time.

Observe that this program is not designed to perform entire compositions (though, given enough memory, it could). I will review two commercial programs in an upcoming issue of

COMPUTE! which allow you to code or perform, then play back, extended melodies.

```
10 REM (as above, first sound routine)
12 DIM B(6)
13 DIM D(6)
14 LET B(1)=157
15 LET D(1)=3
16 LET B(2)=186
17 LET D(2)=1
18 LET B(3)=235
19 LET D(3)=4
20 LET B(4)=186
21 LET D(4)=4
22 LET B(5)=157
23 LET D(5)=4
24 LET B(6)=117
25 LET D(6)=8
26 FOR I=1 TO 6
27 POKE 16520,B(I)
28 POKE 16518,D(I)*1000/B(I)
35 LET A=USR 16514
40 NEXT I
```

After SAVEing the program to prevent possible catastrophe, RUN it. Do you recognize the tune? If the tempo is too slow, you can always POKE the tempo location, 16516 – insert the statement, 11 POKE 16516,15 (or POKE any other number below 24). Experiment with different speeds between 1 and 255.

You may have noticed that there is no provision for rests. Rests are a bit awkward. Perhaps you might want to work out something inside the play loop that checked for a B (frequency) array variable of zero, and converted the D (duration) array variable into an index for a "do nothing" FOR/NEXT loop. Yes, it sounds complicated. Perhaps the following suggestion is some improvement.

Add two statements to the program above: 30 IF B(I)=255 THEN POKE 16528,255, and 32 IF B(I)<>255 THEN POKE 16528,254. Now, if you code a FREQUENCY value of 255, you'll get a rest of the specified duration rather than a pitch. Make sure to reserve the "pitch" of 255 for a rest. Or you may choose, and reserve, any other value greater than zero to 255 for this purpose. Unfortunately, zero won't work. After inserting this code, try substituting 255 for one of the B array values in the melody. You should get a note hole.

Comparable additions to the BASIC code for the other sound routines are also possible. I suggest reserving the value of 255 for the *lower* frequency byte. In the routine that deals with square waves throughout the frequency range, add the statements IF C(I)=255 THEN POKE 16540,255 and IF C(I)<>255 THEN POKE 16540,254 during the play loop. Note the change in memory location. The C array represents the lower frequency byte. For the routine with variable pulse width, use the same two statements, but POKE 16541 instead of 16540.

What follows is a list of delay loop (frequency) POKE values for equally tempered pitches through five octaves around middle C. The first pitch is two octaves below middle C, the last, three octaves above middle C. Be sure to *add three* to these values when using the one-byte, simple sound routine. This routine handles only pitches with a high byte of zero, which is omitted.

Pitch Values

PITCH	HIGH BYTE	LOW BYTE
C	7	105
C#/DF	6	250
D	6	145
D#/EF	6	35
E	5	218
F	5	130
F#/GF	5	50
G	4	228
G#/AF	4	155
A	4	90
A#/BF	4	23
B	3	225
c	3	167
c#/dF	3	114
d	3	65
d#/eF	3	17
e	2	230
f	2	188
f#/gF	2	149
g	2	110
g#/aF	2	75
a	2	42
a#/bF	2	11
b	1	238
c1 (middle c)	1	210
c#1/dF1	1	184
d1	1	159
d#1/eF1	1	136
e1	1	113
f1	1	92
f#1/gF1	1	72
g1	1	53
g#1/aF1	1	36
a1	1	19
a#1/bF1	1	3
b1	0	247
c2	0	232
c#2/dF2	0	219
d2	0	207
d#2/eF2	0	195
e2	0	183
f2	0	173
f#2/gF2	0	163
g2	0	154
g#2/aF2	0	145
a2	0	136
a#2/bF2	0	128
b2	0	121
c3	0	114
c#3/dF3	0	107
d3	0	101
d#3/eF3	0	95
e3	0	89
f3	0	84
f#3/gF3	0	79
g3	0	74
g#3/aF3	0	70
a3	0	66
a#3/bF3	0	62
b3	0	58
c4	0	55

THUNDERBIRD

Dave Sanders, Garland, UT

For TRS-80 Color Computer, Atari and Unexpanded VIC, this game should prove a challenge for all age levels. So far, none of the players who've tried it have been able to get past the second level. But if you do, the game will keep getting harder.

"Thunderbird" will demand your undivided attention and all of the memory the unexpanded VIC-20 has to offer. The object of Thunderbird is to score as high as possible. The high score will be kept from game to game. The scoring is as follows: 200 points for taking out a tree, 50 points for taking out a saucer, 75 points for deflecting off either wing of the Thunderbird, 25 points for deflecting off the main body of the Thunderbird, and 1000 points for breaking out the bottom of the playing field. When the satellite drops into a well, 125 points are subtracted from the score.

You score these points by keeping the satellite in the playing field. The satellite can break out the top and the bottom of the screen. When it breaks out the bottom, you score 1000 points, and a new and more difficult playing field is set up for you. If the satellite breaks out the top of the field, your game is half over. You can lose only two satellites out the top. You prevent the satellite from breaking out the top by deflecting it back into the field with the Thunderbird. The Thunderbird is moved across the top of the field with the cursor control keys.

The display on the right side of the screen tells you if you are playing the first or second satellite. When the satellite drops into a well, the Thunderbird lasers down from one to three multicolored saucers to further hinder the satellite from breaking out the bottom. You will notice that the Thunderbird deflects the satellite one way off its main body and a different way off its wings. You have to keep the Thunderbird moving across the screen in conjunction with the direction the satellite is moving, or you will not play for very long.

With a little practice, the first breakout is not too hard. The second breakout will not be out of reach either, but no one in our neighborhood has broken out the third time. Just in case you are a whiz though, the game will continue to get harder.

Crunching It Into The VIC

Because of the length of the program, I had to use a technique known as "crunching." You can pack more instructions – and power – into your BASIC programs by making each program as short as possible.

Crunching programs lets you squeeze the maximum possible number of instructions into your program. It also helps you reduce the size of programs which might not otherwise run in a given size.

A list of keyword abbreviations is given in the Appendix D in the *Personal Computing Guide* that you received with your VIC-20. This is helpful when you program because you can actually crowd more information on each line by using these abbreviations. In this program it is mandatory to use this technique on many of the lines when you type them in. The most frequently used abbreviation in this program is PO (P shifted-O) which is the BASIC abbreviation for the POKE command. However, if you LIST a program that has abbreviations, the VIC-20 will automatically print out the listing with the full-length keywords.

If any program line exceeds 88 characters (four lines on the screen) with the keywords unabbreviated, and you want to change it, you will have to re-enter that line with the abbreviations before saving the program.

SAVEing a program incorporates the keywords without inflating any lines because BASIC keywords are tokenized by the VIC-20. Usually, abbreviations are added after a program is written and do not have to be LISTed any more before SAVEing.

REM statements are helpful in reminding yourself – or showing other programmers – what a particular section of a program is doing. However, when the program is completed and ready to use, you probably will not need those REM statements any more; you can save quite a bit of space by removing them. If you plan to revise or study the program structure in the future, it is a good idea to keep a copy on file with the REM statements intact.

computer mail order west

HOME COMPUTERS

ATARI

400

16K........$199

32K (Non Atari Ram) $274

48K (Non Atari Ram) $359

800

48K........$499

New low price effective January 1, 1983

410 Recorder	$74.00
810 Disk Drive	$429.00
822 Printer	$269.00
825 Printer	$589.00
830 Modem	$159.00
820 Printer	$259.00
850 Interface	$169.00
CX40 Joysticks (pair)	$18.00
CX853 Atari 16K Ram	$77.95

Microtek 16K Ram	$74.95
Axlon Ramdisk (128K)	$429.95
Intec 48K Board	$159.00
Intek 32K Board	$74.00
One Year Extended Warranty	$70.00
CX481 Entertainer Package	$69.00
CX482 Educator Package	$130.00
CX 483 Programmer Package	$54.00
CX 484 Communicator Package	$344.00

PERCOM

DISK DRIVES FOR ATARI COMPUTERS

S1 Single Drive	$549.00
A1 Add-On Drive	$339.00
S2 Dual Drive	$879.00
Single Side Dual Head	$679.00
Dual Drive Dual Head	$1046.00

μ-SCI

MICRO-SCI

DISK DRIVES FOR FRANKLIN & APPLE

A2	$299.00
A40	$369.00
A70	$499.00
C2 Controller	$79.00
C47 Controller	$89.00

SOFTWARE FOR ATARI

ATARI

Pac Man	$33	Missile Command	$29
Centipede	$33	Star Raiders	$35
Caverns of Mars	$32	Galaxian	$33
Asteroids	$29	Defender	$33

ON-LINE

Jawbreaker	$27	Mission Asteroid	$22
Softporn	$27	Mouskattack	$31
Wizard & Princess	$29	Frogger	$31
The Next Step	$34	Cross Fire (ROM)	$36

SYNAPSE

File Manager 800	$89	Shamus	$26
Chicken	$26	Protector	$26
Dodge Racer	$26	Nautilus	$26
Synassembler	$30	Slime	$26
Page 6	$19	Disk Manager	$24

DATASOFT

Pacific Highway	$25	Graphic Generator	$13
Canyon Climber	$25	Micro Painter	$25
Tumble Bugs	$25	Text Wizard	$79
Shooting Arcade	$25	Spell Wizard	$64
Clowns & Balloons	$25	Bishop's Square	$25
Graphic Master	$30	Sands of Egypt	$25

EPYX

Crush, Crumble	$24	Morloc's Tower	$16
Undead Crypt	$24	Rescue at Rigel	$24
Curse of Ra	$16	Ricochet	$16
Datestones	$16	Star Warrior	$29
Invasion Orion	$19	Temple Apshai	$29
Arthur's Heir	$24	Upper Reaches	$16

APX

Text Formatter	$18.50	Holy Grail	$24
Family Budget	$18.50	Player Piano	$18.50
Eastern Front	$24	Keyboard Piano	$18.50
Family Cash	$18.50	Number Blast	$13
Jukebox	$13	Frogmaster	$18.50
Downhill	$18.50	747 Land Simul	$18.50
Outlaw	$18.50	Word Processor	$40

CBS

K-razy Shoot Out	$32	K-razy Antics	$32
K-razy Kritters	$32	K-star Patrol	$32

STICK STAND $6.99

VISICORP

For Apple, IBM & Franklin

Visidex	$189.00
Visifile	$189.00
Visiplot	$159.00
Visiterm	$89.00
Visitrend/Plot	$229.00
VisiSchedule	$229.00
Desktop Plan	$189.00
VISICALC for Apple II plus, Atari, CBM & IBM	179.00

CONTINENTAL

The Home Acountant(Apple/Franklin)	$59.00
The Home Accountant (IBM)	$119.00
1st Class Mail (Apple/Franklin)	$59.00

SIRIUS

Free Fall	$24	Space Eggs	$24
Beer Run	$24	Sneakers	$24
Snake Byte	$24	Bandits	$28

BRODERBUND

Apple Panic	$23	Arcade Machine	$34
David's Magic	$27	Choplifter	$27
Star Blazer	$25	Serpitine	$27

INFOCOM

Deadline	$35	Zork I	$29
Star Cross	$29	Zork II or III	$29

MPC

Bubdisk (128K Ram)	$719.00

PRINTERS

Smith Corona

TP 1	$599.00

C.ITOH (TEC)

Starwriter (F10-40CPS)	$1399.00
Printmaster (F10-55CPS)	$1749.00
Prowriter 80 Col (P)	$499.00
Prowriter 80 Col (S)	$629.00
Prowriter 2 (132 Col)	$799.00

OKIDATA

82A	$429.00
83A	$659.00
84P	$1079.00
84S	$1199.00

IDS

MicroPrism	$649.00
132 (fully configured)	$1599.00
80 (fully configured)	$1399.00

Call for other configurations.

DAISYWRITER

Letter Quality	$1049.00

DIABLO

620	$1179.00
630	$1849.00

FLOPPY DISKS

MAXELL		VERBATUM	
MD I (Box of 10)	$32	5¼" SS DD	$26
MD II (Box of 10)	$44	5¼" DS DD	$36
MFD I (8")	$40	ELEPHANT	
MFD II (8" DD)	$50	5¼" SS DD	$19.99

MONITORS

AMDEK

300G	$169.00
Color I	$339.00
Color II	$699.00
Color III	$429.00

BMC

12" Green	$85.00
13" Color 1400	$279.00
13" Color 1401 (Mid Res)	$369.00

ZENITH

ZVM121	$99.00

SHARP

Sharp 13" Color TV	$275.00

PANASONIC

TR-120 MIP (High Res. Green)	$159.00
CT-160 Dual Mode Color	$299.00

MODEMS

HAYES

Smart	$239.00
Smart 1200 (1200 Baud)	$549.00
Chronograph	$199.00
Micromodem II (with Term)	$309.00
Microdem 100	$309.00

NOVATION

Cat	$144.00
D-Cat	$159.00
Auto Cat	$219.00
212 Auto Cat	$589.00
Apple Cat II	$279.00
212 Apple Cat II	$609.00

ANCHOR

Mark I (RS-232)	$79.00
Mark II (Atari)	$79.00
Mark III (TI-99)	$109.00
Mark IV (CBM/PET)	$125.00
Mark V (OSBORNE)	$95.00
Mark VI (IBM-PC)	$179.00
Mark VII (Auto Answer Call)	$119.00
TRS-80 Color Computer	$99.00
9 Volt Power Supply	$9.00

800-648-3311

west

IN NV. CALL (702) 588-5654

P.O.BOX 6689 STATELINE, NV. 89449

west

INTERNATIONAL ORDERS: All shipments outside continental United States must be pre-paid by certified check only! Include 3% (minimun $3.00) shipping and handling. **EDUCATIONAL DISCOUNTS:** Additional discounts are available from both Computer Mail Order locations to qualified Educational Institutions.

FRANKLIN ACE 1000 SYSTEM

ACE 1000
ACE 10 with Controller Card
ACE Writer Word Processor
CALL..
FOR SYSTEM PRICE.

IBM® PC

NEC 3550 Printer $2099

PERCOM DRIVES

5¼" 160K Disk Drive	$329
5¼" 320K Disk Drive	$449

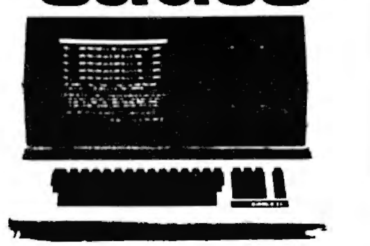
AMDEK

310A Amber Monitor	$179
Amdisk(3¼") Drive	$729
DXY Plotter	$759

SOFTWARE

I.U.S. Easywriter II	$249
I.U.S. Easyspeller	$129
Peachtree Peach Pak (GL/AP/AR)	$419
MPC Bubdisk	call

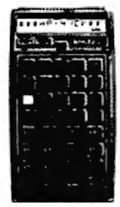
EAGLE



64K RAM
780 KB Disk Storage
Word Processing, Ultracalc CP/M
C-Basic Software
Smith Corona TP1
Letter Quality Printer
$2995.00
Retail Value $4895.00

RANA DISK DRIVES
Call for price and availability on the new Rana Disk Drives for the Apple and Franklin Computer Systems

HEWLETT PACKARD HP 41CV CALCULATOR $209



HP 41C	$149.00
HP 10C	$69.00
HP 11C	$79.00
HP 12C	$114.00
HP 15C	$109.00
NEW 16C	$114.00

HEWLETT PACKARD HP•85 $1969

HP-125	$1969.00
HP-85 16K Memory Module	$169.00
5¼"Dual Master Disk Drive	$1799.00
Hard Disk w/Floppy	$4349.00
Hard Disk	$3549.00
"Sweet Lips" Printer	$1199.00
80 Column Printer	$649.00

commodore

8032	$999.00
CBM 64	CALL
4032	$749.00
8096 Upgrade Kit	$369.00
Super Pet	$1599.00
2031	$369.00
8250 Double Sided Disk Drive	$1699.00
D9060 5 Megabyte Hard Disk	$2399.00
D9060 7.5 Megabyte Hard Disk	$2699.00
8050	$1299.00
4040	$969.00
8300 (Letter Quality)	$1549.00
8023	$599.00
4022	$399.00
New Z-Ram, Adds CP/M and 64K Ram	$549.00
The Manager	$209.00
Magis	CALL
Word Pro 5 plus	$319.00
Word Pro 4 plus	$299.00
Word Pro 3 plus	$199.00
The Administrator	$379.00
InfoPro Plus	$219.00
Power	$79.00
VIC 20 Dust Cover	$6.99
CBM 8032 Dust Cover	$14.99
CBM 8050/4040 Dust Cover	$10.99

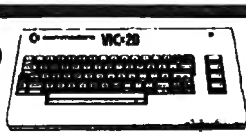
SHARP PC-1500 POCKET COMPUTER $209

CE 150 Printer, Plotter and Cassette Interface Unit	$172.00
CE152 Cassette Recorder	$69.00
CE 155 8K Ram Expansion Module	$94.00

TELEVIDEO TERMINALS

910	$579.00
912C	$699.00
920C	.749.00
925C	$749.00
950	$950.00
800A	$1319.00
802	$2649.00
802H	$4695.00
806	$5495.00
816	$9495.00

VIC 20 $179



VIC 1530 Commodore Datassette	$69.00
VIC 1540 Disk Drive	$339.00
VIC 1541 (64 Disk Drive)	CALL
VIC 1525 Graphic Printer	$339.00
VIC 1210 3K Memory Expander	$32.00
VIC 1110 8K Memory Expander	$53.00
VIC 1111 16K Expansion	$94.00
VIC 1011 RS232C Terminal Interface	$43.00
VIC 1112 VIC IEEE-488 Interface	$86.00
VIC 1211 VIC 20 Super Expander	$53.00
VIC Mother Board	$99.00

Timex Sinclair 1000 $89

16K Memory Module $44 95

Vu-Calc	$17 95	The Organizer	$14 95
Super Math	$12 95	The Budgeter	$13 95
Check Book Manager	$13 95	Stock Option	$14 95

Loan & Mortgage Amortizer $12 95

NEC

COMPUTERS

8001A	$729.00
8031	$729.00
8012	$549.00

PRINTERS

8023	$499.00
7710/7730	$2399.00
3510/3530	$1599.00

MONITORS

JB-1260	$129.00
JB-1201	$159.00
JC-1201	$319.00
JC-1203	$729.00

800-233-8950

east IN PA. CALL (717) 327-9575 **east**
477 E. THIRD ST., WILLIAMSPORT, PA. 17701

In-stock items shipped same day you call. No risk, no deposit on C.O.D. orders. Pre-paid orders receive free shipping within the continental United States with no waiting period for certified checks or money orders. Add 3% (minimum $3.00) shipping and handling on all C.O.D. and Credit Card orders. NV and PA residents add sales tax. All items subject to availability and price change. **NOTE:** We stock manufacturer's and third-party software for most all computers on the market! Call today for our new catalogue

computer mail order east

Following is a list of REM statements that would have been in my program if there had been room.

Program Line No.	Description
4-17	Routine to set up playing field.
25-26	Subroutine for printing score.
50-54	Routine for making game more difficult.
55-59	Routine for displaying instructions and a short game.
65-80	Routine for firing laser and starting satellite back at a random location.
85	Routine for the graphics when satellite takes out saucers.
90-91	Routine for sounds and points on breakout.
95-98	Routine for sounds and colors on losing satellite out the top of the field.
100-103	Routine for moving Thunderbird across screen.
104-118	This section moves satellite and has all the PEEKs for the other routines in the program.

One of the easiest ways to reduce the size of your program is to eliminate all the spaces. Although programmers often include spaces in sample programs to provide clarity, you actually do not need any spaces in your program and will save memory if you eliminate them.

Instead of PRINTing several cursor commands to position a character on the screen, it is often more economical to use the TAB and SPC instructions to position words or characters on the screen. Well, that's enough on "crunching." You can find these and many other useful instructions in the VIC-20 *Programmers Reference Guide* VM110 published by Commodore.

On line 68 a couple of saucers are lasered down by the Thunderbird. The screen code POKEd for the saucers is the same as all the other saucers, but they certainly look different. This effect is achieved by POKEing a 9 into the color code location for these saucers. POKEing a color location with a number above eight will switch that location into multicolor mode. You can get some very interesting shapes and colors by using multicolor.

In lines four through seven, the (Q) is the ball graphic, and the (W) is the circle.

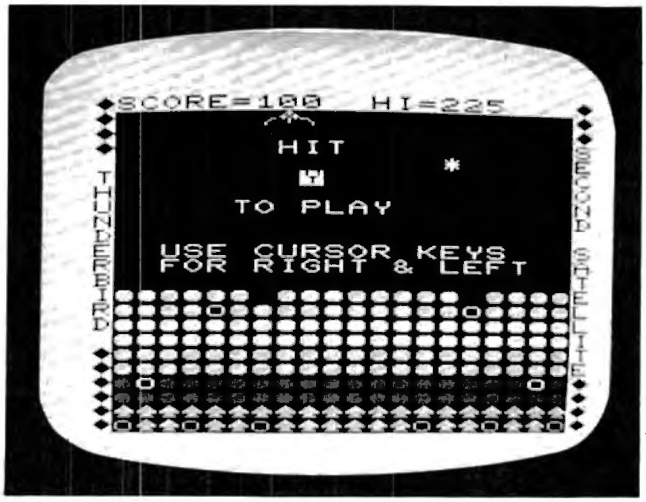
If you do not want to punch the program in, I will be happy to make a copy (VIC only) for you on tape. Send a cassette with a self-addressed, stamped (requires 40 cents postage) envelope, and a check for $3 to:

Dave Sanders
P.O. Box 533
Garland, UT 84312

Program 1: VIC-20 Version

```
2 VD=36874:F=125:OX=30742:OF=30720:P1=1:L=1:
   SC=0:HI=0:K=1:M=7703:RS=1:VA=VD+2:C=V
   A+3
4 PRINT"{CLEAR}":POKEC,105:FORR=1TO17:PRINT:
   NEXT:PRINT"{YEL}QQQQQQQQQQQQQQQQQQQQQ
   QQQQQQ{WHT}W{YEL}QQQQQQQQQQ{WHT}W{YEL
   YEL}QQQQQ"
5 PRINT"{WHT}{UP}QQQQQQQQQQQQQQQQQQQQQQQQQQQ
   QQQQQQQQQQQQQQQQQ"
6 PRINT"{CYN}{UP}QQQQQQQQQQQQQQQQQQQQQQQQQQQ
   QQQQQQQQQQQQQQQQQ"
7 PRINT"{PUR}{UP}QQ{WHT}W{PUR}QQQQQQQQQQQQQQ
   QQ{WHT}W{PUR}QQQQQQQQQQQQQQQQQQQQQQQQ
   "
8 FORR=8142TO8186:POKER,65:NEXT:J=87:G=81:FO
   RR=38423TO38442:POKER,7:NEXT:POKE3687
   8,15
9 FORR=38863TO38882:POKER,5:NEXT:FORR=38885T
   O38904:POKER,5:NEXT:POKE8165,J:POKE81
   68,J:POKE8171,J
10 POKE8178,J:POKE8181,J:POKE8184,J:FORR=7987
   TO8141STEP22:POKER+OF,1:NEXT:FORR=768
   0TO8164STEP22
11 READA:POKER,A:NEXT:FORR=7701TO8185STEP22:R
   EADA:POKER,A:NEXT:FORR=7681TO7700:REA
   DA:POKER,A:NEXT
12 FORR=7966TO8120STEP22:POKER+OF,1:NEXT:REST
   ORE:IFP1=>2THEN50
13 PRINTSPC(6)"{REV}{WHT}{22 UP}"SC:POKE7686,
   189:PRINTSPC(14)"{REV}{WHT}{UP}"HI:PO
   KE7694,189
14 X=1:Y=1:DX=1:DY=1:POKEM+1,85:POKEM+2,88:PO
   KEM+3,73:IFRS=1THENRS=RS+1:GOTO55
15 IFTT=500THENTT=1:X=12:L=1:SC=0:PRINT"{HOME
   HOME}{07 RIGHT}{REV}      ":GOTO104
16 IFL<>2THEN104
17 POKE7767,147:POKE7789,133:POKE7811,131:POK
   E7833,143:POKE7855,142:POKE7877,132:G
   OTO104
25 PRINTSPC(6)"{UP}{REV}"SC:POKE7686,189:IFSC
   >HITHENHI=SC:PRINTSPC(14)"{REV}{UP}"H
   I:POKE7694,189
26 RETURN
50 FORR=7945TO7964:POKER,G:NEXT:POKE8059,J:PO
   KE8070,J:FORR=7945TO7964:POKER+OF,7:N
   EXT
51 IFP1=>3THENPOKE8012,J:POKE8029,J
52 IFP1=>4THENPOKE8105,J:POKE8112,J
53 IFP1=>5THENPOKE7951,J:POKE7958,J
54 GOTO13
55 POKE7754,8:POKE7755,9:POKE7756,20:POKE7799
   ,153:POKE7840,20:POKE7841,15:POKE7843
   ,16:POKE7844,12
56 POKE7845,1:POKE7846,25:POKEM+1,85:POKEM+2,
   88:POKEM+3,73:POKE7783,42:POKE7903,21
   :POKE7904,19:POKE7905,5
57 POKE7907,3:POKE7908,21:POKE7909,18:POKE791
   0,19:POKE7911,15:POKE7912,18:POKE7914
   ,11:POKE7915,5
58 POKE7916,25:POKE7917,19:POKE7925,6:POKE792
   6,15:POKE7927,18:POKE7929,18:POKE7930
   ,9:POKE7931,7
59 POKE7932,8:POKE7933,20:POKE7935,38:POKE793
   7,12:POKE7938,5:POKE7939,6:POKE7940,2
   0
60 GETA$:IFA$="Y"THENSC=0:L=1:GOTO4
61 IFA$<>="Y"THENPOKEVA,0:TT=TT+1:IFTT=500THE
   N4
62 GOTO60
65 SC=SC-F:G=M+2:IFHI=SC+FTHENHI=HI-F
66 POKEG+22,77:POKEC,10:POKEG+OF+22,1:G=G+22
67 IFPEEK(G+22)=81ORPEEK(G+22)=65THENPOKEG,81
   :POKEG+OF,9:GOTO72
68 IFPEEK(G+22)=87THENPOKEG,81:POKEG-22,81:G=
   G-22:POKEG+OF,9:POKEG+OX,9:GOTO74
69 IFG>8185THENPOKEG,81:POKEG+OF,9:GOTO72
70 IFPEEK(G)=77THENPOKEG+22,78:POKEG+22+OF,1:
   G=G+22:GOTO67
71 GOTO66
72 IFPEEK(G-1)=32THENPOKEG-1,81:POKEG-1+OF,9
73 IFPEEK(G+1)=32THENPOKEG+1,81:POKEG+1+OF,9
74 POKEG-22,32:G=G-22:IFPEEK(G-22)=88THEN76
```

```
75 GOTO74
76 FORR=255TO128STEP-.9:POKEVA,R:NEXT:POKEVA,
   0
77 X=INT(RND(1)*18)+1:DY=1:Y=1:DX=1:IFX=>11TH
   ENDX=-DX
78 IFX=<12THENDX=+DX
79 IFDX=>50THENDX=1
80 GOSUB25:FORR=1TO750:POKEC,105:GOTO105
85 POKEBD,91:POKEBD,90:DX=+DX:DY=-DY:POKEBD,9
   1:GOSUB25:POKEBD,32:GOTO105
90 FORR=1TO15:FORW=250TO240STEP-1:POKEVA,W:NE
   XT:FORW=240TO250:POKEVA,W:NEXT:POKEVA
   ,0:NEXT:P1=P1+1
```



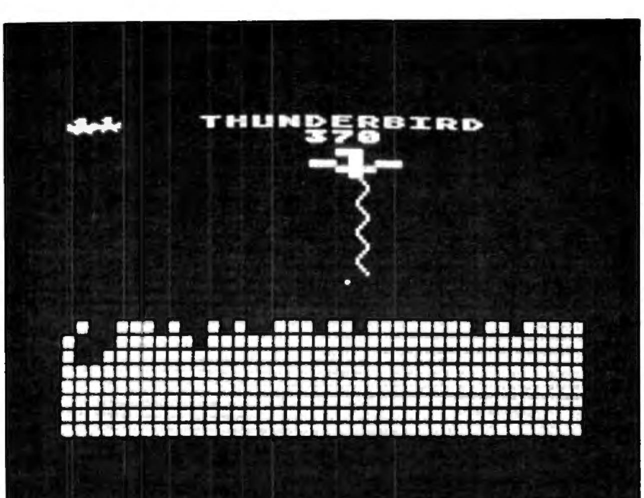
Getting ready to play another game of the VIC-20 version of "Thunderbird."

```
91 FORR=1TO100:SC=SC+10:POKEVA,245:GOSUB25:FO
   RW=1TO10:NEXT:POKEVA,0:NEXT:GOTO4
95 IFL>1THENP1=1:FORR=0TO255:POKEC,R:POKEVA,I
   NT(RND(0)*128+127):NEXT:POKEC,105:POK
   EVA,0:GOTO55
96 IFL<3THENL=L+K:POKE7767,147:POKE7789,133:P
   OKE7811,131:POKE7833,143:POKE7855,142
   :POKE7877,132:X=17
97 DX=1:Y=1:DY=1:POKEC,47:FORR=1TO28:READA:PO
   KEVA,A:POKEC,A:FORW=1TO50:NEXTW,R
98 POKEVA,0:RESTORE:POKEC,105:DX=-DX:DY=+DY:G
   OTO104
100 IFM<7703THEN104
101 POKEM,85:POKEM+1,88:POKEM+2,73:POKEM+3,32:
    M=M-1:GOTO104
102 IFM>7718THEN104
103 POKEM+2,85:POKEM+3,88:POKEM+4,73:POKEM+1,3
    2:M=M+1
104 POKEBO,32:BO=7703+X+22*Y:POKEBO,42
105 X=X+DX:IFX=0ORX=19THENDX=-DX:POKEVA,240
106 Y=Y+DY:IFY=-1THEN95
107 IFY=22THEN90
108 IFDX=0THENDX=1
109 POKEVA,0:POKEVD,0:BD=7703+X+22*Y
110 IFPEEK(BD)=32THEN116
111 POKEBO,32:IFPEEK(BD)=JTHEN65
112 IFPEEK(BD)=81THENPOKEVA,238:POKEVD,238:SC=
    SC+50:GOTO85
113 IFPEEK(BD)=65THENSC=SC+200:FORR=128TO255ST
    EP2:POKE36875,R:NEXT:POKE36875,0:GOTO
    85
114 IFPEEK(BD)=85ORPEEK(BD)=73THENPOKEVA,140:S
    C=SC+75:GOSUB25:DX=+DX:DY=-DY:GOTO105

115 IFPEEK(BD)=88THENPOKEVA,212:SC=SC+25:GOSUB
    25:DX=0:DY=-DY:GOTO105
116 IFPEEK(197)=31THEN100
117 IFPEEK(197)=23THEN102
118 GOTO104
125 DATA 218,218,218,218,160,148,136,149,142,1
    32,133,146,130,137,146,132,160,218,21
    8,218
126 DATA218,218,218,218,218,218,218,134,137,14
    6,147,148,160,147,129,148,133,140,140
    ,137
127 DATA148,133,218,218,218,218,147,131,143,14
    6,133,160,160,160,160,160,160,136,137
    ,160
128 DATA160,160,160,160,160,160
```

Program 2: Atari Version

```
100 REM  THUNDERBIRD 
110 REM Atari Version
120 GRAPHICS 0:BASE=(PEEK(106)-16)*25
    6:GOSUB 1560:REM remove old playe
    rs from screen
130 DIM A$(40),BALL$(4):POKE 82,0:BAL
    L$="*■(J}(■)":BALLS=4
140 CHSET=BASE:IF PEEK(CHSET+9)<>252
    THEN GOSUB 1200:REM If not initia
    lized
150 GRAPHICS 0:POKE 752,1:POKE 559,0:
    REM Turn off cursor, screen
160 DLIST=PEEK(560)+256*PEEK(561)+4:R
    EM location of display list
170 FOR I=3 TO 24:POKE DLIST+I,4:NEXT
     I:REM Change mode zero lines to
    IRG 4 (mulicolor character)
180 POKE DLIST-1,6+64:POKE DLIST+2,6:
    REM top two lines GRAPHICS 1
190 POKE 756,CHSET/256:REM turn on ch
    aracter set
200 SETCOLOR 0,0,12:SETCOLOR 1,3,6:RE
    M white and red
210 RESTORE 240:REM draw brick area
220 POSITION 0,0:? BALL$(1,BALLS):REM
     display # of balls (birds) left
230 REM Pattern of wall:
240 DATA 1,1,2,3,14,129,130,131
250 SCR=PEEK(88)+256*PEEK(89):REM loc
    ate screen memory
260 REM put bricks on screen
270 FOR I=SCR+520 TO SCR+800 STEP 40:
    READ A:FOR J=I TO I+39:POKE J,A:N
    EXT J:NEXT I
280 POSITION 5,0:? #6;"THUNDERBIRD"
290 P0=BASE+1024:PADR=P0+48:REM playe
    r zero.
300 POKE 704,28+176*(DIFF=1)+80*(DIFF
    =2):REM Gold, green, or violet
310 POKE 54279,BASE/256:REM single-li
    ne res.
320 POKE 53277,3:POKE 53256,3-2*(DIFF
    =1)-3*(DIFF=2):REM Start P/M DMA,
     select width according to diffic
    ulty
330 RESTORE 370
340 FOR I=0 TO 21:POKE PADR+I,0:NEXT
    I:REM clear out player
350 FOR I=0 TO 7*(3-DIFF) STEP 3-DIFF
    :READ A:FOR J=0 TO 3-DIFF:POKE PA
    DR+I+J,A:NEXT J:NEXT I
360 REM bird pattern
370 DATA 0,24,8,107,28,8,0,0
380 IF PEEK(547)<>6 THEN A=USR(1536):
    REM turn on VBLANK if necessary
390 POKE 559,62:GOSUB 750:REM turn on
     screen (single-line res. P/M), w
    ait for START
400 DY=1:DX=0.5:IF RND(1)>0.5 THEN DX
    =-0.5:REM Set up ball direction
410 BX=INT(40*RND(0)):BY=INT(7*RND(0)
    +3):REM select random starting po
```

```
sition
420 REM  Main Loop 
430 IF STRIG(0)=0 THEN GOSUB 800:REM
    allow "thunder"
440 IF STICK(0)<>15 THEN POKE 77,0
450 TX=BX+DX:TY=BY+DY:REM update ball
460 IF TY<1 THEN GOSUB 600:GOTO 430:R
    EM check for miss
470 IF TY>20 THEN DY=-DY:GOSUB 920:GO
    TO 430:REM check for breakthrough
480 IF TX<0 OR TX>39 THEN DX=-DX:REM
    bounce off wall
490 TPOS=SCR+TX+40*TY:REM check for o
    bstacles
500 IF PEEK(TPOS)=0 THEN POKE TPOS,5:
    POKE SCR+BX+40*BY,0:BX=TX:BY=TY:G
    OTO 430
510 REM Rebound tiles (lasered down)
520 IF PEEK(TPOS)=4 THEN GOSUB 890:SC
    ORE=SCORE-50:DY=ABS(DY):GOTO 560
530 DY=-ABS(DY):IF RND(0)>0.5 THEN DX
    =-DX
540 FOR W=14 TO 0 STEP -2:SOUND 0,W*5
    ,10,W:NEXT W
550 SCORE=SCORE+(BY-11)*5:BLOCKS=BLOC
    KS+1:REM score according to row
560 POKE TPOS,0:POSITION 29-LEN(STR$(
    SCORE))/2,0:? " ";SCORE;" ";
570 IF BLOCKS=320 THEN 1000:REM BREAK
    -OUT!
580 IF SCORE<0 THEN 720
590 GOTO 500
600 REM Hit bird?
610 IF PEEK(53252) THEN DY=-DY:Z=1:GO
    TO 630
620 GOTO 660
630 FOR W=14 TO 0 STEP -2:SOUND 0,W+1
    0,10,W:NEXT W
640 POKE SCR+BX+40*BY,0:BX=BX+DX:BY=B
    Y+DY
650 POKE 53278,255:RETURN
660 REM Ball out of bounds (past bird
    )
670 POKE SCR+BX+40*BY,0
680 FOR W=100 TO 0 STEP -5:SOUND 0,W,
    12,8:NEXT W:FOR W=W=0 TO 100 STEP
     5:SOUND 0,W,12,8:NEXT W:SOUND 0,
    0,0,0
690 POKE 53278,255
700 BALLS=BALLS-1:POSITION BALLS,0:?
    " ";
```



Unleashing a lightning bolt in the Atari version of "Thunderbird."

```
710 IF BALLS>0 THEN 400
720 REM  GAME OVER 
730 POSITION 5,0:? " GAmE OVeR "
740 GOSUB 750:RUN
750 IF PEEK(53279)=6 THEN POSITION 20
    ,0:? "{5 SPACES}":RETURN
760 IF PEEK(20)>20 THEN POSITION 20,0
    :? "PRESS"
770 IF PEEK(20)>40 THEN POSITION 20,0
    :? "start":POKE 20,0
780 GOTO 750
790 REM LASER DOWN
800 XPOS=(PEEK(1664)-48)/4+4:FLIP=0:R
    EM equate player pos. to screen p
    os.
810 FOR I=3 TO 12:WHERE=SCR+XPOS+40*I
820 P=PEEK(WHERE):POKE WHERE,6+FLIP:F
    LIP=1-FLIP:REM zig-zag line
830 SOUND 0,I*10,0,15-I:POKE 710,PEEK
    (53770):REM scintillate color
840 NEXT I
850 FOR I=3 TO 12:POKE SCR+XPOS+40*I,
    0:NEXT I:REM erase lightning
860 WHERE=SCR+12*40+XPOS:SOUND 0,0,0,
    0:POKE WHERE-1,4:POKE WHERE+1,4:P
    OKE WHERE,4:REM lay down tiles
870 SETCOLOR 2,9,4:RETURN
880 REM sound effect:
890 FOR W=0 TO 240 STEP 30:SOUND 0,W,
    12,15-W/17:SOUND 1,W+10,10,15-W/1
    7:NEXT W:SOUND 0,0,0,0:SOUND 1,0,
    0,0
900 RETURN
910 REM break-through
920 IF DONE THEN RETURN
930 FOR I=1 TO 100:POKE 53274,PEEK(53
    770):SOUND 0,I,0,15-I/10:NEXT I
940 SOUND 0,0,0,0:POSITION 4,0:? "bre
    akthrough":POSITION 22,0:? "1000
    point BONUS"
950 FOR I=1 TO 10:POSITION 22,0:? "10
    00":FOR W=1 TO 20:NEXT W:POSITION
     22,0:? "{4 SPACES}":FOR W=1 TO 2
    0:NEXT W:NEXT I
960 POSITION 4,0:? " THUNDERBIRD ":PO
    SITION 22,0:? "{17 SPACES}"
970 FOR I=1 TO 10:FOR J=0 TO 15 STEP
    5:SOUND 0,50+10-I,0,15-J:NEXT J:S
    CORE=SCORE+100
980 POSITION 29-LEN(STR$(SCORE))/2,0:
    ? " ";SCORE;" ";
990 NEXT I:DONE=1:RETURN
1000 REM  All bricks cleared
1010 GOSUB 1100:REM do "BLAST"
1020 FOR I=1 TO 50:FOR J=0 TO 3:POKE
     708+J,PEEK(53770):NEXT J:Z=Z*(Z<
     5)+1
1030 SOUND 0,I+Z,10,I/10:SOUND 1,I+Z+
     10,10,I/10:NEXT I
1040 SOUND 0,0,0,0:SOUND 1,0,0,0:GOSU
     B 1560
1050 GRAPHICS 18:POSITION 0,6:? #6;"
     {Q}{P}{L}{3 P} point bonus{R}"
1060 FOR W=1 TO 100:SOUND 0,PEEK(5377
     0),0,15-W/10:POKE 712,(3-FLIP*2)
     *16+FLIP*4+4:FLIP=1-FLIP:NEXT W
1070 SCORE=SCORE+10000:SOUND 0,0,0,0
1080 DIFF=DIFF+1:IF DIFF>2 THEN DIFF=
     2
1090 GOTO 150
1100 POKE 82,5:POSITION 5,10
1110 ? "!!!!  #{6 SPACES}■{4 SPACES}.
     ..   !!!!!"
1120 ? "!{3 SPACES}! #{5 SPACES}■ ■
     .{3 SPACES}.{3 SPACES}■"
```

Atari® Games On Your VIC-20?

The "CARDAPTER/1" will allow Video Game Cartridges designed for use on the Atari Video Computer System to be played on a standard VIC-20®

SUGGESTED RETAIL $89.95

DEALER INQUIRIES INVITED

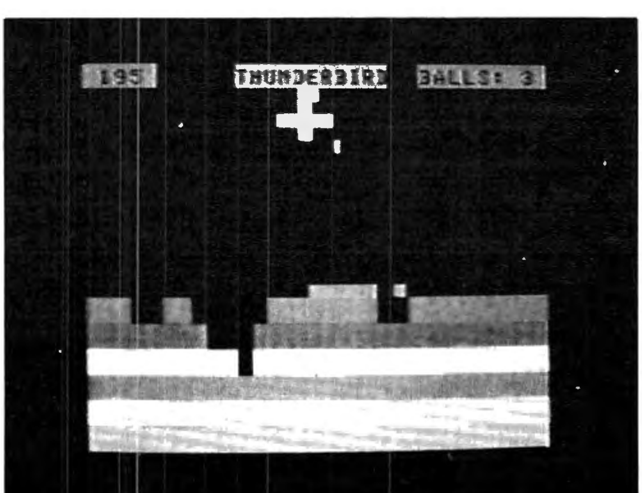
United States:
Cardco, Inc.
3135 Bayberry
Wichita, KS 67226
(316) 685-9536

West Canada:
LSI Distributing
Attn: Mr. Wong
2091 W. 61st Avenue
Vancouver, BC. CA V6J 1Z2
(604) 733-0211

England & Europe:
Audiogenic
Martin Manary
34-36 Crown St.
Reading, Berkshire England
(0734) 595647

East Canada:
Hobby Craft Canada
24 Ronson Drive
Rexdols Ontario M9W1B4
(416) 241-2661

```
1130 ? "!{3 SPACES}! #{4 SPACES}■
     {3 SPACES}■ .{7 SPACES}■"
1140 ? "!!!!  #{4 SPACES}■{3 SPACES}■
       ...{4 SPACES}■"
1150 ? "!{3 SPACES}! #{4 SPACES}■■■■■■■■
     {5 SPACES}.{3 SPACES}■"
1160 ? "!{3 SPACES}! #{4 SPACES}■
     {3 SPACES}■ .{3 SPACES}.
     {3 SPACES}■"
1170 ? "!!!!  #### ■{3 SPACES}■  ...
     {4 SPACES}■"
1180 POKE 82,0:RETURN
1190 END
1200 REM  Initialization stuff
1210 POKE 88,0:POKE 89,BASE/256:? "
     {CLEAR}":GRAPHICS 2+16:REM CLEAR
     S OUT P/M AND CHARACTER MEMORY
1220 POSITION 5,0:? #6;"thunderbird":
     POSITION 6,4:? #6;"patience":POS
     ITION 5,8:? #6;"READING ML"
1230 RESTORE 1260
1240 FOR I=1536 TO 1611:READ A:SOUND
     0,A,10,8:POKE 712,A:POKE I,A:NEX
     T I
1250 A=USR(1536):GOTO 1400
1260 DATA 104,173,34,2,141,74
1270 DATA 6,173,35,2,141,75
1280 DATA 6,169,6,162,6,160
1290 DATA 23,32,92,228,96,24
1300 DATA 173,128,6,141,0,208
1310 DATA 173,124,2,208,6,206
1320 DATA 128,6,206,128,6,173
1330 DATA 125,2,208,6,238,128
1340 DATA 6,238,128,6,173,128
1350 DATA 6,201,1,176,5,169
1360 DATA 200,141,128,6,201,250
1370 DATA 144,5,169,32,141,128
1380 DATA 6,76,73,6
1390 REM
1400 POSITION 3,8:? #6;"LOADING  CHSE
     T"
1410 FOR I=128 TO 510:POKE CHSET+I,PE
     EK(57344+I):SOUND 0,I/2,12,8:POK
     E 712,I/2:NEXT I
1420 RESTORE 1460
1430 READ A:IF A=-1 THEN SOUND 0,0,0,
     0:SOUND 1,0,0,0:RETURN
1440 FOR J=0 TO 7:READ B:SOUND 0,B,10
     ,8:SOUND 1,B+10,10,8:POKE 712,B:
     POKE CHSET+A*8+J,B:NEXT J
1450 GOTO 1430
```



Making a "wing shot" in the TRS-80 Color Computer version of "Thunderbird."

```
1460 DATA 1,0,252,168,84,252,168,252,
     0
1470 DATA 2,0,168,168,252,252,168,168
     ,0
1480 DATA 3,0,216,120,184,228,180,212
     ,0
1490 DATA 4,0,0,0,219,150,0,0,0
1500 DATA 5,0,0,0,16,32,0,0,0
1510 DATA 6,192,192,48,48,12,12,3,3
1520 DATA 7,3,3,12,12,48,48,192,192
1530 DATA 10,24,40,24,153,126,255,20,
     34
1540 DATA 14,0,126,126,126,126,126,12
     6,0
1550 DATA -1
1560 REM KILL P/M GRAPHICS
1570 POKE 53277,0:FOR I=0 TO 3:POKE 5
     3261+I,0:NEXT I
1580 RETURN
```

Program 3: TRS-80 Color Computer Version

```
100 ' THUNDERBIRD
110 ' COLOR COMPUTER VERSION
120 CLS 0
125 BL$=CHR$(128)+CHR$(128)+CHR$(128)
    +CHR$(128)
130 PRINT @10,"THUNDERBIRD";
140 H1$=CHR$(128)+CHR$(157)+CHR$(154)
150 H2$=CHR$(128)+CHR$(149)+CHR$(158)
160 BD$=CHR$(156)+CHR$(157)+CHR$(158)
    +CHR$(156)
165 BALLS=4
170 ' DRAW WALL OF BRICKS
180 FOR I=2 TO 7
190 PRINT @(I+7)*32,"";
200 FOR J=1 TO 32:PRINTCHR$(143+I*16)
    ;:NEXT
210 NEXT
220 ' INITIALIZE OTHER VARIABLES
230 BY=RND(10)+6:BX=RND(32)-1
240 DY=1:DX=-1:IF RND(0)>.5 THEN DX=1
245 PRINT@23,"BALLS:";BALLS;
299 ' MAIN LOOP
300 X=INT(JOYSTK(0)/2)
310 IF X>28 THEN X=28
320 IF X=OLDX THEN 360
325 PRINT@OLDX+32,BL$;:PRINT@OLDX+64,
    BL$;
330 IF X<OLDX THEN PRINT@X+32,H1$; EL
    SE PRINT @X+32,H2$;
340 PRINT@X+64,BD$;
350 OLDX=X
360 IF PEEK(65280)=126 OR PEEK(65280)
    =254 THEN GOSUB 4000
499 ' MOVE BALL
500 TX=BX+DX:TY=BY+DY
515 IF TY=31 THEN DY=-DY:GOTO700
520 IF TY>5 THEN 600
530 IF TX<X*2 OR TX>X*2+LEN(BD$)*2 TH
    EN 1000
540 DY=-DY:IF RND(0)>.5 THEN DX=-DX
550 GOTO 700
600 IF TX<0 OR TX>63 THEN DX=-DX:GOTO
     700
610 P=POINT(TX,TY)
620 IF P=0 THEN RESET(BX,BY):SET(TX,T
    Y,0):BX=TX:BY=TY:GOTO300
625 IFP=1 THEN P=-5
630 SCORE=SCORE+P*5:PRINT@0,SCORE;
635 IF SCORE<0 THEN 1030
640 PRINT@INT(TX/2)+INT(TY/2)*32,CHR$
    (128);
```

the SOFTWARE connection

FOR ALL YOUR SOFTWARE NEEDS AT THE LOWEST PRICES

We have one of the largest selections of software available for your home computer at the lowest prices. You will find all of the top games and office management software in our catalog at from 20% to 30% below retail.

ATARI®

	Retail	Our Price
Monster Maze (Rom)	$39.95	$29.95
Platter Mania (Rom)	$39.95	$29.95
Choplifter (Disk)	$34.95	$25.95
Embargo (Rom)	$49.95	$34.95
Raster Blaster (Disk)	$29.95	$18.00
Frogger (D/C)	$34.95	$24.95
Wizzard of Wor (D/C)	$39.95	$29.95
Slime (D/C)	$34.95	$24.95
Serpentine (D/C/Rom)	$34.95	$24.95
Steller Shuttle (D/C)	$29.95	$20.95

Atari® & Atari APX® programs now available at 25% off suggested retail if you mention this ad.

TG Trackball	$64.95	$51.95
Wico Trackball	$69.95	$57.95

And many, many more. Call for an update of new titles, including Atari VCS®.

IBM

	Retail	Our Price
Apple Panic	$29.95	$23.95
Call to Arms	$29.95	$23.95
Curse of Ra	$19.95	$15.95
Frogger	$34.95	$27.95
Jabber Talky	$29.95	$23.95
Temple of Apshai	$39.95	$29.95
Upper Reaches of Apshai	$19.95	$15.95
Oil Barons	$100.00	$75.00
*MatheMagic	$89.95	$67.50
*Graphmagic	$89.95	$67.50
*Combo Rack	$159.90	$119.95

Visicorp/Micropro - 25% of all titles including Visicalc & Wordstar.

TG Joy Stick	$64.95	$49.95

$24.95
DISKETTE SPECIAL
Box of 10 with Plastic Library Case
Single-sided/Double Density (40 track) with reinforced hub-ring.
Call for Prices on Maxell and Verbatim.

VIC

Call for more information on new software for the fastest growing Home Computer today.

	Retail	Our Price
Crush, Crumble & Chomp (Cass.)	$29.95	$23.95
Astroblitz (Cart.)	$46.95	$37.50
Choplifter (Cart.)	$44.95	$35.95
Martian Raider (Cass.)	$19.95	$15.95
Ricochet (Cass.)	$19.95	$15.95
Sword of Fargoal (Cass.)	$29.95	$23.95
Serpentine (Cart.)	$44.95	$35.95
Apple Panic (Cart.)	$44.95	$35.95
Shark Trap (Cass.)	$19.95	$15.95
Trashman (Cart.)	$46.95	$37.50
Rescue at Rigel (Cass.)	$29.95	$23.95
Multisound Synthesizer (Cass.)	$19.95	$15.95
Monster Maze (Cart.)	$39.95	$31.95

APPLE

	Retail	Our Price
Sea Fox	$29.95	$22.45
Serpentine	$34.95	$26.25
*MatheMagic	$89.95	$67.50
*Graphmagic	$89.95	$67.50
*Combo Pack	$149.95	$112.50
Oil Baron	$100.00	$70.00
New World	$29.95	$22.45
Snooper Troops	$44.95	$33.75
Wizardry	$49.95	$37.50
Star Blazer	$31.95	$20.00
Ruski Duck	$34.95	$18.00
3-Pack: Rescue at Rigel, Morlock's Tower, Datestone of Ryn	$49.95	$37.50
Raster Blaster	$29.95	$18.00
TG Joystick Apple II®	$59.95	$46.00
TG Joystick Apple III®	$64.95	$48.95
TG Select-a-Port	$59.95	$46.00

CALL TOLL FREE 1-800-828-2838 (For Placing Orders Outside California)

For Inside California and Other Inquiries Call 1-916-925-2666

MAIL ORDERS: For fast delivery, send certified check, money orders, or Visa or MasterCard number and expiration date, for total purchase price plus 1% or $2 minimum for postage and handling. Add $5 for shipment outside the continental U.S. California Residents add 6% sales tax.

COD: and Chargecard orders call **1-800-828-2838.** In California call **1-916-925-2666.** Subject to stock on hand. Prices subject to change.

Catalog free with any order or send $2 postage and handling and please specify computer type.

the SOFTWARE connection

5133 Vista Del Oro Way Fair Oaks, CA 95628

```
641 IF TY=28 THEN GOSUB 2000
645 IF P=-5 THEN DY=ABS(DY):GOTO700
650 DY=-ABS(DY)
660 HIT=HIT+1:IF HIT=192 THEN 3000
700 SOUND 240,1
710 GOTO 300
1000 FORI=1TO10:SOUND1,1:SOUND255,1:N
     EXT
1010 RESET(BX,BY)
1020 BALLS=BALLS-1:IF BALLS>0 THEN 23
     0
1030 FORI=255TO1STEP-15:SOUNDI,1:NEXT
1040 CLS5
1050 PRINT@267,"GAME OVER";
1055 PRINT@0,SCORE;
1060 IF PEEK(65280)<>126 AND PEEK(652
     80)<>254 THEN 1060
1070 RUN
1999 ' BREAKTHROUGH
2000 IF FIRST=1 THEN RETURN
2010 FIRST=1
2015 PRINT@6,"1000 POINT BONUS";
2020 FOR J=1 TO 10 STEP 2
2025 T=1-T:IF T THEN PRINT@6,"1000";
     ELSE PRINT@6,BL$;
2030 FOR I=200+J TO 210+J
2040 SOUND I,1
2050 NEXT:NEXT
2060 PRINT@6,BL$;"THUNDERBIRD";BL$;
2070 SCORE=SCORE+1000
2080 PRINT@0,SCORE;:PRINT@23,"BALLS:"
     ;BALLS;
2090 RETURN
2999 ' ALL BRICKS HIT
3000 FOR I=1 TO50
3010 CLS RND(8)-1
3020 PRINT@263,"10,000 POINT BONUS!";
3030 NEXT:CLS 0
3040 SCORE=SCORE+10000
3050 DIF=DIF+1
3060 IF DIF=3 THEN DIF=2
3070 ON DIF GOTO 3100,3200
3100 H1$=""
3110 H2$=H1$
3120 BD$=CHR$(169)+CHR$(166)
3130 GOTO 180
3200 H1$="":H2$=""
3210 BD$=CHR$(243):GOTO180
4000 ' THUNDER DOWN
4010 FORI=3TO7
4020 PRINT@X+1+I*32,CHR$(233);
4050 NEXT
4060 SOUND245,10
4070 FORI=3TO7
4080 PRINT@X+1+I*32,CHR$(128);
4090 NEXT
4100 PRINT@X+I*32,CHR$(131);CHR$(131)
     ;CHR$(131);
4110 RETURN
```

Atari Notes:

Thunderbird

Charles Brannon, Editorial Assistant

Thunderbird for Atari requires a joystick and 16K of memory. You move the bird left and right with your joystick, using it to bounce a ball into a wall of bricks. As in Breakout, the object of the game is to clear out all the bricks, without letting the ball escape past you. A 1,000 point bonus is awarded when you break out the bottom of the wall (a "breakthrough"); and if you're really good, you get 10,000 points for clearing out all the bricks (no mean feat!).

Shades Of Zeus

But the Thunderbird is no mere Breakout paddle! It can unleash the most awesome power of nature – lightning – at the touch of a button (the fire button). Thunderbird will "beam down" several luminous "tiles" that serve to deflect the ball downwards when hit. You can lay down tiles like a cap over a hole the ball has created, to force it to widen the hole. Every time the ball hits a tile, it swoops downward, but 25 points are subtracted from your score. That should discourage overuse of this miraculous feature.

Vertical Blank And IRG 4

Here's a bit of information about the programming. The playing field is a mixed-mode display consisting of two rows of GRAPHICS 1 text, and 21 rows of a multicolored character mode, IRG 4. This lets us have multicolored bricks.

Player/missile graphics are used to represent the bird, which can be any of three sizes, depending on the skill level. The bird is moved left and right by a small machine language routine that is executed every 1/60 second during the TV's vertical blank (when the electron beam is traveling from the lower right-hand corner to the upper left-hand corner of the screen).

IRG mode 4, the multicolor mode, is quite interesting. A single character can be any of three colors. To design these colored characters, divide the character horizontally into four two-bit zones. Each two-bit block controls one pixel of color within the character (a multicolor character's resolution is 4x8). No color would be 00, color one is 01, two 10, and three 11 (simple two-bit binary). For example, one of the bricks consists of several colored bands:

```
1110
2220
3330
1110
2220
3330
1110
0000
```

The numbers correspond to a "COLOR" statement. One side and the bottom row are left blank, so the blocks won't touch. The pattern, when expanded into binary, would look like:

```
01010100
10101000
11111100
01010100
10101000
11111100
01010100
00000000
```

Such a "custom character" would look strange on a normal screen (although you would see some semblance of multicolors, due to artifacting). But when displayed on either an IRG 4 or IRG 5 mode screen, each character is like a tiny 4x8 block of GRAPHICS 7 pixels. Also, any character printed in inverse (with the Atari logo key) will look different. The COLOR 3 pixels in such a character will be displayed as COLOR 4 (normally available only in GRAPHICS 1 or 2).

To create an IRG 4 screen, you must replace the bytes for GRAPHICS 0 by modifying the display list. Luckily, the resolution of IRG 4 is identical to GRAPHICS 0, 40x24.

```
DL=PEEK(560)+256*PEEK(561)+4
POKE DL-1,4+64
FOR I=2 TO 24:POKE DL+I,4:NEXT I
```

See lines 160-180 of Thunderbird. You can also try out IRG 5, which displays these characters in double-height (40x12). C

HOME ENERGY CALCULATOR

David Swaim, Atlanta

You (and your computer) could become pretty popular when word gets out that you can analyze the benefits of home improvements on fuel bills. This program is in Microsoft (Apple, PET, OSI, etc.) and Atari BASIC.

Lately there has been a great deal of interest in saving energy in the home. Nobody needs to be reminded that fuel costs are rising. We all want to reduce our energy bills. The way to do this is simple: reduce household energy consumption. There are a number of ways this can be done.

The cheapest way is to change habits. An example would be setting the thermostat back to a lower temperature and wearing heavier clothes. If you're not too keen on that, the next alternative is to improve the ability of the house to protect you from the elements. Insulation could be added to the walls, floors, attic, and heat ducts. Weatherstripping could be applied to windows and doors. Storm windows and doors could be added.

Improvements such as these reduce the amount of heat that the house will lose to the outside. But which of the above items would save us the most money? Which one would cost the least to implement? Or, better yet, which will give the greatest savings for the least amount of cost? It's this last question we really want to answer.

The best measure of the cost effectiveness of an energy saving improvement is the payback period. That is simply the amount of time (in years) it takes for the savings in energy costs to add up to the total cost of installing the improvement. Obviously, the item with the shortest payback period is the best candidate for implementation. To determine the payback period, we must know two things: how much it will cost to make the improvement, and how much it will save us on utility bills for a year (a heating season). Obtaining the improvement cost requires consulting a contractor or, if we plan to do it ourselves, a building supply store.

Predicting Effectiveness

Finding out how much the improvement will save us in heating costs over a season is not quite as easy to determine. One way would be to keep records of our heating bills for one season, make the improvement, and then keep records of our heating bills for the next heating season. There are two drawbacks to this method.

First, the severity of the weather will vary from one year to the next. If the first year is severe and the second is mild, our heating bills would be less even if we made no improvements. This problem can be corrected by adjusting the heating costs using weather data for the two years.

The second and biggest drawback to this method is that you can't find out if an improvement is cost effective until after you have installed it. If it turns out not to be cost effective, it is too late to decide not to implement it!

What we need is a way of *predicting* savings. If we know the weather and the heat loss characteristics of the house, we can estimate the heating cost. By calculating the heating costs based on heat loss characteristics of the house both before and after the improvements, we can obtain the estimated savings due to the improvements. This is what the program here does.

To gather the data needed by the program, you will need to make some measurements and observe insulation levels in your house. The first thing the program calculates is the heat loss of the house. Heat loss of a house depends on three things: the thermal resistance, known as the R-value, of the structure; the total area of the structure exposed to the elements; and the temperature difference between the inside and outside of the house. So we simply need the area, R-value, and the difference in temperature.

The only problem is that different parts of the house have different R-values. Windows will have a lower R-value than walls, for example. In general, you can divide the external area of the house into five categories: windows, doors, walls, ceiling, and floor. The program requests information on each of these five categories in turn.

For windows it requests height, width, number of windows (it calculates total window

Now the VIC 20 and 64 can communicate with PET peripherals

VIC and 64 users

Would you like to be able to access **any** of these peripherals from your computer?

- ⅓ megabyte disks (Commodore 4040 drive)
- 1 megabyte disks (Commodore 8050 drive)
- 10 megabyte disks (Commodore 9090 hard disk)
- Printers including a wide range of inexpensive IEEE and RS232 matrix and quality printers
- IEEE instruments such as volt meters, plotters etc.

Now you are no longer limited by the VIC or the 64's serial bus. Simply by attaching INTERPOD you can vastly increase the power of your VIC 20 and when used with the new 64, INTERPOD turns the computer into a really powerful system.

With INTERPOD the VIC and 64 become capable of running really professional quality software such as Word-processing, Accounting, Instrument control and many more.

INTERPOD will work with any software. No extra commands are required and INTERPOD does not affect your computer in any way.

Using INTERPOD is as easy as this:

Simply plug INTERPOD into the serial port of your computer, power-up and you are ready to communicate with any number of parallel and serial IEEE devices and any RS232 printer.

INTERPOD costs $180

INTERPOD

Oxford Computer Systems (Software) Ltd. Hensington Road, Woodstock, Oxford OX7 1JR, England Tel. (0993) 812700 Telex 83147 Ref OCSL

INTERNAL MEMO

I want a Word Processor that's simple to understand, fast, and easy to use, , ,

.. it'll need up to 20,000 characters of text, a wide, clear screen, and be able to spell!

She needs 'Superscript'!

Superscript Features

- Provides full screen edit, delete, erase, insert, transfer and document merge facilities. Over 240 columns.
- Simple facilities to send mailshots to standard address lists.
- Powerful aids to produce tables with wide screen, horizontal, decimal and vertical tabs.
- Search with optional replace
- Comprehensive printer controls with control of margins, lines per page, underlining, bold print, super and sub scripts, variable line and character pitch.

Superspell Features

- Dictionary in excess of 30,000 words.
- User definable dictionary with facilities to add, delete, print or merge with standard dictionary.
- Spelling checker verifies largest *'Superscript'* document in less than two minutes.
- Displays totals of words, different words and unrecognized words.
- Editing of unrecognized words in document includes options to accept, ignore, change or add to user dictionary.

'Superscript' transforms your Commodore computer into a true Word Processor, enabling your secretary to turn out high quality letters, mailshots, quotations, etc., faster and easier than ever before.

But spell? — Adding *'Superspell'* gives you access to an extensive dictionary with automatic checking of any document entered, enabling recognition of spelling errors and mistypes.

If she wants a better machine, and you want the very best of Word Processing, then you both need a Commodore with *'Superscript'* and *'Superspell'* available from your local Commodore dealer.

Superscript* and *Superspell run on the 2001, 3016/32, 4016/32, 8032 and 8096 Commodore computers, 2040, 3040, 4040 and 8050 disk drives, all Commodore printers and a wide range of letter quality printers.

Superscript

A Professional's Word Processor

Precision Software

Precision Software Limited,
4, Park Terrace, Worcester Park,
Surrey KT4 7 JZ, England.
Telephone 01-330 7166

area from these items), and type of frame and number of layers of glass. The number of types and/or sizes of windows is requested first. Most houses will have several sizes of windows, and there may be storm windows on some and not on others. The program allows for up to ten different types and/or sizes of windows. If you need more, change the dimension of S in statement 180.

Only one size and type of door is allowed. If you have sliding glass doors, you should consider them another type of window. You need to get the height, width, and number of doors. Remember: these are exterior doors only.

Information needed for the walls consists of type of construction and R-value of the insulation in the wall. If you enter a negative number for the R-value of the wall insulation, the program will give you a list of typical R-values for wall insulation. To get the area of the wall, the program asks for the ceiling height, total perimeter of the house, and the number of stories in the house. The program will calculate the gross wall area from this data and subtract the total window and door area to obtain the proper wall area.

One Hand Calculation

The only time you have to calculate area yourself is for ceiling and floor. For the ceiling, you will be asked for the number of inches of insulation in the attic and the type of insulating material. For the floor, the type of foundation is requested.

In addition to the heat losses mentioned so far, there are two others. The first of these is infiltration of outside air through cracks in windows and doors. The program asks if the windows and doors are weather-stripped. It uses this information and the total length of the cracks around windows and doors to calculate infiltration. The other heat loss is in the heat ducts from the furnace to the heat registers. The program asks if your heat ducts are insulated and where they are located. This concludes the input needed for calculating the total heat loss of the house. At this point the heat losses are displayed, and you are asked if you wish to make improvements to the house.

If the answer is "Y", you will be asked if you wish to improve each item. You can make improvements to one item or to any number of items. As you probably noticed, the first question you are asked is what the outside design temperature is. The outside design temperature for my area (Atlanta, Georgia) is 23 degrees. The outside design temperatures for other areas are tabulated in Table 1. For a more complete list, consult one of the references listed at the end of this article.

Actually, you do not need to put any specific temperature in here as long as it is less than 75 degrees, the inside design temperature used by the program. The program will still give you valid results for savings and payback. However, using the correct outside design temperature gives you the advantage of seeing what the furnace size would be for your house with and without the improvements. In fact, heating engineers use the same basic method as this program does to size furnaces for houses.

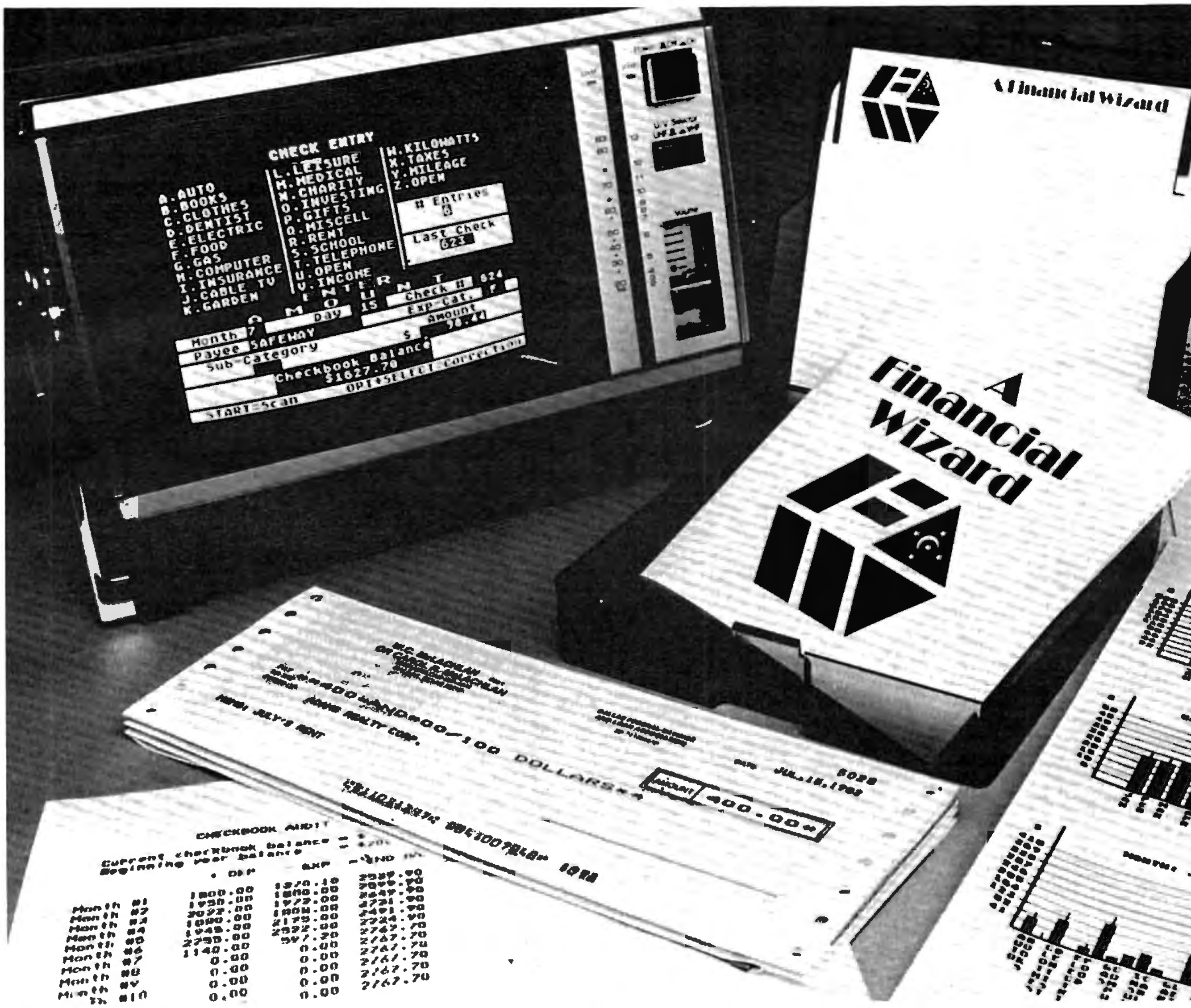
When the program finishes calculating the heat loss of the house after improvements, it is ready to do the cost analysis. First you are asked for the type of heating fuel you use: electricity,

Table 1: Winter Design Temperatures

CITY	TEMPERATURE
MONTGOMERY AL	26
JUNEAU AK	-4
PHOENIX AZ	34
LITTLE ROCK AR	23
SACRAMENTO CA	32
DENVER CO	3
HARTFORD CONN	5
DOVER DEL	15
TALLAHASSEE FL	29
ATLANTA GA	23
HONOLULU HI	62
BOISE ID	10
SPRINGFIELD IL	4
INDIANAPOLIS IN	4
DES MOINES IA	-3
TOPEKA KS	6
LEXINGTON KY	10
BATON ROUGE LA	30
AUGUSTA ME	-3
BALTIMORE MD	20
BOSTON MA	10
LANSING MI	6
ST. PAUL MN	-10
JACKSON MS	24
JEFFERSON CITY MO	6
HELENA MT	-13
LINCOLN NE	0
CARSON CITY NV	7
CONCORD NH	-7
TRENTON NJ	16
SANTA FE NM	11
ALBANY NY	5
RALEIGH NC	20
BISMARCK ND	-19
COLUMBUS OH	7
OKLAHOMA CITY OK	15
SALEM OR	25
HARRISBURG PA	13
PROVIDENCE RI	10
COLUMBIA SC	23
PIERRE SD	-9
NASHVILLE TN	16
AUSTIN TX	29
SALT LAKE CITY UT	9
BURLINGTON VT	-7
RICHMOND VA	18
OLYMPIA WA	25
CHARLESTON WV	14
MADISON WS	-5
CHEYENNE WY	-2

1983 - Take Control

A FINANCIAL WIZARD 1.5



THE HIGHLY ACCLAIMED PERSONAL FINANCE SYSTEM FOR ATARI*

- **Budget Forecast**–26 expense categories
- **Check Entry**–easy data entry–scan & modify 26 major & 36 sub-categories–information block
- **Check Search**–single or multiple parameters–(up to seven) to search entries
- **Tabulations**–detailed expense vs. budget comparisons by month, year-to-date, category
- **Bar Graphs**–screen displays in graph form expenses vs. budget–by month or category–printing with graphic capable printers
- **Check Reconciliation**–fast clearing of resident checks & deposits, complete summary report
- **Checkwriter**–prints your custom checks
- **Printouts**–most popular printers • **Audit Report**
- **Multi-Colored Graphics • Audio Enhancements**
- **7 Utility Programs • User-Friendly Operation**
- **Easy To Use Instruction Manual**
- **Handsome Tinted Plastic Storage Case**

The system is designed for Atari computers having a minimum of 24K and operating from a disk drive. The price is only $59.95 plus $3 for handling/postage.
If your dealer does not have **A Financial Wizard** . . .
Telephone orders are accepted on Mastercharge or Visa credit cards. Mail order must be accompanied by check or money-order or credit card #.
Dealer Inquiries invited. (405)751-2783.

*trademark of Atari Inc. OK residents ad 4% Tax.

A Financial Wizard Exclusively thru ON LINE COMPUTER CENTERS of OKC™
10944 North May
Oklahoma City, OK 73120
(405) 751-2781

☐ CHECK ENCLOSED ☐ VISA ☐ MASTERCHARGE

NAME (Print) ____________________

Address ____________________

City ____________ State ________ Zip ________

Card # ____________ Exp. ________ Sig. ________

fuel oil, or natural gas. Next you must input the cost per fuel unit of the heating fuel.

Note that this unit cost is in dollars, so if natural gas in your area is 35 cents per therm, you should input .35 dollars per therm.

Using this data and the heating degree days, the program calculates the total energy needed to heat the house for the entire heating season. The degree days and name of the city are on line 7010. You should change this line to reflect your own location. Some sample degree days for different cities are listed in Table 2, and a more complete list can be found in any of the references. The last thing you must input is the total cost of the improvements you made. From this data the program calculates the payback period in years.

I got pretty popular in my neighborhood when word got out that my home computer could calculate how cost effective it would be to add insulation. I have also learned a great deal about my own home from running this program. Much of what I concluded was what I expected, but some conclusions surprised me. The program can definitely help home owners in assessing home energy improvements; it can also enable a home owner to spot dishonest "energy-saving" schemes pretty quickly.

Table 2: Yearly Heating Degree Days

CITY	DEGREE DAYS
MONTGOMERY AL	2291
JUNEAU AK	9075
PHOENIX AZ	1765
LITTLE ROCK AR	3219
SACRAMENTO CA	2419
DENVER CO	5524
HARTFORD CONN	6235
WILMINGTON DEL	4930
TALLAHASSEE FL	1485
ATLANTA GA	2961
HONOLULU HI	0
BOISE ID	5809
SPRINGFIELD IL	5429
INDIANAPOLIS IN	5699
DES MOINES IA	6588
TOPEKA KS	5182
LEXINGTON KY	4683
BATON ROUGE LA	1560
PORTLAND ME	7511
BALTIMORE MD	4111
BOSTON MA	5634
LANSING MI	6909
MINNEAPOLIS MN	8382
JACKSON MS	2239
ST. LOUIS MO	4484
HELENA MT	8129
LINCOLN NE	5864
RENO NV	6332
CONCORD NH	7383
TRENTON NJ	4980
ALBUQUERQUE NM	4348
ALBANY NY	6201
RALEIGH NC	3393
BISMARCK ND	8851
COLUMBUS OH	5211
OKLAHOMA CITY OK	3725
SALEM OR	4754
HARRISBURG PA	5251
PROVIDENCE RI	5954
COLUMBIA SC	2484
RAPID CITY SD	7345
NASHVILLE TN	3578
AUSTIN TX	1711
SALT LAKE CITY UT	6052
BURLINGTON VT	8269
RICHMOND VA	3865
OLYMPIA WA	5236
CHARLESTON WV	4476
MADISON WS	7863
CHEYENNE WY	7381

References

1. *ASHRAE Handbook 1981 Fundamentals*. Atlanta, Georgia: American Society of Heating, Refrigerating and Air-conditioning Engineers, Incorporated, 1981.

2. *Other Homes and Garbage*, Jim Leckie, Gil Masters, Harry Whitehouse, and Lilly Young. San Francisco, California: Sierra Club Books, 1975.

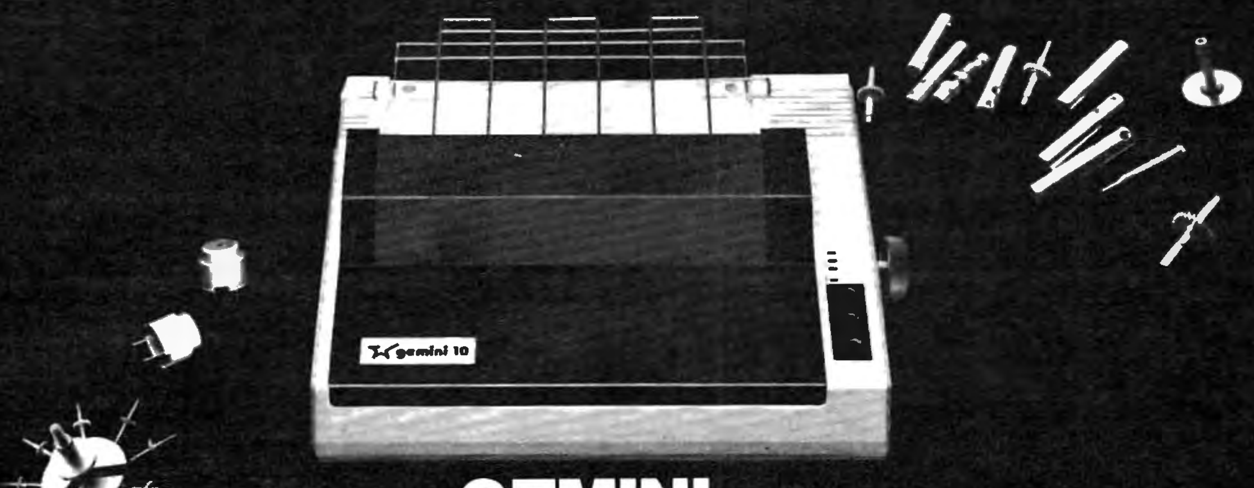
3. *Refrigeration and Air-Conditioning*, Air-Conditioning and Refrigeration Institute. Englewood Cliffs, New Jersey: Prentice-Hall, 1979.

Program 1: Microsoft BASIC

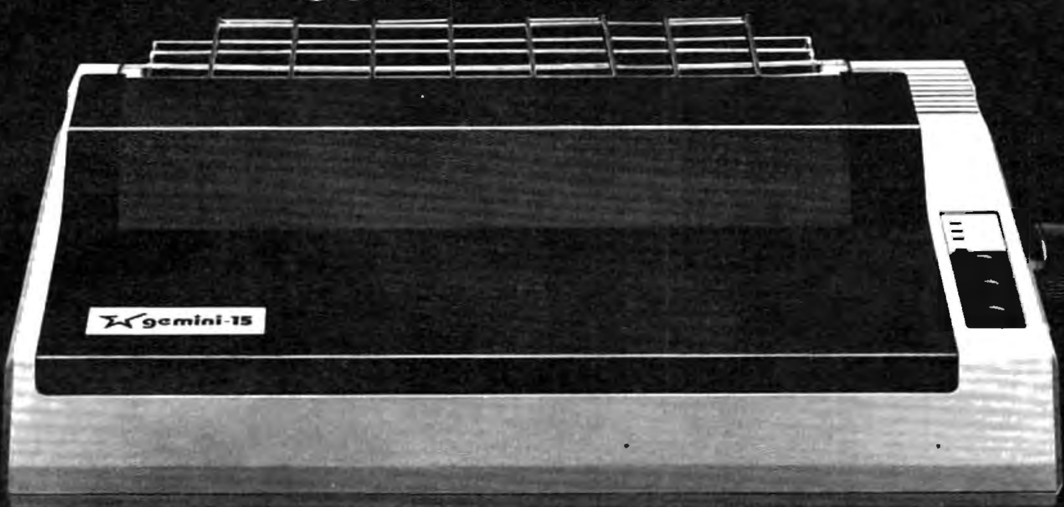
```
100 PRINT"{CLEAR}{02 DOWN}  HOME EN
    ERGY PROGRAM
110 PRINT:PRINT
120 PRINT"  BY DAVID SWAIM
130 PRINT"      P. O. BOX 720126
140 PRINT"      ATLANTA, GEORGIA 303
    58
150 GOSUB 8000
160 REM COPYRIGHT 1981 DAVID C. SWA
    IM II
170 REM
180 DIM A(6),Q(6),R(6),RW(4,3),D(4)
    ,IW(2,3),S(10)
190 DIM RF(3),TC(3),N$(5),IC(5),DM(
    2,3,3),IN(2)
200 REM WINDOW R VALUES
210 DATA 1.01,2.22,1.815,3.155
220 DATA .909,1.667,1.437,2.137
230 DATA .909,2,1.724,2.564
240 REM DOOR R VALUES
250 DATA .41,.75,.95,1.1
260 REM FLOOR R VALUES AND TEMP COR
    R
270 DATA 3.2,0,3.2,30,1.23,0
280 REM CEILING INSULATION R PER IN
    CH
290 DATA 3.5,3,2.5,4.5,5.5
300 N$(1)="WINDOWS":N$(2)="DOORS":N
    $(3)="WALLS"
310 N$(4)="CEILING":N$(5)="FLOOR *"
320 REM DUCT MULTIPLIERS
330 DATA .2,.15,.1,.15,.1,.05,.1,.0
    5,.05
```

```
340 DATA .2,.15,.1,.1,.1,.05,.05,.0
    5,.05
350 REM AIR CHANGES PER FOOT OF CRA
    CK
360 DATA 39,74,52,24,32,33
370 REM READ WINDOW R VALUES
380 FOR F=1 TO 3
390 FOR G=1 TO 4
400 READ RW(G,F)
410 NEXT G,F
420 REM READ DOOR R VALUES
430 FOR I=1 TO 4:READ D(I):NEXT I
440 REM READ FLOOR R VAL AND TEMP C
    ORR
450 FOR I=1 TO 3:READ RF(I),TC(I):N
    EXT I
460 REM READ INSULATION R PER INCH
470 FOR I=1 TO 5:READ IC(I):NEXT I
480 REM READ DUCT MULTIPLIERS
490 FOR KD=1 TO 2
500 FOR K=1 TO 3
510 FOR J=1 TO 3
520 READ DM(KD,J,K)
530 NEXT J,K,KD
540 REM READ AIR CHANGES FOR INFILT
    RATION
550 FOR I=1 TO 2
560 FOR J=1 TO 3
570 READ IW(I,J)
580 NEXT J,I
590 REM INSIDE DESIGN TEMPERATURE
600 IT=75:PK=1
605 GETA$:IFA$=""THEN605
610 PRINT"{CLEAR}{DOWN}WINTER OUTSI
    DE DESIGN TEMPERATURE";
620 INPUT OT
630 DT=IT-OT
640 GOSUB 1000:REM   WINDOWS
650 GOSUB 2000:REM   DOORS
660 GOSUB 3000:REM   WALLS
670 GOSUB 4000:REM   CEILING
680 GOSUB 5000:REM   FLOOR
690 GOSUB 5200:REM   DUCTS
700 GOSUB 6000:REM   REPORT RESULTS
710 Q1=TQ/DT
720 PRINT"{DOWN}DO YOU WISH TO MAKE
     IMPROVEMENTS?"
730 GET A$:IF A$="" THEN 730
740 PK=2:IF A$="N" THEN 999
750 INPUT"{CLEAR}{02 DOWN}DO YOU WI
    SH TO IMPROVE WINDOWS";A$
760 IF LEFT$(A$,1)="Y" THEN GOSUB 1
    000
770 INPUT"{CLEAR}{02 DOWN}DO YOU WI
    SH TO IMPROVE DOORS";A$
780 IF LEFT$(A$,1)="Y" THEN GOSUB 2
    000
790 INPUT"{CLEAR}{02 DOWN}DO YOU WI
    SH TO IMPROVE WALLS";A$
800 IF LEFT$(A$,1)="Y" THEN GOSUB 3
    000
810 INPUT"{CLEAR}{02 DOWN}DO YOU WI
    SH TO IMPROVE CEILING";A$
820 IF LEFT$(A$,1)="Y" THEN GOSUB 4000
830 INPUT"{CLEAR}{02 DOWN}DO YOU WI
    SH TO IMPROVE FLOOR";A$
840 IF LEFT$(A$,1)="Y" THEN GOSUB 5
    000
850 INPUT"{CLEAR}{02 DOWN}DO YOU WI
    SH TO IMPROVE DUCTS";A$
860 IF LEFT$(A$,1)="Y" THEN GOSUB 5
    200
870 GOSUB 6000:REM REPORT RESULTS
880 Q2=TQ/DT
890 PRINT:PRINT"HIT RETURN TO GET S
    AVINGS"
900 GET A$:IF A$="" THEN 900
910 GOSUB 7000:REM CALCULATE A YEAR
     OF SAVINGS
999 END
1000 REM WINDOW SUBROUTINE
1010 I=1:IF PK>1 THEN 1040
1020 PRINT"{CLEAR}{DOWN}HOW MANY DIF
    FERENT TYPES OF WINDOWS";
1030 INPUT NX
1040 IX=1:CW=0:A(I)=0:Q(I)=0
1050 PRINT"{DOWN} ARE WINDOWS WEATHE
    RSTRIPPED";
1060 INPUT WW$
1070 IF LEFT$(WW$,1)="Y" THEN IX=2
1080 FOR J=1 TO NX
1090 PRINT"SIZE";J:IF PK>1 THEN 1160
1100 PRINT"NUMBER OF WINDOWS";
1110 INPUT NW
1120 PRINT"SIZE OF WINDOWS (H,W) FT"
    ;
1130 INPUT H,W
1140 S(J)=H*W*NW
1150 CW=CW+(H+W)*NW
1160 A(I)=A(I)+S(J)
1170 PRINT"TYPE OF WINDOWS"
1180 PRINT"   1. SINGLE GLASS"
1190 PRINT"   2. SINGLE + STORM"
1200 PRINT"   3. DOUBLE PANE"
1210 PRINT"   4. TRIPLE (DOUBLE + ST
    ORM)"
1220 INPUT G
1230 PRINT"TYPE OF WINDOW FRAME"
1240 PRINT"   1. WOOD"
1250 PRINT"   2. METAL OR JALOUSE"
1260 PRINT"   3. FIXED"
1270 INPUT F
1280 RM=RW(G,F)
1290 Q(I)=Q(I)+S(J)*DT/RM
1300 R(I)=RM
1310 PRINT"{CLEAR}{DOWN}";
1320 NEXT J
1330 IN(I)=0.018*DT*IW(IX,F)*CW
1340 RETURN
2000 REM DOORS SUBROUTINE
2010 I=2:IF PK>1 THEN 2080
2020 PRINT"{CLEAR}{DOWN}NUMBER OF DO
    ORS";
2030 INPUT N
2040 PRINT"SIZE OF DOORS (H,W) FT";
2050 INPUT H,W
2060 A(I)=H*W*N
2070 CD=(H+W)*N
```

```
2080 PRINT"{DOWN}TYPE OF DOORS"
2090 PRINT"    1. WOOD"
2100 PRINT"    2. WOOD + STORM"
2110 PRINT"    3. METAL URETHANE CORE
     "
2120 PRINT"    4. METAL POLYSTYRENE C
     ORE"
2130 INPUT T
2140 R(I)=D(T)
2150 Q(I)=A(I)*DT/R(I)
2160 DW=138
2170 PRINT"{DOWN}ARE DOORS WEATHERST
     RIPPED";
2180 INPUT DW$
2190 IF LEFT$(DW$,1)="Y" THEN DW=69
2200 IN(I)=0.018*DT*DW*CD
2210 RETURN
3000 REM WALLS SUBROUTINE
3010 I=3:HO=.17:HI=.68
3020 PRINT"{CLEAR}{DOWN}TYPE OF WALL
      CONSTRUCTION"
3030 PRINT"{DOWN}    1. BRICK VENEER"
3040 PRINT "    2. STONE"
3050 PRINT "    3. WOOD SHINGLES"
3060 PRINT "    4. STUCCO"
3070 PRINT "    5. MASONRY BLOCK"
3080 PRINT "    6. LOG"
3090 PRINT "    7. OTHER:"
3100 PRINT "       ENTER CALCULATED R
      VALUE DIRECTLY"
3110 PRINT "       WHEN ASKED FOR INS
     ULATION R VALUE"
3120 INPUT TY
3130 ON TY GOTO 3140,3150,3160,3170,
     3180,3190,3200
3140 RM=.2*3.5:GOTO 3210:REM  BRICK
3150 RM=.08*5: GOTO 3210:REM  STONE
3160 RM=.87:   GOTO 3210:REM  WOOD
3170 RM=.2*2:  GOTO 3210:REM  STUCCO
3180 RM=2:     GOTO 3210:REM  MASONR
     Y
3190 RM=1.25*8:GOTO 3210:REM  LOG
3200 RM=0:REM                 OTHER
3210 PRINT"   FOR LIST OF R VALUES F
     OR INSULATION"
3220 PRINT"   ENTER -1 FOR INSULATIO
     N R VALUE"
3230 PRINT"INSULATION R VALUE";
3240 INPUT RI
3250 IF RI<0 THEN GOSUB 3500:GOTO 32
     30
3260 R(I)=HO+RM+RI+HI:IF PK>1 THEN 3
     340
3270 PRINT"HOW MANY STORIES IN HOUSE
     ";
3280 INPUT NT
3290 PRINT"WHAT IS THE CEILING HEIGH
     T (FT)";
3300 INPUT CH
3310 PRINT"WHAT IS TOTAL PERIMETER (
     FT)";
3320 INPUT P
3330 A(I)=NT*CH*P-A(1)-A(2)
3340 Q(I)=A(I)*DT/R(I)
3350 RETURN
3500 REM LIST OF INSULATION R VALUES
3510 PRINT"{CLEAR}{DOWN}LIST OF INSU
     LATION R VALUES, WALLS"
3520 PRINT"{DOWN}           NO INSULATI
     ON (AIR) = .94"
3530 PRINT"        BATT INSULATION IN WA
     LL = 11"
3540 PRINT"        HALF INCH ASPHALT BOA
     RD = 2.4"
3550 PRINT"      1/2 IN GYPSUM OR PLAST
     ER = 1.39"
3560 PRINT"        1/4 IN WOOD FIBER BOA
     RD = 1.12"
3570 PRINT"          FIR OR PINE SHEATHI
     NG = 1.92"
3580 PRINT"          3/4 IN PLYWOOD PANE
     LS = 1.88"
3590 PRINT"                 1/2 IN PLYWO
     OD = 1.57"
3600 PRINT:PRINT
3610 RETURN
4000 REM CEILING ROUTINE
4010 I=4
4020 HI=.61:HO=.61:IF PK>1 THEN 4060
4030 PRINT"{CLEAR}{DOWN}WHAT IS TOTA
     L CEILING AREA"
4040 PRINT"OF THE HOUSE";
4050 INPUT A(I)
4060 PRINT"HOW MANY INCHES OF INSULA
     TION IN CEILING";
4070 INPUT CI
4080 PRINT"TYPE OF INSULATING MATERI
     AL"
4090 PRINT"{DOWN}  1. FIBERGLASS"
4100 PRINT "  2. MINERAL WOOL"
4110 PRINT "  3. VERMICULITE OR PERL
     ITE"
4120 PRINT "  4. CELLULOSE FIBER"
4130 PRINT "  5. U-F FOAM{DOWN}"
4140 INPUT T
4150 RM=CI*IC(T)
4160 R(I)=HO+RM+HI
4170 Q(I)=A(I)*DT/R(I)
4180 RETURN
5000 REM FLOOR ROUTINE
5010 I=5:IF PK>1 THEN 5040
5020 PRINT"{CLEAR}{DOWN}WHAT IS TOTA
     L FLOOR AREA";
5030 INPUT A(I)
5040 PRINT"HOW MANY INS OF INSULATIO
     N IN FLOOR";
5050 INPUT FI:IF PK>1 THEN 5110
5060 PRINT"TYPE OF FOUNDATION"
5070 PRINT"   1. OPEN CRAWLSPACE"
5080 PRINT"   2. ENCLOSED CRAWLSPACE
      OR BASEMENT"
5090 PRINT"   3. CONCRETE SLAB"
5100 INPUT TF
5110 R(I)=HO+FI*3.1+RF(TF)+HI
5120 Q(I)=A(I)*(DT-TC(TF))/R(I)
5130 RETURN
5200 REM DUCTS
5210 DI=.1
```



GEMINI–
FOR PRINTER VALUE THAT'S OUT OF THIS WORLD



Over thirty years of down-to-earth experience as a precision parts manufacturer has enabled Star to produce the Gemini series of dot matrix printers—a stellar combination of printer quality, flexibility, and reliability. And for a list price of nearly 25% less than the best selling competitor.

The Gemini 10 has a 10″ carriage and the Gemini 15 a 15½″ carriage. Plus, the Gemini 15 has the added capability of a bottom paper feed. In both models, Gemini quality means a print speed of 100 cps, high-resolution bit image and block graphics, and extra fast forms feed.

Gemini's flexibility is embodied in its diverse specialized printing capabilities such as super/sub script, underlining, back-spacing, double strike mode and emphasized print mode. Another extraordinary standard feature is a 2.3K buffer. An additional 4K is optional. That's twice the memory of leading, comparable printers. And Gemini is compatible with most software packages that support the leading printers.

Gemini reliability is more than just a promise. It's as concrete as a 180 day warranty (90 days for ribbon and print head), a mean time between failure rate of 5 million lines, a print head life of over 100 million characters, and a 100% duty cycle that allows the Gemini to print continuously. Plus, prompt, nationwide service is readily available.

So if you're looking for an incredibly high-quality, low-cost printer that's out of this world, look to the manufacturer with its feet on the ground—Star and the Gemini 10, Gemini 15 dot matrix printers.

star®
micronics•inc

MAKING A NAME FOR OURSELVES

1120 Empire Central Place, Suite 216, Dallas, TX 75247
For more information, please call Bob Hazzard, Vice President, at (214) 631-8560.

```
5220 IF TF=3 THEN KD=3:RETURN
5230 PRINT"{DOWN}IS YOUR DUCTWORK IN
    SULATED";
5240 INPUT D$:IF PK>1 THEN 5310
5250 PRINT"{DOWN}LOCATION OF HEAT DU
    CTS:"
5260 PRINT"     1. ATTIC OR CRAWLSPAC
    E"
5270 PRINT"     2. UNCONDITIONED BASE
    MENT"
5280 PRINT"     3. IN SLAB FLOOR"
5290 PRINT"     4. INSIDE CONDITIONED
     SPACE"
5300 INPUT KD
5310 RETURN
6000 REM WRITE A REPORT
6010 PRINT"{CLEAR}","HEAT LOSS EVALU
    ATION"
6020 PRINT:PRINT:TQ=0
6030 PRINT" ITEM"," AREA"," R-VALUE"
    ,"HEAT LOSS"
6040 PRINT ," SQ.FT.",," BTU/HR":PRI
    NT
6050 FOR I=1 TO 5
6060 A(I)=INT(A(I)*100+.5)/100
6070 R(I)=INT(R(I)*100+.5)/100
6080 Q(I)=INT(Q(I)+.5)
6090 PRINT N$(I),A(I),R(I),Q(I)
6100 TA=TA+A(I):TQ=TQ+Q(I)
6110 NEXT I
6120 REM PRINT INFILTRATION LOSS
6130 PRINT"INFILTRATION",,INT((IN(1)
    +IN(2))/2+.5)
6140 TQ=TQ+(IN(1)+IN(2))/2
6150 REM CALCULATE DUCT LOSS
6160 X=TQ/(A(5)*CH*NT):J=3:K=3
6170 IF X<45 THEN K=2
6180 IF X<35 THEN K=1
6190 DI=.15+.05*(3-K)
6200 IF LEFT$(D$,1)="N" AND KD<2 THE
    N 6240
6205 IF KD>2 THEN DI=0:GOTO 6240
6210 IF OT<15 THEN J=2
6220 IF OT<0 THEN J=1
6230 DI=DM(KD,J,K)
6240 PRINT"DUCT LOSS",,,INT(DI*TQ+.5)
6250 TQ=TQ+TQ*DI
6260 PRINT ,"@@@@@@@@",,"@@@@@@@@"
6270 PRINT"  TOTAL",INT(TA),,INT(TQ)
6280 PRINT:PRINT
6290 PRINT"DESIGN CONDITIONS:"
6300 PRINT"   OUTSIDE DESIGN TEMP";O
    T
6310 PRINT"    INSIDE DESIGN TEMP";I
    T
6320 PRINT"TEMPERATURE DIFFERENCE";D
    T
6330 RETURN
7000 REM FIND SAVINGS USING DEGREE-D
    AYS
7010 DD=2961:DD$="ATLANTA GA"
7012 E1=INT(Q1*DD*24)
7014 E2=INT(Q2*DD*24)
7030 PRINT"{CLEAR}TYPE OF HEATING FU
    EL USED"
7040 PRINT"  1. ELECTRICITY"
7050 PRINT"  2. NATURAL GAS"
7060 PRINT"  3. FUEL OIL"
7070 INPUT FT:PC=.55
7080 ON FT GOTO 7100,7200,7300
7090 GOTO 7030
7100 REM ELECTRICITY
7110 PRINT"IS HEATING UNIT A HEAT PU
    MP";
7120 INPUT HP$:ER=3413
7130 IF LEFT$(HP$,1)<>"Y" THEN 7150
7140 INPUT"ENTER EER OF HEAT PUMP";E
    R:ER=ER*1000
7150 INPUT"AVERAGE $ COST PER KWH";C
    O:FU$="KWH"
7160 E1=INT(E1/ER+.5)
7165 M1=E1*CO
7170 E2=INT(E2/ER+.5)
7175 M2=E2*CO
7180 MS=M1-M2
7190 GOTO 7400
7200 REM NATURAL GAS
7210 INPUT"AVERAGE $ COST PER THERM
    OF NATURAL GAS";CO
7220 E1=INT(E1/(103000*PC)+.5)
7225 M1=E1*CO
7230 E2=INT(E2/(103000*PC)+.5)
7235 M2=E2*CO
7240 MS=M1-M2
7250 FU$="THERMS":GOTO 7400
7300 REM FUEL OIL
7310 INPUT"AVERAGE $ COST PER GALLON
     OF FUEL OIL";CO
7320 E1=INT(E1/(138000*PC)+.5)
7325 M1=E1*CO
7330 E2=INT(E2/(138000*PC)+.5)
7335 M2=E2*CO
7340 MS=M1-M2:FU$="GALLONS"
7400 REM GIVE RESULTS
7410 M1=INT(M1*100)/100
7420 M2=INT(M2*100)/100
7430 MS=INT(MS*100)/100
7440 INPUT"{DOWN}TOTAL $ COST OF YOU
    R IMPROVEMENTS";CI
7450 PB=INT(CI/MS*1000)/1000
7460 REM REPORT SAVINGS AND PAYBACK
7470 PRINT"{CLEAR}","ANALYSIS OF IMP
    ROVEMENTS"
7480 PRINT:PRINT
7490 PRINT,,"ENERGY NEEDED"
7500 PRINT"ORIGINAL HOUSE",E1;FU$
7510 PRINT"IMPROVED HOUSE",E2;FU$
7520 PRINT,,"@@@@@@@@@"
7530 PRINT,"SAVINGS",E1-E2;FU$
7540 PRINT
7550 PRINT,,"OPER. COSTS"
7560 PRINT"ORIGINAL HOUSE","$";M1
7570 PRINT"IMPROVED HOUSE","$";M2
7580 PRINT,,"@@@@@@@@@"
7590 PRINT,"SAVINGS","$";MS
7600 PRINT:PRINT,"PAYBACK",PB;"YEARS
    "
7610 PRINT:PRINT
```

```
7620 PRINT"ABOVE IS BASED ON ONE YEA
    R OF OPERATION"
7630 PRINT"IN ";DD$
7640 RETURN
8000 REM DRAW HOUSE
8010 PRINTCHR$(142):PRINT:PRINT:PRIN
    T
8020 PRINTSPC(8);"           ,;"
8030 PRINTSPC(8);"{REV})            _{
    OFF}"
8040 PRINTSPC(8);"T,;,;,;,;,;,;Y
8050 PRINTSPC(8);"T<><><><><>*{REV}
       _{OFF}"
8060 PRINTSPC(8);"T,;,;,;,;,;,;Y,""";Y
    "
8070 PRINTSPC(8);"T<><>{REV}!{OFF}!<
    ><>Y{REV}!    {OFF}!Y"
8080 PRINTSPC(7);"##################
    ###"
8090 RETURN
```

Program 2:

Make these changes in Program 1 for the Apple II.

```
100  HOME : VTAB 2: PRINT "  HOME
      ENERGY PROGRAM"
3510  HOME : PRINT "LIST OF INSUL
     ATION R VALUES, WALLS"
3520  PRINT : PRINT "        NO INS
     ULATION
4030  HOME : PRINT "WHAT IS TOTAL
      CEILING AREA
4090  PRINT : PRINT "  1. FIBERGL
     ASS
4130  PRINT : PRINT "  5. U-F FOA
     M": PRINT
5020  HOME : PRINT "WHAT IS TOTAL
      FLOOR AREA?";
5230  PRINT : PRINT "IS YOUR DUCT
     WORK INSULATED?";
5250  PRINT "LOCATION OF HEAT DUC
     TS:
6010  HOME : PRINT "HEAT LOSS EVA
     LUATION"
7440  PRINT : INPUT "TOTAL $ COST
      OF YOUR IMPROVEMENTS";CI
7470  HOME : PRINT "ANALYSIS OF I
     MPROVEMENTS"
7520  PRINT ,,"---------"
7580  PRINT ,,"---------"
8000  RETURN
8010  - 8090"DELETE"
```

Program 3: Atari Version

```
100 POKE 82,0:PRINT "{CLEAR}{2 DOWN}
     HOME ENERGY PROGRAM"
110 PRINT :PRINT
150 GOSUB 8000
170 OPEN #1,4,0,"K:"
180 DIM A(6),Q(6),R(6),RW(4,3),D(4),I
    W(2,3),S(10)
190 DIM RF(3),TC(3),N$(5*10),IC(5),DM
    (2,15),IN(2),A$(1),NL(5)
191 DIM WW$(1),DW$(1),D$(1),DD$(20),H
    P$(1),FU$(10)
200 REM WINDOW R VALUES
210 DATA 1.01,2.22,1.815,3.155
220 DATA .909,1.667,1.437,2.137
230 DATA .909,2,1.724,2.564
240 REM DOOR R VALUES
250 DATA .41,.75,.95,1.1
260 REM FLOOR R VALUES AND TEMP CORR
270 DATA 3.2,0,3.2,30,1.23,0
280 REM CEILING INSULATION R PER INCH
290 DATA 3.5,3,2.5,4.5,5.5
300 N$(1)="WINDOWS":N$(11)="DOORS":N$
    (21)="WALLS"
310 N$(31)="CEILING":N$(41)="FLOOR *"
315 NL(1)=7:NL(2)=5:NL(3)=5:NL(4)=6:N
    L(5)=7
320 REM DUCT MULTIPLIERS
330 DATA .2,.15,.1,.15,.1,.05,.1,.05,
    .05
340 DATA .2,.15,.1,.1,.1,.05,.05,.05,
    .05
350 REM AIR CHANGES PER FOOT OF CRACK
360 DATA 39,74,52,24,32,33
370 REM READ WINDOW R VALUES
380 FOR F=1 TO 3
390 FOR G=1 TO 4
400 READ TEMP:RW(G,F)=TEMP
410 NEXT G:NEXT F
420 REM READ DOOR R VALUES
430 FOR I=1 TO 4:READ TEMP:D(I)=TEMP:
    NEXT I
440 REM READ FLOOR R VAL AND TEMP COR
    R
450 FOR I=1 TO 3:READ TEMP:RF(I)=TEMP
    :READ TEMP:TC(I)=TEMP:NEXT I
460 REM READ INSULATION R PER INCH
470 FOR I=1 TO 5:READ TEMP:IC(I)=TEMP
    :NEXT I
480 REM READ DUCT MULTIPLIERS
490 FOR KD=1 TO 2
500 FOR K=1 TO 3
510 FOR J=1 TO 3
520 READ TEMP:DM(KD,J+K*4)=TEMP
530 NEXT J:NEXT K:NEXT KD
540 REM READ AIR CHANGES FOR INFILTRA
    TION
550 FOR I=1 TO 2
560 FOR J=1 TO 3
570 READ TEMP:IW(I,J)=TEMP
580 NEXT J:NEXT I
590 REM INSIDE DESIGN TEMPERATURE
600 IT=75:PK=1
601 ? :? :? "Press RETURN to begin:";
605 GET #1,A
610 PRINT "{CLEAR}{DOWN}WINTER OUTSID
    E DESIGN TEMPERATURE";
620 INPUT OT
630 DT=IT-OT
640 GOSUB 1000:REM WINDOWS
650 GOSUB 2000:REM DOORS
660 GOSUB 3000:REM WALLS
670 GOSUB 4000:REM CEILING
680 GOSUB 5000:REM FLOOR
690 GOSUB 5200:REM DUCTS
700 GOSUB 6000:REM REPORT RESULTS
710 Q1=TQ/DT
720 PRINT "DO YOU WISH TO MAKE IMPROV
    EMENTS?:";
730 GET #1,A:A$=CHR$(A)
740 K=2:IF A$="N" THEN 999
750 PRINT "{CLEAR}{2 DOWN}DO YOU WISH
     TO IMPROVE WINDOWS";:INPUT A$
760 IF A$="Y" THEN GOSUB 1000
770 PRINT "{CLEAR}{2 DOWN}DO YOU WISH
     TO IMPROVE DOORS";:INPUT A$
780 IF A$="Y" THEN GOSUB 2000
790 PRINT "{CLEAR}{2 DOWN}DO YOU WISH
```

```
    TO IMPROVE WALLS";:INPUT A$
800 IF A$="Y" THEN GOSUB 3000
810 PRINT "{CLEAR}{2 DOWN}DO YOU WISH
    TO IMPROVE CEILING";:INPUT A$
820 IF A$="Y" THEN GOSUB 4000
830 PRINT "{CLEAR}{2 DOWN}DO YOU WISH
    TO IMPROVE FLOOR";:INPUT A$
840 IF A$="Y" THEN GOSUB 5000
850 PRINT "{CLEAR}{2 DOWN}DO YOU WISH
    TO IMPROVE DUCTS";:INPUT A$
860 IF A$="Y" THEN GOSUB 5200
870 GOSUB 6000:REM REPORT RESULTS
880 Q2=TQ/DT
890 PRINT :PRINT "HIT RETURN TO GET S
    AVINGS"
900 GET #1,A
910 GOSUB 7000:REM CALCULATE A YEAR O
    F SAVINGS
999 END
1000 REM WINDOW SUBROUTINE
1010 I=1:IF PK>1 THEN 1040
1020 PRINT "{CLEAR}{DOWN}HOW MANY DIF
     FERENT TYPES OF WINDOWS";
1030 INPUT NX
1040 IX=1:CW=0:A(I)=0:Q(I)=0
1050 PRINT "{DOWN} ARE WINDOWS WEATHE
     RSTRIPPED";
1060 INPUT WW$
1070 IF WW$="Y" THEN IX=2
1080 FOR J=1 TO NX
1090 PRINT "SIZE ";J:IF PK>1 THEN 116
     0
1100 PRINT "NUMBER OF WINDOWS";
1110 INPUT NW
1120 PRINT "SIZE OF WINDOWS (H,W) FT"
     ;
1130 INPUT H,W
1140 S(J)=H*W*NW
1150 CW=CW+(H+W)*NW
1160 A(I)=A(I)+S(J)
1170 PRINT "TYPE OF WINDOWS"
1180 PRINT "{3 SPACES}1. SINGLE GLASS
     "
1190 PRINT "{3 SPACES}2. SINGLE + STO
     RM"
1200 PRINT "{3 SPACES}3. DOUBLE PANE"
1210 PRINT "{3 SPACES}4. TRIPLE (DOUB
     LE + STORM)"
1220 INPUT G
1230 PRINT "TYPE OF WINDOW FRAME"
1240 PRINT "{3 SPACES}1. WOOD"
1250 PRINT "{3 SPACES}2. METAL OR JAL
     OUSE"
1260 PRINT "{3 SPACES}3. FIXED"
1270 INPUT F
1280 RM=RW(G,F)
1290 Q(I)=Q(I)+S(J)*DT/RM
1300 R(I)=RM
1310 PRINT "{CLEAR}{DOWN}";
1320 NEXT J
1330 IN(I)=0.018*DT*IW(IX,F)*CW
1340 RETURN
2000 REM DOORS SUBROUTINE
2010 I=2:IF PK>1 THEN 2080
2020 PRINT "{CLEAR}{DOWN}NUMBER OF DO
     ORS";
2030 INPUT N
2040 PRINT "SIZE OF DOORS (H,W) FT";
2050 INPUT H,W
2060 A(I)=H*W*N
2070 CD=(H+W)*N
2080 PRINT "{DOWN}TYPE OF DOORS"
2090 PRINT "{3 SPACES}1. WOOD"
2100 PRINT "{3 SPACES}2. WOOD + STORM
     "
2110 PRINT "{3 SPACES}3. METAL URETHA
     NE CORE"
2120 PRINT "{3 SPACES}4. METAL POLYST
     YRENE CORE"
2130 INPUT T
2140 R(I)=D(T)
2150 Q(I)=A(I)*DT/R(I)
2160 DW=138
2170 PRINT "{DOWN}ARE DOORS WEATHERST
     RIPPED";
2180 INPUT DW$
2190 IF DW$="Y" THEN DW=69
2200 IN(I)=0.018*DT*DW*CD
2210 RETURN
3000 REM WALLS SUBROUTINE
3010 I=3:HO=0.17:HI=0.68
3020 PRINT "{CLEAR}{DOWN}TYPE OF WALL
      CONSTRUCTION"
3030 PRINT "{DOWN}{3 SPACES}1. BRICK
     VENEER"
3040 PRINT "{3 SPACES}2. STONE"
3050 PRINT "{3 SPACES}3. WOOD SHINGLE
     S"
3060 PRINT "{3 SPACES}4. STUCCO"
3070 PRINT "{3 SPACES}5. MASONRY BLOC
     K"
3080 PRINT "{3 SPACES}6. LOG"
3090 PRINT "{3 SPACES}7. OTHER:"
3100 PRINT "{6 SPACES}ENTER CALCULATE
     D R VALUE DIRECTLY"
3110 PRINT "{6 SPACES}WHEN ASKED FOR
     INSULATION R VALUE"
3120 INPUT TY
3130 ON TY GOTO 3140,3150,3160,3170,3
     180,3190,3200
3140 RM=0.2*3.5:GOTO 3210:REM BRICK
3150 RM=0.08*5:GOTO 3210:REM STONE
3160 RM=0.87:GOTO 3210:REM WOOD
3170 RM=0.2*2:GOTO 3210:REM STUCCO
3180 RM=2:GOTO 3210:REM MASONRY
3190 RM=1.25*8:GOTO 3210:REM LOG
3200 RM=0:REM OTHER
3210 PRINT "{3 SPACES}FOR LIST OF R V
     ALUES FOR INSULATION"
3220 PRINT "{3 SPACES}ENTER -1 FOR IN
     SULATION R VALUE"
3230 PRINT "INSULATION R VALUE";
3240 INPUT RI
3250 IF RI<0 THEN GOSUB 3500:GOTO 323
     0
3260 R(I)=HO+RM+RI+HI:IF PK>1 THEN 33
     40
3270 PRINT "HOW MANY STORIES IN HOUSE
     ";
3280 INPUT NT
3290 PRINT "WHAT IS THE CEILING HEIGH
     T (FT)";
3300 INPUT CH
3310 PRINT "WHAT IS TOTAL PERIMETER (
     FT)";
3320 INPUT P
3330 A(I)=NT*CH*P-A(1)-A(2)
3340 Q(I)=A(I)*DT/R(I)
3350 RETURN
3500 REM LIST OF INSULATION R VALUES
3510 PRINT "{CLEAR}{DOWN}LIST OF INSU
     LATION R VALUES, WALLS"
3520 PRINT "{DOWN}{8 SPACES}NO INSULA
     TION (AIR) = .94"
3530 PRINT "{4 SPACES}BATT INSULATION
      IN WALL = 11"
```

```
3540 PRINT "{4 SPACES}HALF INCH ASPHA
     LT BOARD = 2.4"
3550 PRINT "{3 SPACES}1/2 IN GYPSUM O
     R PLASTER = 1.39"
3560 PRINT "{4 SPACES}1/4 IN WOOD FIB
     ER BOARD = 1.12"
3570 PRINT "{6 SPACES}FIR OR PINE SHE
     ATHING = 1.92"
3580 PRINT "{6 SPACES}3/4 IN PLYWOOD
     PANELS = 1.88"
3590 PRINT "{13 SPACES}1/2 IN PLYWOOD
     = 1.57"
3600 PRINT :PRINT
3610 RETURN
4000 REM CEILING ROUTINE
4010 I=4
4020 HI=0.61:HO=0.61:IF PK>1 THEN 406
     0
4030 PRINT "{CLEAR}{DOWN}WHAT IS TOTA
     L CEILING AREA"
4040 PRINT "OF THE HOUSE";
4050 INPUT TEMP:A(I)=TEMP
4060 PRINT "HOW MANY INCHES OF INSULA
     TION IN CEILING";
4070 INPUT CI
4080 PRINT "TYPE OF INSULATING MATERI
     AL"
4090 PRINT "{DOWN}  1. FIBERGLASS"
4100 PRINT "  2. MINERAL WOOL"
4110 PRINT "  3. VERMICULITE OR PERLI
     TE"
4120 PRINT "  4. CELLULOSE FIBER"
4130 PRINT "  5. U-F FOAM{DOWN}"
4140 INPUT T
4150 RM=CI*IC(T)
4160 R(I)=HO+RM+HI
4170 Q(I)=A(I)*DT/R(I)
4180 RETURN
5000 REM FLOOR ROUTINE
5010 I=5:IF PK>1 THEN 5040
5020 PRINT "{CLEAR}{DOWN}WHAT IS TOTA
     L FLOOR AREA";
5030 INPUT TEMP:A(I)=TEMP
5040 PRINT "HOW MANY ITEMS OF INSULAT
     ION IN FLOOR";
5050 INPUT FI:IF PK>1 THEN 5110
5060 PRINT "TYPE OF FOUNDATION"
5070 PRINT "{3 SPACES}1. OPEN CRAWLSP
     ACE"
5080 PRINT "{3 SPACES}2. ENCLOSED CRA
     WLSPACE OR BASEMENT"
5090 PRINT "{3 SPACES}3. CONCRETE SLA
     B"
5100 INPUT TF
5110 R(I)=HO+FI*3.1+RF(TF)+HI
5120 Q(I)=A(I)*(DT-TC(TF))/R(I)
5130 RETURN
5200 REM DUCTS
5210 DI=0.1
5220 IF TF=3 THEN KD=3:RETURN
5230 PRINT "{DOWN}IS YOUR DUCTWORK IN
     SULATED";
5240 INPUT D$:IF PK>1 THEN 5310
5250 PRINT "{DOWN}LOCATION OF HEAT DU
     CTS:"
5260 PRINT "{4 SPACES}1. ATTIC OR CRA
     WLSPACE"
5270 PRINT "{4 SPACES}2. UNCONDITIONE
     D BASEMENT"
5280 PRINT "{4 SPACES}3. IN SLAB FLOO
     R"
5290 PRINT "{4 SPACES}4. INSIDE CONDI
     TIONED SPACE"
5300 INPUT KD
5310 RETURN
6000 REM WRITE A REPORT
6010 PRINT "{CLEAR}","HEAT LOSS EVALU
     ATION"
6020 PRINT :PRINT :TQ=0
6030 PRINT " ITEM"," AREA"," R-VALUE"
     ,"HEAT LOSS"
6040 PRINT ," SQ.FT.",," BTU/HR":PRIN
     T
6050 FOR I=1 TO 5
6060 A(I)=INT(A(I)*100+0.5)/100
6070 R(I)=INT(R(I)*100+0.5)/100
6080 Q(I)=INT(Q(I)+0.5)
6090 PRINT N$(I*10-9,(I-1)*10+NL(I)),
     A(I),R(I),Q(I)
6100 TA=TA+A(I):TQ=TQ+Q(I)
6110 NEXT I
6120 REM PRINT INFILTRATION LOSS
6130 PRINT "INFILTRATION",,INT((IN(1)
     +IN(2))/2+0.5)
6140 TQ=TQ+(IN(1)+IN(2))/2
6150 REM CALCULATE DUCT LOSS
6160 X=TQ/(A(5)*CH*NT):J=3:K=3
6170 IF X<45 THEN K=2
6180 IF X<35 THEN K=1
6190 DI=0.15+0.05*(3-K)
6200 IF D$="N" AND KD<2 THEN 6240
6205 IF KD>2 THEN DI=0:GOTO 6240
6210 IF OT<15 THEN J=2
6220 IF OT<0 THEN J=1
6230 DI=DM(KD,J+K*4)
6240 PRINT "DUCT LOSS",,,INT(DI*TQ+0.
     5)
6250 TQ=TQ+TQ*DI
6260 PRINT ,"{8 R}",,"{8 R}":?
6270 PRINT " TOTAL  ",INT(TA),,INT(TQ
     )
6280 PRINT
6290 PRINT "DESIGN CONDITIONS:"
6300 PRINT "{3 SPACES}OUTSIDE DESIGN
     TEMP:";OT
6310 PRINT "{4 SPACES}INSIDE DESIGN T
     EMP:";IT
6320 PRINT "TEMPERATURE DIFFERENCE:";
     DT
6330 RETURN
7000 REM FIND SAVINGS USING DEGREE-DA
     YS
7010 DD=2961:DD$="ATLANTA GA"
7012 E1=INT(Q1*DD*24)
7014 E2=INT(Q2*DD*24)
7030 PRINT "{CLEAR}TYPE OF HEATING FU
     EL USED"
7040 PRINT "  1. ELECTRICITY"
7050 PRINT "  2. NATURAL GAS"
7060 PRINT "  3. FUEL OIL"
7070 INPUT FT:PC=0.55
7080 ON FT GOTO 7100,7200,7300
7090 GOTO 7030
7100 REM ELECTRICITY
7110 PRINT "IS HEATING UNIT A HEAT PU
     MP";
7120 INPUT HP$:ER=3413
7130 IF HP$<>"Y" THEN 7150
7140 PRINT "ENTER EER OF HEAT PUMP";:
     INPUT ER:ER=ER*1000
7150 PRINT "AVERAGE $ COST PER KWH";:
     INPUT CO:FU$="KWH"
7160 E1=INT(E1/ER+0.5)
7165 M1=E1*CO
7170 E2=INT(E2/ER+0.5)
7175 M2=E2*CO
7180 MS=M1-M2
```

```
7190 GOTO 7400
7200 REM NATURAL GAS
7210 PRINT "AVERAGE $ COST PER THERM
     OF NATURAL GAS";:INPUT CO
7220 E1=INT(E1/(103000*PC)+0.5)
7225 M1=E1*CO
7230 E2=INT(E2/(103000*PC)+0.5)
7235 M2=E2*CO
7240 MS=M1-M2
7250 FU$="THERMS":GOTO 7400
7300 REM FUEL OIL
7310 PRINT "AVERAGE $ COST PER GALLON
      OF FUEL OIL";:INPUT CO
7320 E1=INT(E1/(138000*PC)+0.5)
7325 M1=E1*CO
7330 E2=INT(E2/(138000*PC)+0.5)
7335 M2=E2*CO
7340 MS=M1-M2:FU$="GALLONS"
7400 REM GIVE RESULTS
7410 M1=INT(M1*100)/100
7420 M2=INT(M2*100)/100
7430 MS=INT(MS*100)/100:IF MS=0 THEN
     MS=1.0E-05
7440 PRINT "{DOWN}TOTAL $ COST OF YOU
     R IMPROVEMENTS";:INPUT CI
7450 PB=INT(CI/MS*1000)/1000
7460 REM REPORT SAVINGS AND PAYBACK
7470 PRINT "{CLEAR}","ANALYSIS OF IMP
     ROVEMENTS"
7480 PRINT :PRINT
7490 PRINT ,,"ENERGY NEEDED"
7500 PRINT "ORIGINAL HOUSE ",E1;" ";F
     U$
7510 PRINT "IMPROVED HOUSE ",E2;" ";F
     U$
7520 PRINT ,,"{9 R}":?
7530 PRINT ,"SAVINGS",E1-E2;" ";FU$
7540 PRINT
7550 PRINT ,,"OPER. COSTS"
7560 PRINT "ORIGINAL HOUSE","$";M1
7570 PRINT "IMPROVED HOUSE","$";M2
7580 PRINT ,,"{9 R}":?
7590 PRINT ,"SAVINGS","$";MS
7600 PRINT :PRINT ,"PAYBACK ",PB;" YE
     ARS"

7610 PRINT :PRINT
7620 PRINT "ABOVE IS BASED ON ONE YEA
     R OF OPERATION"
7630 PRINT "IN ";DD$
7640 RETURN
8000 REM DRAW HOUSE
8010 PRINT :PRINT :PRINT :PRINT
8020 POKE 85,8:? ;"{5 SPACES}{I}{O}"
8030 POKE 85,8:? "{H}{10 SPACES}{J}"
8040 POKE 85,8:? "{V}{I}{O}{I}{O}{I}
     {O}{I}{O}{I}{O}{B}"
8050 POKE 85,8:? "{V}{K}{L}{K}{L}{K}
     {L}{K}{L}{K}{L}{B}{5 SPACES}{J}"
8060 POKE 85,8:? "{V}{I}{O}{I}{O}{I}
     {O}{I}{O}{I}{O}{B}{I}{4 U}{B}"
8070 POKE 85,8:? "{V}{K}{L}{K}{L}{Y}
     {Y}{K}{L}{K}{L}{B}{Y}{4 SPACES}
     {B}"
8080 POKE 85,7:? "{21 M}"
8090 RETURN
```

©

To receive additional information from advertisers in this issue, use the handy reader service cards in the back of the magazine

Warehouse Automation With Personal Computers

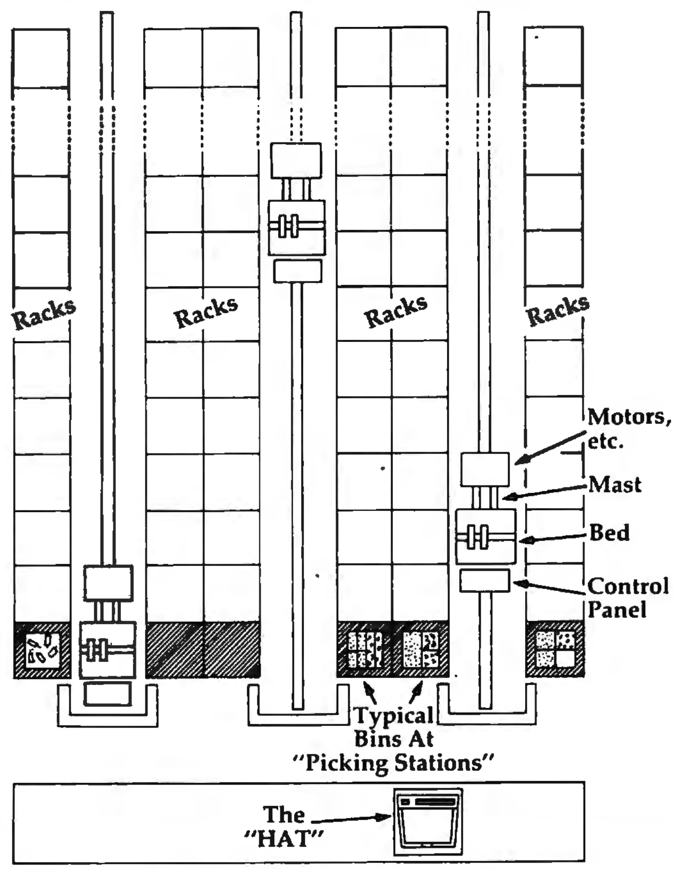
Timothy Stryker, Pompano Beach, FL

While this is not a home application – it does demonstrate the capabilities and sophistication of today's personal computers. This is the story of how a Commodore 8032 runs a large warehouse.

Maybe you use your personal computer for balancing your checkbook, or maybe you use it mainly for playing games. Maybe you even use it to control your furnace or air conditioning system. But did you ever use a personal computer to control a sophisticated industrial automated warehousing system? We did, and it worked out great.

Today's personal computers are sometimes belittled by professional engineers who feel that any computer suitable for home use could not possibly be appropriate for use in an industrial environment. In many cases, they are right: some personal computers are not designed, mechanically, for a great deal of wear and tear, and others have numerous games-related features that would prove less than valuable in an industrial setting. Some personal machines, however, are reliable enough, both mechanically and electrically, not ony to survive, but also to excel when used in industry. The Commodore CBM 8032 is one such machine.

Figure 1: The Warehouse Layout



The Project

Our customer originally started out with an ancient (vintage 1974) automated warehousing system based largely on hard-wired controls. The system consisted of three automatic "cranes" running along tracks in the floors of three separate "aisles," one crane per aisle (see Figure 1).

Each crane could move horizontally up and down its aisle, and had a "bed" that could be moved vertically up and down a 20 foot "mast" to reach packages at different heights (see Figure 2). Packages were arrayed in racks on both the left and the right face of each aisle, and each bed incorporated a "shuttle" which could move left and right, energizing an electromagnet when necessary to pull a package on board.

Each of the three cranes communicated, over a 300-baud asynchronous link, with a single master controller, which sent out commands to the cranes to retrieve various packages, replace others, and so forth. The packages being stored and retrieved contained bins of loose parts. The idea was to fill orders for these loose parts by bringing their respective bins to the front of the aisles, where human "parts pickers" would extract the appropriate parts and then send the bins back into the racks until the next time they were needed.

Our mission was to replace virtually all of the electronic portions of this system with modern, programmable equipment. This involved replacing each of the hard-wired control systems in the cranes with microprocessor-based hardware, as well as coming up with a new master controller, a so-called "Head-of-Aisle Terminal" (HAT).

The HAT would have to communicate with

Figure 2:
A crane, as seen from the front of an aisle, with its bed partway up its mast

each of the three cranes, and would in addition have to communicate over a 4800-baud "bisync" line with the customer's remote IBM mainframe. Simultaneously, it would have to provide for operator input of bin requests, display of system status, diagnostics, and the like.

We had had good results previously with a Motorola 6800-based approach to the replacement of the crane hardware, so this is what we used for the cranes. When it came to replacing the HAT, however, we were momentarily stumped. We knew we wanted a CRT console for operator input, and it had to be reliable. In addition, the computer part of it had to be easy for us to program, and it had to be fast.

We would need the ability to augment the basic system with things like special-purpose synchronous (bisync) communications hardware. We considered the possibility of using an M6800 in an EXOR-bus configuration, with a high-speed asynchronous line going out to a semi-intelligent terminal of some kind, but we realized that this would tend both to bog down the display and to overload the processor.

What we eventually settled on was, of course, the CBM 8032. This fine personal computer combines a 6502 processor, 32K of RAM, a couple of spare 2532-compatible ROM sockets, and a complete memory-mapped video driver circuit, including the CRT tube itself, in a single enclosure.

One major advantage of the 8032, from my standpoint, was that I could use it to write most of the software needed in RPL, a language of my own design that I had specifically optimized for the development of high-speed, memory-critical applications like this one. You may have seen Robert Baker's review of RPL in the February 1982 issue of *Microcomputing;* everything he says in that review is the gospel truth.

By using RPL, I could see that I would have the luxury of writing virtually all of the code in high-level terms, resorting to assembly language only for extraordinarily time-critical functions such as interrupt servicing, block POKEs to the screen, and the like. In addition, the use of the RPL Symbolic Debugger would allow me to test and debug the software in record time, which, since I was working on a fixed-price contract basis, was important.

So, we cut our costs significantly by making use of the machine we planned to sell as its own development system. The fact that the video was memory-mapped meant that updating of the display could proceed at processor speed, yielding instant, random screen updating without any interrupt overhead. The fact that the computer and its display were integrated in a single cabinet lessened the potential mechanical problems of upset and breakage.

Also, the 8032's "memory expansion bus" allowed us to augment the machine's built-in capabilities with boards and other components of our own design, physically located in a separate enclosure. Figure 3 shows the external appearance of the resulting HAT: sharp and professional looking enough for the most discriminating of tastes.

Trials And Tribulations

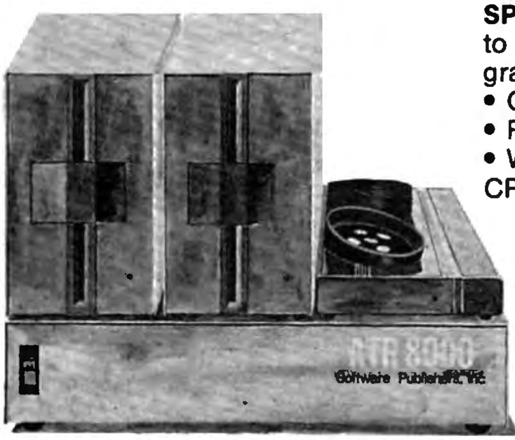
No project of this magnitude, of course, is without its share of problems. The first was that we would need at least 8K of ROM space, thus consuming the two spare 4K ROM sockets, for our control software. Not only did this consume the available

Figure 3: **The HAT**

ATR8000: THE VERSATILE 4MHz Z80 DOUBLE DENSITY DISK INTERFACE FOR THE ATARI® 800/400

Connect the ATR8000 to convert your ATARI 800/400 into a viable business machine. Open the doors to a further dimension by adding the 64K CP/M upgrade—then you can enter the vast CP/M marketplace to select programs tailored to your specific needs.

The ATR8000 is versatile, so it can grow as you need it to. Its handling of disk drives is revolutionary . . . it runs four 5¼" or 8" drives, single or double or quad density, single or double-sided *AND* allows you to mix them!



SPECS: 4MHz Z80 processor • 16k RAM standard • Connects to expansion port of the ATARI 800/400 • 12½" x 11½" x 2½" gray and beige enclosure • Runs four drives of mixed definition • Centronics parallel and RS-232 serial port (and printer drivers) • Runs single density ATARI DOS and existing ATARI software • With OSA+, Ver. 4, is double density • 64K upgrade includes CP/M.

PRICING:

ATR8000	$499.95	OSA+, Ver. 4	$49.95
5¼" Drive	$399.95	Printer Cable	$29.00
64K Upgrade (with CP/M)	$250.00	5¼" Drive Cable	$35.00
		8" Drive Adapter	—CALL—

CONTACT: **SOFTWARE PUBLISHERS, INC.**
2500 E. Randol Mill Rd., Suite 125
Arlington, TX 76011
(817) 469-1181

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

LET'S GET SERIOUS

The ATARI® is a Powerful Personal Computer.
It's time to get down to business.

FORECASTER - Disk - $49.00

Forecast future events based on past information. Forecast profits, costs, sales trends, prices, test scores, virtually anything. Edit, save on disk and test various elements to determine the outcome. Forecaster effectively uses the ATARI® color and sound capabilities. Forecaster is a powerful "what if" program - a must for business. 24K disk.

STAT PLUS - Disk - $49.00

Stat Plus is the most powerful statistic tool we have seen. Run sample space, mean, variance and standard deviation. Do probabilities using binomial, poisson or normal distribution. Also, do students' t-test, Mann-Whitney U test and Chi Square. Do linear regressions; may use 1-12 independent variables. Interact a sample from one module to another, edit, rank and print out. Minimum 24K.

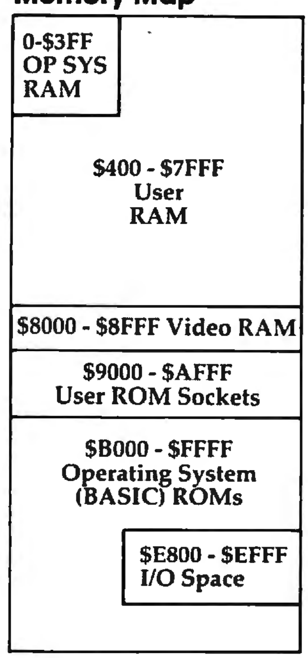
The Programmers Workshop
5230 Clark Ave., Suite 23
Lakewood, CA 90712
Phone Orders (213) 804-1475

TO ORDER: VISA/MasterCard, check or money order accepted. If charge, please include expiration date of card. Add $1.50 for shipping and handling. California residents add sales tax. Phone or mail.

® ATARI is a registered trademark of Warner Communications sockets, but, more importantly, it also consumed (so we thought) the only remaining, non-dedicated portion of the 8032's address space. As shown in Figure 4, Commodore supplies the 8032 with 32K of user RAM occupying addresses 0 through $7FFF, the memory-mapped video display RAM from $8000 to $8FFF, and the main operating system and BASIC interpreter ROMs from $B000 through $FFFF.

Now, there is actually a little gap, running from $E800 to $EFFF, which the 8032 decodes as I/O space rather than ROM. It is in this range that Commodore has placed its PIAs ("Peripheral Interface Adapters", i.e., parallel I/O ports) and VIAs ("Versatile Interface Adapters", another flavor of same) for communicating with the outside world via keyboard, tapes, the IEEE bus, etc.

Figure 4: 8032 Memory Map



What we did not realize at first is that not all of the address space up there is consumed by Commodore's built-in devices: in particular, the range from $EA00 up would appear to be available for user use. By the time we perceived this, though, we had already committed to a bank-switched approach involving the $C000 ROM (*why* it had to be the $C000 ROM is too complicated to go into here).

In so doing, however, we ran afoul of another little peculiarity of the 8032's design, which you may need to know about if you attempt anything like this yourself. This applies whether you use the $EA00 area for I/O or not. What it is, is that the 74LS244's (8-bit tri-state bus drivers) that Commodore uses to bidirectionally buffer the memory expansion data bus are hard-wired to point in the CPU-write direction for all memory accesses to addresses $A000 and above (for addresses in the $9000 to $9FFF range, interestingly enough, this is jumper-selectable).

This leads to the disconcerting problem that, whenever you try to *read* from anything above $A000 on the memory expansion bus, the data comes in as garbage because the bidirectional bus drivers are stupidly driving the wrong way. Our solution to this was to jumper the direction-control inputs of the 74LS244's to a signal we generated off-board which "knew" when a read to the expansion bus was occurring and inverted the drive direction appropriately.

Another little quirk worth noting (forgive me if I'm being overly technical here, but these little insights will be worth about $2000 apiece to you if you ever get involved in this sort of thing) is that the R/W inputs to the ROMs in the 8032 are hard-wired to V_{cc}, implying that you cannot expect a *write* to a ROM address to disable the ROM, or for that matter to do anything reasonable at all.

Why would you want to write to ROM, you may ask? Well, suppose you wanted to substitute a 2532-compatible RAM chip for one of the ROMs, for example? It won't work unless you jumper the socket's R/W line to the CPU's R/W line (and, of course, cut the V_{cc} trace while you're at it). Remembering this can save you days of frenzy and heartburn, when the time comes.

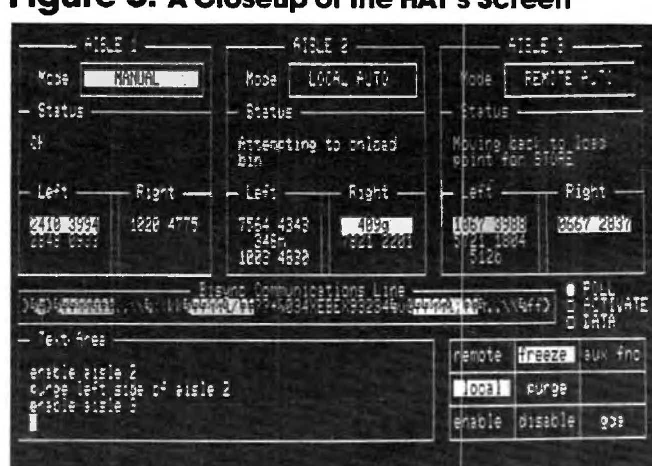
Once these problems were out of the way, the rest of the project went fairly smoothly. Figure 5 shows what the display looked like during actual operation. If you are a Commodore buff, you will notice that some of the characters in the "Bisync Communications Line" box are not part of the standard Commodore character set – this was achieved by substituting a custom 2532 EPROM for the standard character generator chip in the 8032.

This is remarkably easy to do if your character generator chip is socketed (some of the newer ones, unfortunately, are not): you just pop out the existing chip, copy it, with whatever changes you like, into a 2532 (or, if you like, a 2716), and pop the 2532 into the socket. We used a model 2704 PROM programmer/editor from Micro-Link, Inc., of Carmel, Indiana, and found it easy as pie and very satisfying at that.

Software Design

As you can see in Figure 5, our HAT has to keep track of a fair number of things at once. In order to make the system easy for operators to control, I used the "soft-key" technique for input of com-

Figure 5: A Closeup of the HAT's Screen



mands. This unsung hero of a technique, used by Hewlett-Packard in much of their equipment, will, I predict, become the standard command-input technique of the future. The idea behind it is to combine the best features of menu-driven operation with the best features of random-command-driven operation by providing a set of "soft-keys," which, in effect, present the operator with a menu of the currently allowable command options at all times.

Normally, a keyboard must be designed with soft-key operation in mind in order to be so used, but one can sometimes improvise. In the case shown here, for example, the soft-keys are the keys 1 through 9 in the 8032's numeric keypad. The little block of legends in the lower right-hand corner of the display can be thought of as appearing directly on the keys themselves, and the only difference between these legends and normal, "hard" legends is that these legends *change* once you have hit a particular soft-key.

For example, if you hit the "enable" soft-key on the HAT, the word "enable" is echoed onto the bottom line of the "Text Area" shown to the left, and the soft-key legends change to offer you the option of enabling aisle one, aisle two, or aisle three. When you select the aisle you want enabled, your choice is echoed to the Text Area, and the soft-key legends change again to tell you that the only thing you can now do is to hit RETURN (or CLEAR, in case you have changed your mind).

Once you hit RETURN, the command is executed, and the Text Area display scrolls up, retaining a record for you of what you did, in the king's English. Very little possibility for confusion here, especially when you consider that this arrangement makes it *physically impossible* for you to enter a command with invalid syntax!

A fair amount of "human-engineering" (that always sounds to me like android design) also went into the rest of the display shown in Figure 5. The "Bisync Communications Line" box acts as a continuous window onto the line connecting the HAT with its remote IBM mainframe, so that communications problems can be easily diagnosed and corrected. Bytes received are displayed here in normal field (green on black), while bytes transmitted are displayed in reverse field.

Since the HAT is on a multi-drop, shared communications line, it is "polled" in various ways by a network controller. These polling sequences cause the little circles beside the legends POLL, ACTIVATE, and DATA to light up from time to time. This ability to easily simulate LEDs, discrete scrolling windows, and so forth – as though designing a real, mechanical front panel – is one of the big advantages of a direct memory-mapped video display. I made the most of it.

The upper part of the HAT display is devoted to system status, which is logically grouped by aisle. A queue of pending bin requests is maintained for each side of each aisle, and bins currently active in each case are highlighted through the use of reverse field. Aisle mode and status are displayed in English, with highlighting where appropriate.

These fields are all continuously updated in realtime on the screen – the memory-mapped video of the 8032 makes it possible to maintain a high data rate to the tube with very little processor overhead, while the graphics and reverse field capabilities of the machine allow this information to be organized and presented in a clear and aesthetically pleasing way. The fact that the 8032's display is a full 80 columns wide was certainly a boon to us too, as you can see.

No static photograph can really give you an adequate idea of what the HAT screen looks like when the system is in operation. The bytes in the bisync window are constantly whizzing around, while the status fields for each of the aisles are continually changing as the cranes go through their paces. Meanwhile, the bin queues are constantly filling up and emptying out, allowing you to watch as each bin request gradually moves up in sequence until it becomes current.

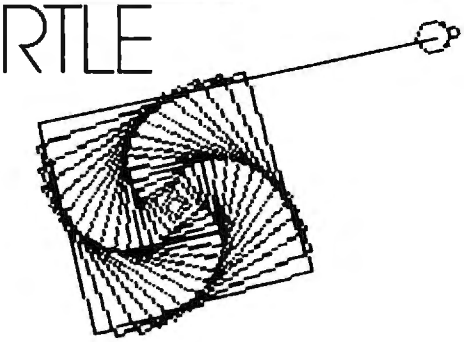
The little POLL, ACTIVATE, and DATA lights blink on and off like mad. And, in spite of all this activity, the operator can enter commands to add new queue entries, purge old ones, enable and disable aisles, and so forth whenever he or she likes, without regard to what else the system might be doing at the time. Operating the HAT actually has a lot in common with playing a realtime video arcade game. It's (dare I say it?) actually fun.

Everybody's Happy

There is a good deal more to the HAT, and to the system as a whole, than I have been able to address here, but I hope that this has given you some feel for what a personal computer is really capable of when pressed to its limits. The HAT and its M6800-controlled cranes have been in operation for several months now and, so far, have run virtually trouble-free. Our customer reports system throughput on the order of double what it was before, and his operators are happy because their jobs are now easier and less confusing.

All of this makes my boss happy that we went with the CBM 8032, and that makes me happy because I'm the one that talked him into it. I'm also happy that I could use this project to demonstrate the viability of RPL in a highly demanding and cost-sensitive application. The next time you're designing a process control system for serious industrial use, think twice about the possibility of rounding it out with a "personal" computer. ©

FRIENDS OF THE TURTLE



David D Thornburg, Associate Editor

Turtle Graphics For The VIC

Judging from the amount of mail I have received on this topic, there are thousands of VIC owners who are waiting for the chance to see turtle graphics on their computer. It was thus with excitement that I viewed the chance to try the Turtle Graphics program cartridge from HES.

This cartridge comes nicely packaged with a thorough manual that contains both a tutorial and a reference section. The cartridge contains an 8K byte program that loaded immediately when the VIC was turned on. Since the program worked on my vintage VIC from Japan, I'm sure it works on every VIC ever made.

I have used this program for a month and am quite ambivalent about it. As someone who uses turtle graphics packages of all sizes and vintages for a host of computers, I confess to having a predefined set of expectations. In order to help you assess my review, it is only fair that I list what I feel are the important aspects of WSFN, Atari PILOT, Apple SuperPILOT, Logo, and the Big Track toy with a felt-tip pen attached to the back:

1. Turtle graphics is just that – a graphics environment. It is capable of drawing continuous lines on a screen (or a sheet of paper).
2. Turtle graphics is richly endowed with commands that allow the incremental movement along a heading and the incremental rotation of the heading by amounts whose values can be stored in variables.
3. The highly interactive and experimental nature of those who use turtle graphics most effectively requires that graphics commands and user-defined procedures be capable of execution directly from the keyboard. A true immediate mode is present in all my favorite turtle systems – including the $40 Big Trak.

Unfortunately, the Turtle Graphics package from HES fails all these tests – and I could have made the list longer with the same result. My biggest complaint is that this program does not use the VIC graphics mode at all, but builds pictures by printing trails of characters on the 22 column by 23 row display screen. This restricts one to very primitive pictures.

This does not mean that the program isn't useful. I feel that, under a new name, this program can find tremendous application in another field, but more on that later.

Structurally, the program is quite nice. It contains its own line editor that performs some error checking before accepting each line. The language itself uses English language commands that, in most cases, are instantly understood by the user. For example, TURTLE COLOR RED changes the turtle's color to red.

When the system is turned on, the user is presented with a menu that allows the addition, insertion, deletion and replacement of program lines, the listing and printing of programs, and the execution and tracing of programs. All aspects of this menu driven system work well.

To get a feel for the language's syntax, let's examine a simple program from the manual:

```
SCREEN COLOR YELLOW
BORDER COLOR PURPLE
TURTLE COLOR BLUE
PEN DOWN
TEXT HI THERE
PEN UP
CHARACTER TO +
TURTLE COLOR GREEN
MOVE TO 6-3
PEN DOWN
RIGHT 5
DOWN 5
LEFT 5
UP 5
STOP
```

The first three commands set colors. Any of the standard VIC colors can be used. The turtle starts with the pen up (unlike all the other turtle systems with which I am familiar), so if you want to see something, you must remember to put the pen down first. The TEXT command functions somewhat like the PILOT T: command in that it prints whatever follows the command. It does not, however, allow you to print out the contents of variables, so it can't be used to print out the results of calculations. Also, unfortunately, there is no INPUT command to allow data to be entered interactively during the execution of a program.

The command CHARACTER TO + tells the turtle to leave a trail of + marks as it moves. It is better to think of the turtle as carrying a rubber stamp rather than a pen. As it moves, it stamps images of characters on the screen. The command MOVE TO 6-3 moves the turtle to the sixth column and third row of the screen. The commands RIGHT, DOWN, LEFT, UP move the turtle the indicated number of spaces in the indicated direction. Note that RIGHT and LEFT are *not* turn commands. They shift the turtle to the right or left. Movement is allowed to take place only along columns or rows, and there is no command that lets the turtle move incrementally along a diagonal.

The user can create labeled procedures with this system and can then invoke the procedures with the USE command. Among other valuable features, the user can have the program check to see if a particular symbol is underneath the turtle. This allows the creation of simple maze-following programs, and is quite valuable.

The conditional branching command, IF, is quite non-standard in its use. Consider this example from the manual:

```
IF (X>2)
SCREEN COLOR BLUE
JUMP DONE
LABEL FALSE
SCREEN COLOR RED
LABEL DONE
```

If the value of X is greater than 2, then the commands immediately following the IF command will be executed. If it is false, execution branches to the next label. I would guess that the reason this was done was to make sure that each program line did only one thing. The IF command then starts to look like the Logo TEST command. I would have been happier if HES had used TEST, IFT and IFF, as does TI Logo. The present construction is quite convoluted and cumbersome, in my opinion.

While no fault of HES, the aspect ratio of the display screen makes any accurate correspondence between a procedure and a drawn figure hard to detect. Suppose, for example, that a child draws a square on a sheet of paper and then translates this square to the procedure:

```
LABEL SQUARE
RIGHT 10
DOWN 10
LEFT 10
UP 10
ROUTINE END
```

When this procedure is executed, a wide rectangle will be drawn on the screen. The procedure *should* give a square, but the aspect ratio of the VIC's character screen will never let you plot a square with this procedure. Does this adversely affect the child's understanding of programming? I think so. This hunch has been reinforced by my limited testing of this package with an eight-year-old boy who certainly knew what a square looked like. This might seem like a minor point to some, unless they are expecting to use this program in an educational environment.

On the positive side, I think that Turtle Graphics is an excellent program for the creation of animated titles and text displays. The user can create musical sounds and can adjust the rate at which characters are placed on the screen. Any VIC owner who makes home video tapes or who needs an inexpensive, eye-catching attraction for a retail display can benefit from this package. In fact, the cost of the VIC plus the $39.95 Turtle Graphics program is far less expensive than the titling systems presently being sold to VCR owners. The VIC's video output connects easily to all VCR's, making this a natural application for the system.

My recommendation is for HES to market this program to the massive number of home video users who can use it to title their recordings. HES should use the proceeds from these sales to develop a *true* turtle graphics package for the VIC.

It is sorely needed.

[*See manufacturer's statement, p. 112.*]

Microworlds For Atari PILOT

I recently heard from Martin Suey, an elementary school teacher in Tulare, California who has been making good use of Atari's turtle graphics in his second grade classroom. After reading about the creation of microworlds – user-controlled environments which one can change at will – he decided to see if he could implement such an environment with Atari PILOT. His program, *Day and Night*, is designed for primary-aged children. The computer displays a scene showing a house with a movable pet (dog or cat) that can be made to walk in front of or behind the house. Pressing the button on the joystick changes the scene slowly from day to night (or from night to day).

Of technical interest to those of you who use this language, Martin's program uses player graphics, priority data registers, and color registers. The program is controlled with a joystick in port #1. Holding the joystick to the left or right moves the player in the indicated direction (with wraparound).

Pushing the joystick up moves the player "behind" the playfield image (behind the house, for example), and pulling the joystick down moves the player to the front of the image. Pressing the button causes the scene to change from day to

Cardco, Inc. announces five All-American ways to . . .

Expand your VIC® at affordable prices

card? A

A universal centronics parallel printer interface for the VIC-20 & C-64 computers. Obeys all standard VIC print commands.
Suggested Reatil — $79.95

cardette 1

A universal cassette interface for the VIC-20 & C-64 computers. Emulates all functions of the data cassette.
Suggested Retail — $29.95

cardboard 3

The CARDBOARD 3 is a fuse protected, economy expansion interface designed to allow the user to access more than one of the plug-in-type memory or utility cartridges Inow available. It will accept up to three cartridges at once.
Suggested Retail — $29.95

cardriter 1

A light pen for the VIC-20 and C-64 computers with a switch on the barrel and 6 good programs.
Suggested Retail — $29.95

cardboard 6

The CARDBOARD 6 is a fuse protected expansion interface designed to allow the user to access more than one of the plug-in-type memory or utility cartridges now available. Additionally it allows switch selection of games and other programs now available in the cartridge format, without the necessity of turning the computer off and on again, thereby saving a great deal of stress on your VIC-20 and on your television or monitor.
Suggested Retail — $139.95

All Cardco products are **Made in the U.S.A.** and are individually tested to ensure quality and reliability. Superior technological engineering optimizes the value/performance ratio of all of our products.

Specifications and prices subject to change.

Cardco, Inc. • 3135 Bayberry • Wichita, KS 67226 • (316) 685-9536

Compuserve® E-Mail Address: Cardco, Inc. • 73575, 1325

Dealer inquiries invited.

®VIC-20 is a registered trademark of Commodore

Manufacturer's Reply

In the interests of providing readers with a fair and balanced report on the features of a product, we asked the manufacturer of VIC Turtle Graphics to reply to David Thornburg's remarks. The following comments are from the author of VIC Turtle Graphics, David Malmberg, and from Human Engineered Software.

While I was developing VIC Turtle Graphics, I had a number of design objectives in mind. Specifically, I wanted TURTLE to be:

- Inexpensive to buy, yet offer a good value for the price,
- Easy to use, yet "bullet proof" and friendly enough for first-time programmers not to get frustrated when they make a mistake,
- Usable with a standard VIC with only 3.5K of available user memory and no disk drive,
- Capable of fully exploiting the VIC's great sound, color and graphics characters,
- An effective vehicle for teaching programming concepts to children and other first-time programmers,
- Fun and educational to use.

On the whole, I believe these objectives have been met or exceeded in the final TURTLE product. Specifically, at $39.95 for a cartridge-based system with a 72-page tutorial instruction manual, TURTLE is an economical and highly effective means of introducing programming concepts.

However, several of the above design objectives are clearly incompatible with Mr. Thornburg's "predefined set of expectations." The most significant incompatibility is obviously TURTLE's lack of high-resolution graphics. This omission was a conscious decision on my part. Hi-res graphics would have required that approximately 4K of the 3.5K available to the user in a standard VIC be set aside for a bit-mapped video display area. This would mean that it would take extra memory and a cartridge slot expander for it to work. None of the Atari or Apple Logo or PILOT systems that Mr. Thornburg cites as his standards of comparison had to deal with the VIC's limited memory – all have at least 16K of usable memory and several are 64K with a required system disk drive. As a result, such a comparison is quite unfair. Had the developers of these systems had only 3.5K to work with, they probably would not have opted for hi-res either.

Furthermore, it is not clear that hi-res is as important to the child who is being introduced to programming concepts as it is to Mr. Thornburg. Does the novice programmer learn more and/or have more fun if his turtle draws a line in hi-res than if it draws a "line" made of VIC graphic characters, such as red hearts or green diamonds? I think not. The acts of planning and debugging the drawing seem to me to be much more important to developing skills in the child than the aesthetics or resolution of the lines used.

Let me correct a few possible misunderstandings that might result from reading the review:

1. The VIC, like all Commodore computers, has an extensive graphic character set. When combined with the VIC's palette of eight colors, it is possible to create some dramatic displays. Only if you lack imagination would you be restricted to "very primitive pictures."

2. TURTLE does have turn commands; and the square in the example could also have been drawn with:

```
LOOP 4
FORWARD 5
TURN RIGHT (OR TURN LEFT)
LOOP END
```

3. Movement on the diagonal is possible by using the MOVE TO ROW – COLUMN (not COLUMN – ROW as explained by Mr. Thornburg). This command causes the Turtle to move to the specified location by the most direct path. If the PEN is DOWN, a trail of graphic characters will be left behind.

4. If someone is bothered by the VIC display screen's aspect ratio and its inability to draw perfectly square squares, I suggest they relabel the procedure as BOX, rather than SQUARE. The aspect ratio is certainly not enough of a problem to conclude that TURTLE has no value in an educational environment – as Mr. Thornburg implies.

Even though I disagree with the overall tenor and conclusions of the review, Mr. Thornburg did make several good observations. His identification of the need for INPUT and PRINT statements that can handle variables is quite valid. His suggestions on ways to improve the IF statement are good. These and other improvements are currently being incorporated into a version of Turtle Graphics for the Commodore 64.

TYPE-SHARE TYPESETTING
EDUCATIONAL SPECIALISTS
SILICON OFFICE SPECIALISTS

ALL UNDER ONE ROOF

SOUTHERN CALIFORNIA
MAINTENANCE CENTER
FOR COMMODORE EQUIPMENT

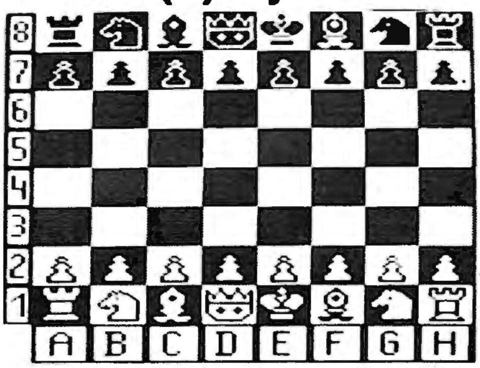
ONE STOP CENTER
for
commodore

VIC-VILLE ™ SOFTWARE

division of Data Equipment Supply Corporation

BOSS (c) by Kavan Software

Exclusive distributors of Kavan Software

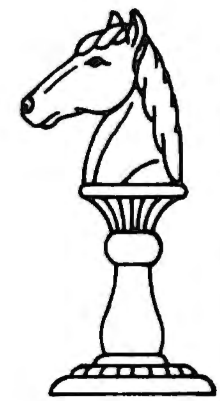


B O S S

0.02.15 S1H 0.00.00
2000

The Definitive Chess Game for the VIC-20

- ★ 10 Levels of Play
- ☆ Beats Sargon II
- ★ Two Clocks
- ☆ Wide range of opening moves
- ★ En passant, queening, castling
- ☆ Change screen and board colors
- ★ Cassette
- ☆ Requires 8K minimum expansion
- ★ 100% machine language

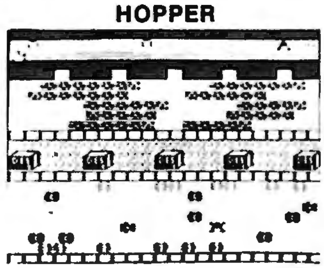


$39.95

BONZO (c) by Kavan

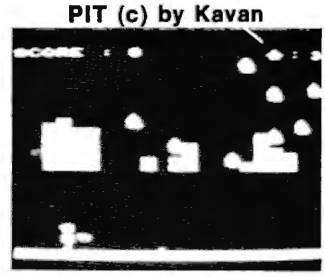
One of the most popular games in Europe. You control BONZO as he climbs the ladders and picks up the point blocks. Watch out for the alien guards. 100% machine language, cassette based. Joystick or keyboard, minimum 8k expansion. **$20.00**

HOPPER



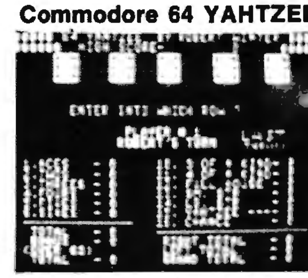
Avoid the cars, dragsters, buildings, logs and other obstacles to bring the frog safely home. Machine language for fast and smooth arcade action. Joystick, standard VIC. **$20.00**

PIT (c) by Kavan



BONZO strikes again as he takes money bags out of the pit. Avoid the alien rain by standing under the shields. Every successfully removed bag of money reinforces your sheilds. 100% machine language, cassette based. Joystick or keyboard, standard VIC. **$18.00**

Commodore 64 YAHTZEE



Commodore 64 version of the famous dice game. 10 player capacity. Watch dice roll across the screen. Automatic tabulation of score and bonuses. Sprite graphics and sound. Cassette based. **$20.00**

Night Crawler **$25.00**
by Interesting Software
Shoot down centipedes, spiders, mushrooms and all kinds of bugs before they get you. Machine language arcade action on standard VIC with joystick.

The Black Castle **$20.00**
Adventure, travel the countryside, fight demons, buy goods, storm the castle. Requires 3k or more expansion.

A Maze Ing **$12.00**
Travel through the maze. Game of skill and tense action. Standard VIC.

Gobbler **$11.00**
Sounds easy? You have 25 seconds to get him and the time gets shorter at each higher level. Standard VIC.

Hang U **$12.00**
Traditional Hangman plays against the VIC's 250 word dictionary or another person. Standard VIC.

Coggle **$11.00**
Computerized version of Boggle. Standard VIC.

Gold Brick **$14.00**
Many levels of play, sound, and color.

3-D Labyrinth **$14.00**
Escape from the labyrinth. Shown in 3-D perspective view with randomly generated mazes. Standard VIC.

Air Strike **$11.00**
Fly the new super bomber V-20 on a mission. Standard VIC.

Attack on Silo III **$12.00**
You are the commander of Silo III. Defend your country. Standard VIC.

Baseball Strategy **$12.00**
The excitement of baseball as a video strategic game. Standard Vic.

Vic Poker **$14.00**
Play poker against the VIC. Hi-res graphics and sound. Standard VIC.

Frogger by (c) Kavan **$14.00**
Eat the flies and avoid the car. Standard VIC.

Space Phreeks **$25.00**
by Interesting Software
Pilot the spaceship "Infinity" and fight the "Space Phreeks". 15 different attack patterns, 33 levels. Machine language, arcade quality. Standard VIC, joystick.

Mailing List **$25.00**
Keep mailing list, print reports, labels. 8k expansion or 16k expansion required.

Astro-Miners **$17.00**
Hi-res graphics and sound space game. Requires 3k or 8k expansion.

Panzer Attack **$14.00**
Enemy tanks are attacking and you must destroy them. Hi-res graphics. Standard VIC.

Pedestrian Polo **$14.00**
Drive your car thru the streets. Based upon Death Race. Standard VIC.

Yahtzee **$12.00**
Solitaire version of this famous dice game. Standard VIC.

Commodore 64 Software **Available Now !!**
64 Monopoly from AP Software
64 Mailing List from VIC-VILLE™ Software
64 Finance from VIC-VILLE™ Software
64 Time Manager 2.0 from TOTL Software

Look for more 64 Software from VIC-VILLE™ & get on our mailing list for all 64 updates and users' group.

Complete descriptive catalog $2.00

Dealers Welcome - Authors Wanted !

Add $3.00 for shipping & handling

Network your CBM, VIC and COMMODORE 64 with the PET SWITCH and VIC SWITCH from DATATRONICS.

Distributors for Datatronic AB

(714) 778-5455

DES

Data Equipment Supply Corp.
8315 Firestone Blvd., Downey, CA 90241

(213) 923-9361

night or vice versa. Michael is interested in hearing from those of you interested in these types of educational applications. I will gladly forward any comments to him.

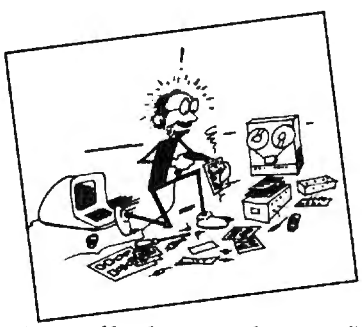
```
10 GR:QUIT
20 R: WRITTEN SEPTEMBER 11,1982
30 C:@B1373=16
40 C:@B1374=2
50 WRITE:S;
60 WRITE:S;
70 WRITE:S;
80 WRITE:S;{4 SPACES}Day and Night
90 WRITE:S;
100 WRITE:S;{9 SPACES}by
110 WRITE:S;
120 WRITE:S;{5 SPACES}Martin Suey
130 *COUNT
140 C:#C=#C+1
150 J(#C=500):*CLEAR
160 J:*COUNT
170 *CLEAR
180 GR:CLEAR
190 *TURTLE
200 C:@B708=20
210 C:@B709=196
220 C:@710=16
230 C:#L=126
240 *LITE
250 C:#P=0
260 C:#L=#L+2
270 C:@B712=#L
280 J(#L=134):*DAY
290 *KNT
300 C:#P=#P+1
310 J(#P=100):*LITE
320 J:*KNT
330 *DAY
340 GR:PEN ERASE

350 GR:GOTO -74,10;5(DRAW 4;TURN 144)
360 GR:GOTO -50,30;5(DRAW 8;TURN 144)
370 GR:GOTO -30,15;5(DRAW 8;TURN 144)
380 GR:GOTO 0,40;5(DRAW 6;TURN 144)
390 GR:GOTO 50,25;5(DRAW 8;TURN 144)
400 GR:GOTO 74,42;5(DRAW 4;TURN 144)
410 GR:PEN RED
420 GR:GOTO 10,-20;TURNTO 0
430 GR:2(DRAW 30;TURN 90;DRAW 40;TURN
     90)
440 GR:PEN BLUE
450 GR:GO 30;TURN 30
460 GR:3(DRAW 40;TURN 120)
470 GR:FILL 39
480 GR:PEN YELLOW
490 GR:GOTO 26,-20;TURNTO 0
500 GR:2(DRAW 10;TURN 90;DRAW 10;TURN
     90)
510 GR:GOTO -79,-32;FILL 11

520 GR:GOTO 26,-20;FILL 10
530 GR:PEN RED
540 GR:GOTO 10,-20;FILL 30
550 GR:GOTO 36,-20;FILL 10
560 GR:GOTO -30,-20;TURNTO 0
570 GR:10(DRAW 5;TURN 90;GO 2;TURN 90
     ;DRAW 5;TURN -90;GO 2;TURN -90)
580 GR:GOTO 51,-20;TURNTO 0
590 GR:8(DRAW 5;TURN 90;GO 2;TURN 90;
     DRAW 5;TURN -90;GO 2;TURN -90)
600 C:@B559=62
610 C:#I=@B106-32
620 C:@B54279=#I
630 C:@B53277=3
640 C:@B704=208
650 C:@B53256=0
660 C:@B53248=125
670 C:#J=#I*256+1024
680 C:#Y=#J+160
690 C:@B#Y=3
700 C:#Y=#J+161
710 C:@B#Y=5
720 C:#Y=#J+162
730 C:@B#Y=6
740 C:#Y=#J+163
750 C:@B#Y=58
760 C:#Y=#J+164
770 C:@B#Y=100
780 C:#Y=#J+165
790 C:@B#Y=212
800 C:#Y=#J+166
810 C:@B#Y=76
820 C:#Y=#J+167
830 C:@B#Y=126
840 C:#Y=#J+168
850 C:@B#Y=202
860 C:#X=125
870 *MOVET
880 J(%T8=1):*CAT
890 J(%J0=2):*PLAYERT
900 J(%J0=1):*PFT
910 J(%J0=4):*LEFTT
920 J(%J0=8):*RIGHTT
930 J:*MOVET
940 *LEFTT
950 C:#X=#X-1
960 C:@B53248=#X
970 J:*MOVET
980 *RIGHTT
990 C:#X=#X+1
1000 C:@B53248=#X
1010 J:*MOVET
1020 *PLAYERT

1030 C:@B623=1
1040 J:*MOVET
1050 *PFT
1060 C:@B623=8
1070 J:*MOVET
1080 *CAT
1090 C:@B708=228
1100 C:@B709=192
1110 C:@B710=16
1120 C:#D=134
1130 *DARK
1140 C:#P=0
1150 C:#D=#D-2
1160 C:@B712=#D
1170 J(#D=128):*NITE
1180 *KOUNT
1190 C:#P=#P+1
1200 J(#P=100):*DARK
1210 J:*KOUNT
1220 *NITE
1230 C:#B=4
1240 *BLACK
1250 C:#P=0
1260 C:#B=#B-2
1270 C:@B712=#B
1280 J(#B=0):*SCENE
1290 *CNT
```

```
1300 C:#P=#P+1
1310 J(#P=100):*BLACK
1320 J:*CNT
1330 *SCENE
1340 GR:PEN RED
1350 GR:GOTO -74,10;5(DRAW 4;TURN 144
     )
1360 GR:GOTO -50,30;5(DRAW 8;TURN 144
     )
1370 GR:GOTO -30,15;5(DRAW 8;TURN 144
     )
1380 GR:GOTO 0,40;5(DRAW 6;TURN 144)
1390 GR:GOTO 50,25;5(DRAW 8;TURN 144)
1400 GR:GOTO 74,42;5(DRAW 4;TURN 144)
1410 GR:GOTO 10,-20;TURNTO 0
1420 GR:2(DRAW 30;TURN 90;DRAW 40;TUR
     N 90)
1430 GR:PEN ERASE
1440 GR:GOTO -30,-20;TURNTO 0
1450 GR:10(DRAW 5;TURN 90;GO 2;TURN 9
     0;DRAW 5;TURN -90;GO 2;TURN -90)
1460 GR:GOTO 51,-20;TURNTO 0
1470 GR:8(DRAW 5;TURN 90;GO 2;TURN 90
     ;DRAW 5;TURN -90;GO 2;TURN -90)
1480 GR:PEN BLUE
1490 GR:GO 30;TURN 30
1500 GR:3(DRAW 40;TURN 120)
1510 GR:FILL 39
1520 GR:PEN YELLOW
1530 GR:GOTO 26,-20;TURNTO 0
1540 GR:2(DRAW 10;TURN 90;DRAW 10;TUR
     N 90)
1550 GR:GOTO -79,-32;FILL 11
1560 GR:GOTO 26,-20;FILL 10
1570 GR:PEN RED
1580 GR:GOTO 10,-20;FILL 30
1590 GR:GOTO 36,-20;FILL 10
1600 C:@B559=62
1610 C:#I=@B106-32
1620 C:@B54279=#I
1630 C:@B53277=3
1640 C:@B704=6
1650 C:@B53256=0
1660 C:@B53248=125
1670 C:#J=#I*256+1024
1680 C:#Y=#J+160
1690 C:@B#Y=10
1700 C:#Y=#J+161
1710 C:@B#Y=14
1720 C:#Y=#J+162
1730 C:@B#Y=21
1740 C:#Y=#J+163
1750 C:@B#Y=219
1760 C:#Y=#J+164
1770 C:@B#Y=68
1780 C:#Y=#J+165
1790 C:@B#Y=124
1800 C:#Y=#J+166
1810 C:@B#Y=124
1820 C:#Y=#J+167
1830 C:@B#Y=68
1840 C:#Y=#J+169
1850 C:@B#Y=170
1860 C:#Y=#J+168
1870 C:@B#Y=202
1880 C:#X=125
1890 *MOVE
1900 J(%TB=1):*TURTLE
1910 J(%JO=2):*PLAYER
1920 J(%JO=1):*PF
1930 J(%JO=4):*LEFT
1940 J(%JO=8):*RIGHT
1950 J:*MOVE
1960 *LEFT
1970 C:#X=#X-1
1980 C:@B53248=#X
1990 J:*MOVE
2000 *RIGHT
2010 C:#X=#X+1
2020 C:@B53248=#X
2030 J:*MOVE
2040 *PLAYER
2050 C:@B623=1
2060 J:*MOVE
2070 *PF
2080 C:@B623=8
2090 J:*MOVE
```

©

A New, Improved Computer Friend For Your Apple

Fred D'Ignazio, Associate Editor



This column catches me in the midst of a move from Chapel Hill, North Carolina, to Roanoke, Virginia. Most of my dozen computers are still in Chapel Hill being looked after by a trusted babysitter. My roof leaks, my shower floods the bathroom floor, my study is buried in boxes, and Catie and Eric just came down with ear infections.

In addition to their sore ears, Catie and Eric are going through something you might call "computer-starvation shock." They think they are still a multi-computer family, and they brag about it to all their friends. They gather a horde of neighborhood kids with the promise of a dozen beeping, flashing computers. They climb the stairs, peek into my study, and what do they see? One lone computer. A rather sad-looking machine, vintage 1977. It doesn't talk, doesn't make pictures, doesn't play music. What a letdown!

Thanks, Chuck!

It's times like these when you readers come in handy. Thanks to one reader – Chuck Johnston of Manhattan Beach, California – I can still provide you with a useful column this month.

Chuck recently sent me a program he wrote that modifies my "Talking Head" program for the Apple. In my opinion, Chuck's program is a substantial improvement on the original version. It's exactly the kind of feedback I'd like to get from my readers. Thanks, Chuck!

Below is Chuck's letter and his program:

I am writing in regard to your column which appears in the September issue of **COMPUTE!** *Magazine. I found your article interesting, but the changes you suggest for the Apple II were, in my opinion, inadequate.*

The Apple is incapable of printing a reverse slash (as is this ancient typewriter), so the head shape you designed does not work. Also, you suggest deleting the sound subroutine, but it makes the program much more interesting. I have revised your program to run on the Apple and thought you might like to see it. I also failed to understand why you didn't draw the head using graphics; as you can see, the resulting animation is much more effective.

Included also is a sound driver program for the Apple in line 20, since, as we know, the Apple is only capable of rudimentary buzzes and clicks in Applesoft. It is POKEd into memory at $0300 and the POKEs in the sound subroutines are as follows: POKE 768,x (where x is a number between 1 and 255) sets the tone frequency. POKE 769,y (y also between 1 and 255) sets the tone duration. In the program enclosed I used the same values as the original program; whether it sounds the same is unlikely, but with some adjustment it could come close. Well, I hope you like the program and thank you for your time.

Chuck Johnston

```
10  REM *** POKE SOUND DRIVER INTO MEMORY
20  FOR I = 770 TO 795: READ M: POKE I,M: NEXT
```

Fred D'Ignazio is a computer enthusiast and author of several books on computers for young people. He is presently working on two major projects: he is writing a series of books on how to create graphics-and-sound adventure games. He is also working on a computer mystery-and-adventure series for young people.

As the father of two young children, Fred has become concerned with introducing the computer to children as a wonderful tool rather than as a forbidding electronic device. His column appears monthly in **COMPUTE!**.

THE LEARNING CENTER

"Perhaps the finest educational software that I have ever had the pleasure of reviewing. It's easy for kids to use and effective in teaching basic concepts and skills.... My kids are learning with it right now!"

Fred D'Ignazio, Associate Editor—*Compute!*, Associate Editor—*Softside*,
Author of bestseller—*Katie and the Computer*

We hope you've been using your ATARI for more than just games ... it is, and can be, a valuable educational tool for you and your children.

Bruce and Dianne Mitchell realized this potential and designed a series of programs for use in their Small World Preschool & Kindergarten located in Durham, N.C. Presented on TV's *PM Magazine* these unique educational programs will introduce your 3-9 year olds to the era of learning with computers. Using the graphics and sound capabilities of the Atari, each program develops a particular skill and reinforces correct responses with happy faces and music.

Beginning with basic concepts such as colors, shapes and the alphabet, your child will progress to an understanding of counting, arithmetic, and language skills.

Widely acclaimed, classroom designed and tested, these unique educational tools are now available to you ... *for your children.*

SPECIAL SKILLS	Preschool	K	1	2
* Color For The Non-Reader	●	●		
* Name That Color		●	●	
* Like Shape Identification	●	●	●	
* Different Shape Identification	●	●	●	
Cave Game	●	●	●	●
MATH AND NUMBER SKILLS				
Count With Me	●	●	●	
Number Recognition	●	●	●	●
Addition		●	●	
Subtraction		●	●	
Add.—Vertical/Horizontal		●	●	
Sub.—Vertical/Horizontal		●	●	
Advanced Addition/Subtraction		●	●	●
Ones and Tens			●	●
LANGUAGE SKILLS				
* Alphabet Recognition			●	●
* Letter Sequence	●	●	●	
* Like Symbol Discrimination	●	●		
* Different Symbol Discrimination	●	●		

Pricing Information	Cass.	Disk		Cass.	Disk
Single Program	6.95	9.95	Pre-School	36.95	41.95
Special Skills	24.95	29.95	Kindergarten	54.95	59.95
Language Skills	19.95	24.95	First Grade	49.95	54.95
Math and Number Skills	34.95	39.95	Second Grade	21.95	26.95
Complete Set	59.95	64.95			
Edumate™ Light Pen	19.95		Add $2.00 for postage & handling		

* Compatible with our Edumate™ Light Pen
NOTE: All software requires 8K cassette/16K disk

Start competing with your 4-year-old for computer time. Enroll in THE LEARNING CENTER, it could be the best investment you ever make ... *in your children's future.* See your local dealer or order direct.

Available now for the ATARI 400/800
COMING SOON for the TRS-80 Model I, III, Color Apple, VIC, TI-99
Free Catalog Upon Request

THE PROGRAMMER'S INSTITUTE
—a division of FUTUREHOUSE™
P.O. Box 3191 Dept C
Chapel Hill, NC 27514

1-919-967-0861 10 am - 9 pm, Monday - Saturday

```
40  GR : HOME
50  FOR P = 1 TO 800: NEXT
100  REM ***DIMENSION VARIABLES
120 N = 1: REM *MESSAGE POINTER
500  REM  *** FRIEND MASTER
510  GOSUB 1010: REM  *FRIEND WAKE UP
520  GOSUB 2010: REM  *FRIEND TALK
530  GOSUB 3210: REM  *STORE CHILD'S NAME
540  GOSUB 2010: REM  *FRIEND TALK
550  PRINT : PRINT : PRINT : PRINT : PRINT :
     END
1000  REM *** FRIEND WAKE UP
1010  GOSUB 5010: REM  *DRAW FACE
1020  GOSUB 5410: REM  *DRAW SLEEPING EYES
1035  FOR P = 1 TO 800: NEXT
1040  GOSUB 4000: REM  *WAKE UP BELL
1050  GOSUB 5460: REM  *DRAW OPEN EYES
1060  FOR P = 1 TO 600: NEXT
1070  GOSUB 5320: REM  *WINK EYE
1080  FOR P = 1 TO 100: NEXT
1085 M = 0: GOSUB 4820: REM  *WINK NOISE
1090  GOSUB 5460: REM  *DRAW OPEN EYES
1100  FOR P = 1 TO 800: NEXT
1110  RETURN
2000  REM *** FRIEND TALK
2005  REM  * SELECT MESSAGE
2006 N = N + 1: REM  * SET POINTER TO NEXT M
     ESSAGE
2010  READ SNUM: REM  * SNUM = NO. OF MESSAG
     ES IN SET
2015  FOR K = 1 TO SNUM
2020  GOSUB 3010: REM  *FRIEND TALK--1 SCREEN
2033  FOR P = 1 TO 1000: NEXT
2035  GOSUB 5510: REM * CLEAR MESSAGE WINDOW
2040  NEXT
2050  RETURN
3000  REM *** FRIEND TALKING--1 SCREEN
3010 PY = 21: REM  *SET VERTICAL TAB FOR TEXT
3040  READ M$
3050  IF M$ = "-1" THEN  RETURN
3051  IF M$ = "*" THEN M$ = N$
3060  VTAB PY
3070  PRINT M$;" ";: GOSUB 5250
3075  GOSUB 4810: REM  *FRIEND SOUND
3080  FOR P = 1 TO 50: NEXT : REM  *KEEP MOU
     TH OPEN
3090  GOSUB 5200: REM  *CLOSE MOUTH
3095  FOR P = 1 TO 100: NEXT : REM  *KEEP MO
     UTH CLOSED
3110  GOTO 3040
3200  REM *** FRIEND ASKS CHILD'S NAME
3210  REM
3220  VTAB 21: HTAB 10: INPUT N$
3265  FOR P = 1 TO 75: NEXT
3267  GOSUB 5510: REM * CLEAR MESSAGE WINDOW
3270  RETURN
4000  REM ***WAKE UP BELL
4010  POKE 768,30: POKE 769,105: CALL 770
4020  FOR P = 1 TO 100: NEXT
4030  POKE 768,20: POKE 769,132: CALL 770
4040  RETURN
4080  RETURN
4625  FOR P = 1 TO 15: NEXT
4800  REM *** FRIEND'S VOICE
4810 M =  INT ( RND (1) * 51) + 15
4820  FOR A = M + 25 TO M STEP  - 8
4830  POKE 768,3: POKE 769,A: CALL 770
4840  NEXT
4880  RETURN
5000  REM *** FRIEND'S FACE
5010  GR
5020  COLOR= 9: PLOT 20,10: PLOT 20,12: HLIN
     19,21 AT 11
5025  COLOR= 7: PLOT 20,13: HLIN 19,21 AT 14
     : HLIN 17,23 AT 16: HLIN 17,23 AT 18
5030  COLOR= 2: HLIN 18,22 AT 15: HLIN 17,23
      AT 17: HLIN 17,23 AT 19
5035  COLOR= 11: FOR I = 20 TO 31: HLIN 17,2
     3 AT I: NEXT : PLOT 16,24: PLOT 24,24: HL
     IN 18,22 AT 32: HLIN 18,22 AT 33
5040  COLOR= 12: HLIN 16,24 AT 34: HLIN 15,2
     5 AT 35: HLIN 15,25 AT 36: HLIN 14,26 AT
     37
5050  COLOR= 1: HLIN 19,21 AT 28
5200  REM  ***CLOSE MOUTH
5220  POKE 1852,177
5230  RETURN
5250  REM *** OPEN MOUTH
5260  POKE 1852,16
5280  RETURN
5300  REM  ***LEFT EYE WINK
5320  POKE 1467,176: POKE 1469,190
5330  FOR I = 1 TO 150: NEXT
5340  RETURN
5400  REM  ***EYES ASLEEP
5410  POKE 1467,190: POKE 1469,190
5420  RETURN
5450  REM  ***EYES AWAKE
5460  POKE 1467,176: POKE 1469,176
5470  RETURN
5500  REM  ***CLEAR MESSAGE WINDOW
5510  HOME
5550  RETURN
5600  REM *** SOUND SUBROUTINE
5610  DATA 172,01,03,174,01,03,169,04,32,168
     ,252,173,48,192,232,208,253,136,208,239
     ,206,0,03,208,231,96
6000  REM  ***MESSAGES
6010  DATA 3
6011  DATA    HI,I'M,GEB,-1
6012  DATA  YOU,TURNED,ME,ON,-1
6013  DATA       WHO'S,OUT,THERE?,-1
6020  DATA 2
6021  DATA    I'M,SO,HAPPY,-1
6022  DATA TO,SEE,YOU,*,-1
```

©

Glenn M Kleiman

Gentle Introductions To Programming

Everyone should understand the fundamentals of programming; learning about programming is an important step towards becoming computer literate. Without a good concept of programming, one cannot really understand the nature of computers, their capabilities, and their limitations.

In addition, programming is an excellent vehicle for developing thinking skills. Many teachers have reported that when children learn to program, their work in other subjects improves. The teachers attribute this general improvement to the students learning to approach problems more systematically and to pay greater attention to detail.

While they acknowledge its importance, few teachers are experienced programmers, and fewer still are well prepared to teach programming. Many dedicated teachers, realizing the need for computer literacy, are making extraordinary efforts to learn about computers and programming so that they can help their students learn. In this month's column, I discuss two courseware packages which can be extremely valuable for such teachers.

The two packages, *Kidstuff* and *Karel the Robot*, are designed to be "gentle introductions to programming." They each contain a simplified programming language and a book with step-by-step lessons for teaching it. With *Kidstuff* or *Karel the Robot*, students (and teachers) can learn many of the fundamental concepts of programming. Both packages can also serve as stepping-stones for students who want to go on to learn BASIC, Logo, PILOT or Pascal.

Kidstuff and *Karel* are not the only existing gentle introductions to programming, and I expect more will be developed in the next few years. Therefore, before turning to the specifics of these two packages, I will discuss in general what we might expect from courseware designed to introduce students to programming.

Structuring Programming

We can think of creating a computer program as involving three main activities. First, we must *design* the program. Recommended approaches to design have been labeled structured programming, successive refinement, top-down programming, and modular programming. In brief, the recommended approach is to start with the most general aims of the program and successively refine them into more and more specific sub-tasks. We want to design the program so that we can work on one sub-task at a time, handling each in a separate module of the program. The modules are then combined to form the program.

Some languages encourage structured programming more than others. For example, in some languages, variables can be local to a module, so you do not have to worry about using the same variable name in different modules of a program. Introductions to programming should encourage structured programming so that students acquire proper habits from the beginning.

The second main activity of programming is to *code* the instructions – translate them into a language the computer can follow. We can discuss computer languages in terms of three types of elements: (1) Commands, such as those which print words and text on the screen, accept inputs from the users of programs, perform mathematical operations, and manipulate text; (2) Control Elements, which are used to iterate (repeat) sets of commands (e.g., FOR/NEXT loops in BASIC), follow commands when a given condition is true (e.g., IF-THEN conditionals), branch to other parts of the program (e.g., GOTO), or use a module (subroutine or procedure) and then return to the current part of the program (e.g., GOSUB); and (3) Data Structures, or ways information can be organized and stored (e.g., simple variables, subscripted variables, hierarchical trees).

BUSINESS and UTILITIES

NEW
PORTFOLIO MANAGEMENT (Apple only) Price: $69.95 two Diskettes
The PORTFOLIO MANAGEMENT package was written by a stock broker to help manage portfolios for individual customers. With this program data files can be easily created and kept up to date. A variety of reports can be generated for clients which are attractively and professionally laid out. The user may define his/her own investment categories. PORTFOLIO MANAGEMENT is a top quality, professional tool which will not only provide you with more convenience but will also serve to enhance your appearance as an efficient and up to date advisor to your clients. Comes complete on two diskettes along with a 38 page instruction manual.

NEW
PERSONAL FINANCE SYSTEM (Available for all computers) Price: $39.95 Diskette
PFS is a single diskette, menu-oriented system composed of ten different programs. Besides recording your expenses and tax deductible items, PFS will sort and summarize expenses by payee, and display information on expenditures by any of 26 user defined codes by month or by payee. PFS will even produce monthly bar graphs of your expenses by category! This powerful package requires only one disk drive, minimal memory (24K Atari, 32K North Star) and will store up to 600 records per disk (and over 1000 records per disk by making a few simple changes to the programs). You can record checks plus cash expenses so that you can finally see where your money goes and eliminate guesswork and tedious hand calculations. Contains high speed machine language sort. PFS has been demonstrated on network (CBS) TV!

FAMILY BUDGET (Apple and Atari only) Price: $34.95 Diskette
FAMILY BUDGET is a very convenient financial record-keeping program. You will be able to keep track of cash and credit expenditures as well as income on a daily basis. You can record tax deductible items and charitable donations. FAMILY BUDGET also provides a continuous record of all credit transactions. You can make daily cash and charge entries to any of 21 different expense accounts as well as to 5 payroll and tax accounts. Data are easily retrieved giving the user complete control over an otherwise complicated (and unorganized!) subject.

NEW
TEXT MASTER (Apple 32K, diskette only) Price: $49.95 Diskette
TEXT MASTER is a general purpose text editor for the Apple II computer. It features powerful English-oriented commands which permit the complete manipulation of textural information. The information treated may be correspondence, computer programs, data to be used by other programs, and more. TEXT MASTER also interfaces with any printer connected to your Apple. The minimum system requirements are 32K of RAM, Applesoft on ROM, at least one disk, and a lower case adapter. TEXT MASTER can process any length file segment by segment. Thus it is possible to process files as large as a diskette. Comes complete with an efficient 16 page manual. TEXT MASTER is equivalent in capability to many word processing systems costing 3 or more times its price. The commands available in TEXT MASTER are: COLLECT, USE, EXECUTE, NONUM, INSERT, MERGE, STOP, RENUM, DELETE, SAVE, COPY, FREE, REPLACE, WAIT, MOVE, LENGTH, LIST, RESAVE, CLEAR, AUTO, RETRY, APPEND, SET, MANUAL, CHANGE, SCRATCH, SHOW, CATALOG, MODIFY, COMPARE, NUM, DISPLAY, HELP.

INTELINK (Atari only) Price: $49.95 Diskette
This software package contains a menu-driven collection of programs for facilitating efficient two-way communications through a full duplex modem (required for use). In one mode of operation you may connect to a data service (e.g., The SOURCE or MicroNet) and quickly load data such as stock quotations onto your diskette for later viewing. This greatly reduces "connect time" and thus the service charge. You may also record the complete contents of a communications session. Additionally, programs written in BASIC, FORTRAN, etc. may be built off-line using the support text editor and later "uploaded" to another computer, making the Atari a very smart terminal. Even Atari BASIC programs may be uploaded. Further, a command file may be built off-line and used later as controlling input for a time-share system. That is, you can set up your sequence of time-share commands and programs, and the Atari will transmit them as needed, batch processing. All this adds up to saving both connect time and your time.

NEW
PAYFIVE (Apple II plus diskette, two drives required) Price: $149.95
This is an enormously flexible employee payroll system with extraordinarily good human engineering features. PAYFIVE prints checks and compiles the required federal, state and local forms for up to 148 employees. The pay methods may be hourly, salary, commissions or any combination. There are multiple options for pay periods, and they also can be used in any combination. PAYFIVE includes many other features and comes extremely well documented with a 200 page manual. The manual may be purchased separately for $30, and that payment later applied to the software purchase.

SHOPPING LIST (Atari only) Price: $12.95 Cassette $16.95 Diskette
SHOPPING LIST stores information on items you purchase at the supermarket. Before going shopping, it will remind you of all the things you might need, and then display (or optionally print) your shopping list and the total cost. Adding, deleting, changing and storing data is very easy. Runs with 16K.

NEW
TAX OPTIMIZER (Available for all computers) Price: $59.95 Diskette
The TAX OPTIMIZER is an easy-to-use, menu-oriented software package which provides a convenient means for analyzing various income tax strategies. The program is designed to provide a quick and easy data entry. Income tax is computed by all tax methods (regular, income averaging, maximum and alternate minimum tax). The user may immediately observe the tax effect of critical financial decisions. TAX OPTIMIZER has been thoroughly field tested in CPA offices and comes complete with the current tax tables in its data files. TAX OPTIMIZER is tax deductible!

STOCK MASTER STOCK PLOT (Apple 48K) Price: $59.95 Diskette
This is a full-featured stock portfolio management and analysis system. Ten years of records on up to thirty stocks may be maintained. You may record prices, revenue, earnings, share, ROL, quarterly earnings and dividends, transactions, long/short term gains, P.E. indexes (DJ, S & P, NASDAQ) and MORE! You may plot (HIRES) the price history of any stock against any other, time, or the indexes. Portfolio value may be evaluated at any time. Comes complete with superbly written instructions and sample files on a second diskette.

TURNKEY AND MENU (Atari only) Price: $17.95 Diskette
TURNKEY is a utility program which allows you to create autoboot autorun diskettes easily. Simply load and run TURNKEY, load the program diskette to be modified, and answer the questions! The TURNKEY diskette also comes with DOS 2.0 and includes another program, MENU. MENU lists the contents of your diskette alphabetically, and permits the running of any BASIC program on the diskette by typing a single key. TURNKEY and MENU provide you with the ability to run any program on your diskette by simply turning on the computer and pressing a single key.

STOCKAID (Atari only) Price: $29.95 Diskette
STOCKAID provides a powerful set of tools for stock market analysis. With STOCKAID you can display point and figure charts, as well as bar charts with oscillators. You can also examine long term moving averages and on-balance volume features. STOCKAID allows you to input daily data with a single diskette storage capability of 239 days × 16 stocks. Included are stock dividend and split adjustment capabilities. A very professional package!

NEW
NYINDEX (Atari only) Price: $29.95 Diskette
NYINDEX is a comprehensive software package for storing, retrieving and plotting New York Stock Exchange information. The daily data treated includes the composite index, advances, declines, new highs and new lows. Graphical displays include the above plus the index oscillator, cumulative advances declines and moving averages. Data entry and editing is easy. The diskette includes more than two years of daily data. NYINDEX is an excellent companion to STOCKAID.

PLAYER-MISSILE GRAPHICS TABLET (Atari only) Price: $19.95 Diskette
The PLAYER-MISSILE GRAPHICS TABLET was designed to take the drudgery out of developing four color displays in GRAPHICS MODE 7. No longer will you have to read the locations of those tiny blocks on your graph paper and calculate PLOTs and DRAWTOs. With PMG you will be able to easily design colorful graphic displays with your joystick and save them on diskette for later recall.

LIFE CYCLE ANALYSIS AND DEPRECIATION (Apple diskette only) Price: $39.95 Diskette
This software package creates a data file of business expenses for equipment which can later be used to calculate and display a variety of reports. You may project annual costs, find the present worth, create depreciation schedules and justify tax deductions. The evaluation techniques conform to standards set by federal agencies. This is an invaluable package for any businessman who has invested in equipment. LIFE CYCLE ANALYSIS features an easy to use data file creation section and provides formatted hardcopy reports for use in presentations or for tax record keeping purposes. When used for generating tax information, this package is tax deductible! Requires 48K. Comes on two diskettes.

NEW
MICROMAGIC (Apple diskette only) Price: $39.95 Diskette
The emphasis of this program is clearly the MAGIC! MICROMAGIC offers outstanding versatility in its ability to function as a stand alone entertainment package or as a utility program to create stunning animated graphics for use in other programs. The secret lies in MICROMAGIC's special on screen graphics editor. You control a graphics cursor directly from the keyboard creating high resolution images using all 16 available colors. When you are done with a picture it can be saved on disk with a single key command. Up to 24 images can be saved as "frames" of a movie, and then played back at high speed to create short animated sequences. The effects are truly stunning.
This package comes complete with demonstration programs so that even novice users can get immediate results. No programming skills are necessary to use MICROMAGIC. If you have been frustrated by the effort required to create graphics images with your computer, MICROMAGIC will delight you.

SHAPE MAGICIAN (Apple II, 48K, diskette only) Price: $29.95
At last! An utility for painlessly creating graphics shapes for the Apple. Create, edit and save up to 30 shapes which can then be used to develop arcade games or to simply enhance your programs. Add that professional touch!

ORDERING INFORMATION

All orders are processed and shipped within 48 hours. Please enclose payment with order and include the appropriate computer information. If paying by VISA or Master Card, include all numbers on card. Purchase orders accepted.

Shipping and Handling Charges
Within North America: Add $2.00
Outside North America: Add 15% (Air Mail)

Delivery
All orders (excluding books) are sent First Class.

Quantity Discounts
Deduct 10% when ordering 3 or more programs, 20% when ordering 5 or more. Dealer discount schedules available upon request.

8" CP/M Disks
Add $2.50 to the listed diskette price for each 8" floppy disk (IBM soft sectored CP/M format). Programs run under Microsoft MBASIC or BASIC-80.

5¼" CP/M Disks
All software available on 8" CP/M disks is also available on 5¼" disks, North Star, Osborne, Superbrain and NEC format.

Ask for DYNACOMP programs at your local software dealer. Write for detailed descriptions of these and other programs from DYNACOMP.

DYNACOMP, Inc. (Dept. E)
1427 Monroe Avenue
Rochester, New York 14618
24 hour message and order phone: (716) 442-8731
Toll free order phones: (800) 828-6772
(800) 828-6773
Office phone (9AM-5PM EST): (716) 442-8960

New York State residents please add 7% NYS sales tax.

EDUCATION

HODGE PODGE (Apple 48K only) Price: $14.95 Cassette $18.95 Diskette
Let HODGE PODGE be your child's teacher. Pressing any key on your Apple will result in a different and intriguing "happening" related to the letter or number of the chosen key. The program's graphics, color and sound are a delight for children from ages 1½ to 7. HODGE PODGE is a non-intimidating teaching device which brings a new dimension to the use of computers in education. See the excellent reviews of this very popular program in INFOWORLD and SOFTALK.

TEACHER'S AIDE (Atari and PET only) Price: $13.95 Cassette/$17.95 Diskette
TEACHER'S AIDE consists of three basic modules contained in one program. The first module provides addition and subtraction exercises of varying levels of difficulty. The second module consists of multiplication problems in which the student may be tested both on the final answer and/or on the subtotal answers in the long hand procedure. Several levels of complexity are provided here as well. The third module consists of division problems; one particularly nice feature of the division module is that the long hand division steps can be displayed along with the remainder in order to clearly demonstrate the procedure by which the remainder is derived. Using TEACHER'S AIDE is not merely a drill, but rather a learning experience.

STATISTICS and ENGINEERING

DIGITAL FILTER (Available for all computers) Price: $39.95 Cassette/$43.95 Diskette
DIGITAL FILTER is a comprehensive data processing program which permits the user to design his own filter function or choose from a menu of filter forms. In the explicit design mode the shape of the frequency transfer function is specified by directly entering points along the desired filter curve. In the menu mode, ideal low pass, high pass and bandpass filters may be approximated to varying degrees according to the number of points used in the calculation. These filters may optionally also be smoothed with a Hanning function. In addition, multi-stage Butterworth filters may be selected. Features of DIGITAL FILTER include plotting of the data before and after filtering, as well as display of the chosen filter functions. Also included are convenient data storage, retrieval and editing procedures.

DATA SMOOTHER (Not available for Atari) Price: $19.95 Cassette $23.95 Diskette
This special data smoothing program may be used to rapidly derive useful information from noisy business and engineering data which are equally spaced. The software features choice in degree and range of fit, as well as smoothed first and second derivative calculation. Also included is automatic plotting of the input data and smoothed results.

FOURIER ANALYZER (Available for all computers) Price: $19.95 Cassette $23.95 Diskette
Use this program to examine the frequency spectra of limited duration signals. The program features automatic scaling and plotting of the input data and results. Practical applications include the analysis of complicated patterns in such fields as electronics, communications and business.

TFA (Transfer Function Analyzer) Price: $19.95/$23.95 Diskette
This is a special software package which may be used to evaluate the transfer functions of systems such as hi-fi amplifiers and filters by examining their response to pulsed inputs. TFA is a major modification of FOURIER ANALYZER and contains an engineering-oriented decibel versus log-frequency plot as well as data editing features. Whereas FOURIER ANALYZER is designed for educational and scientific use, TFA is an engineering tool. Available for all computers.

HARMONIC ANALYZER (Available for all computers) Price: $24.95 Cassette $28.95 Diskette
HARMONIC ANALYZER was designed for the spectrum analysis of repetitive waveforms. Features include data file generation, editing and storage/retrieval as well as data and spectrum plotting. One particularly unique facility is that the input data need not be equally spaced or in order. The original data is sorted and a cubic spline interpolation is used to create the data file required by the FFT algorithm.

FOURIER ANALYZER, TFA and HARMONIC ANALYZER may be purchased together for a combined price of $51.95 (three cassettes) and $63.95 (three diskettes).

REGRESSION I (Available for all computers) Price: $19.95 Cassette $23.95 Diskette
REGRESSION I is a unique and exceptionally versatile one-dimensional least squares "polynomial" curve fitting program. Features include very high accuracy; an automatic degree determination option, an extensive internal library of fitting functions, data editing, automatic data, curve and residual plotting, a statistical analysis (e.g. standard deviation, correlation coefficient, etc.) and much more. In addition, new fits may be tried without reentering the data. REGRESSION I is certainly the cornerstone program in any data analysis software library.

REGRESSION II (Available for all computers) Price: $19.95 Cassette $23.95 Diskette
PARAFIT is designed to handle those cases in which the parameters are imbedded (possibly nonlinearly) in the fitting function. The user simply inserts the functional form, including the parameters (A(1), A(2), etc.) as one or more BASIC statement lines. Data, results and residuals may be manipulated and plotted as with REGRESSION I. Use REGRESSION I for polynomial fitting, and PARAFIT for those complicated functions.

MULTILINEAR REGRESSION (MLR) (Available for all computers) Price: $24.95 Cassette $28.95 Diskette
MLR is a professional software package for analyzing data sets containing two or more linearly independent variables. Besides performing the basic regression calculation, this program also provides easy to use data entry, storage, retrieval and editing functions. In addition, the user may interrogate the solution by supplying values for the independent variables. The number of variables and data size is limited only by the available memory.

REGRESSION I, II and MULTILINEAR REGRESSION may be purchased together for $51.95 (three cassettes) or $63.95 (three diskettes).

ANOVA (Not available on Atari cassette or for PET CBM) Price: $39.95 Cassette $43.95 Diskette
In the past the ANOVA (analysis of variance) procedure has been limited to the large mainframe computers. Now DYNACOMP has brought the power of this method to small systems. For those conversant with ANOVA, the DYNACOMP software package includes the 1-way, 2-way and N-way procedures. Also provided are the Yates 2^{K-P} factorial designs. For those unfamiliar with ANOVA, do not worry. The accompanying documentation was written in a tutorial fashion (by a professor in the subject) and serves as an excellent introduction to the subject. Accompanying ANOVA is a support program for building the data base. Included are several convenient features including data editing, deleting and appending.

BASIC SCIENTIFIC SUBROUTINES, Volumes 1 and 2 (Not available for Atari)
DYNACOMP is the exclusive distributor for the software keyed to the popular texts *BASIC SCIENTIFIC SUBROUTINES, Volumes 1 and 2* by F. Ruckdeschel (see advertisements in BYTE magazine). These subroutines have been assembled according to chapter. Included with each collection is a menu program which selects and demonstrates each subroutine.

Volume 1
Collection #1 Chapters 2 and 3 - Data and function plotting, complex variables and functions
Collection #2 Chapter 4 - Extended matrix and vector operations
Collection #3 Chapters 5 and 6 - Random number generators (Poisson, Gaussian, etc.), series approximations
Price per collection: $16.95 Cassette $20.95 Diskette*
All three collections are available for $44.95 (three cassettes) and $53.95 (three diskettes)

Volume 2
Collection #1 Chapter 1 - Linear, polynomial, multidimensional, parametric least squares
Collection #2 Chapter 2 - Series approximation techniques (economization, inversion, reversion, shifting, etc.)
Collection #3 Chapter 3 - Functional approximations by iteration and recursion
Collection #4 Chapter 4 - CORDIC approximations to trigonometric, hyperbolic, exponential and logarithmic functions
Collection #5 Chapter 5 - Table interpolation, differentiation and integration (Newton, LaGrange, splines)
Collection #6 Chapter 6 - Methods for finding the real roots of functions
Collection #7 Chapter 7 - Methods for finding the complex roots of functions
Collection #8 Chapter 8 - Optimization by steepest descent
Price per collection: $14.95 Cassette $18.95 Diskette

All eight collections are available for $99.95 (eight cassettes) and $129.95 (eight diskettes).
Because the texts are a vital part of the documentation, *BASIC SCIENTIFIC SUBROUTINES, Volumes 1 and 2* are available from DYNACOMP.
BASIC SCIENTIFIC SUBROUTINES, Vol. 1 (319 pages) $19.95 + 75¢ postage
BASIC SCIENTIFIC SUBROUTINES, Vol. 2 (790 pages) $23.95 + $1.50 postage
See reviews in KILOBAUD, Dr. Dobbs, and ACCESS

NEW
SOFTNET (Apple II and TRS-80 48K diskette only) Price: $129.95
SOFTNET may be used to create models of liquid pipeline systems to evaluate their flow performance. Up to 150 nodes with up to 150 connecting elements may be simulated, and models may be combined to form yet larger models. If you are involved in water distribution systems, chemical fluid flow problems, building plumbing, or similar situations, this is an ideal analysis tool.

NEW
FILTER ANALYSIS (Apple only) Price: $19.95 Cassette $23.95 Diskette
FILTER ANALYSIS is the ideal program for determining the frequency response of passive filters. Any number of RLC components may be included and any number of poles treated. FILTER ANALYSIS features its own mini-language which makes circuit description simple. Results may be printed in tabular form or plotted in HIRES graphics (decibels versus log-frequency).

ACTIVE CIRCUIT ANALYSIS (Available for all computers) Price: $35.95 Cassette $39.95 Diskette
With ACAP you may analyze the response of an active or passive component circuit. The circuit may be probed at equal steps in frequency, and the resulting complex voltages at each component juncture examined; the frequency response of a filter or amplifier may be completely determined with respect to both amplitude and phase. In addition, ACAP prints a statistical analysis of the range of voltage responses which result from tolerance variations in the components. ACAP is easy to learn and use. Circuit descriptions may be saved onto cassette or diskette to be recalled at a later time for execution or editing. ACAP should be part of every circuit designer's program library. Requires 48K.

LOGIC SIMULATOR (Available for all computers) Price: $35.95 Cassette $39.95 Diskette
Test your complicated digital logic design with respect to given set of inputs to determine how well the circuit will operate. The elements which may be simulated include multiple input AND, OR, NOR, EXOR, EXNOR and NAND gates, as well as inverters, J-K and D flip-flops, and one-shots. Inputs may be clocked in with varying clock cycle lengths, displacements and delays may be introduced to probe for glitches and race conditions. A timing diagram for any given set of nodes may be plotted. Save your breadboarding until the circuit is checked by LOGIC SIMULATOR. Requires 48K.

NEW
BEAM DEFLECTION (Available for all computers) (disk, diskette only) Price: $29.95 Diskette $32.45 Disk
BEAM DEFLECTION is the first in DYNACOMP's new series of structural analysis software packages. It consists of two programs. The first program permits the development of data files which describe the problem. For example, the ends of the beam may be pinned, clamped or free. The beam may be uniformly supported by an elastic bed, or held up by springs variously placed and having differing spring constants. The elasticity and cross section of the beam may vary along its length. The load may be uniformly distributed or it may include discrete forces. The beam may be pinned at various points along its length. And so on. All this information may be easily entered and edited using the data input program.

Following this the analysis program is called. The calculated results are the stress and deflections of the beam, both in numerical and graphical form. Since the input data is saved, cases may be easily re-run with modification, thereby permitting iterative design.

The documentation which comes with BEAM DEFLECTION clearly shows how to use the software. In addition, three test problems are described and demonstrated to ensure that you understand how to use the program. Also, helpful theoretical information is supplied in the appendix.

STATTEST (Not available on Atari cassette or for PET CBM) Price: $19.95 Cassette $23.95 Diskette
This is a statistical inference package which helps you make wise decisions in the face of uncertainty. In an interactive fashion you can build and edit data files and test the differences in means, variances and proportions. STATTEST will also perform data analysis as well as do linear correlation and regression. This menu-directed statistical workhorse is rounded out with a chi-square contingency test and a (uniform and normal) random sample generator. The documentation is written by a college professor who guides you through the various tests.

ABOUT DYNACOMP

DYNACOMP is a leading distributor of small system software with sales spanning the world (currently in excess of 50 countries). During the past three years we have greatly enlarged the DYNACOMP product line, but have maintained and improved our high level of quality and customer support. The achievement in quality is apparent from our many repeat customers and the software reviews in such publications as COMPUTRONICS, 80 Software Critique, A.N.A.L.O.G., Softalk, Creative Computing and Kilobaud. DYNACOMP software has also been chosen for demonstration on network television. Our customer support is as close as your phone. It is always friendly. The staff is highly trained and always willing to discuss products or give advice.

No matter which language is being taught, in introducing programming to students we generally use a few simple commands, control structures for iteration, conditionals, branching and modules, and only the simplest (if any) data structures. These are the elements we would expect to make up a simplified language.

The third activity of programming is *testing* and *debugging*. Beginners often suffer a great deal of frustration in finding and correcting errors. Some programming languages facilitate debugging by such things as catching syntax errors as the program is entered and allowing the program to be run one step at a time so it can be analyzed carefully. Languages designed to introduce programming should contain such debugging aids.

Kidstuff and *Karel the Robot* provide simple languages so students can learn programming fundamentals with a minimal amount of frustration and delay. They can help students master recommended principles of program design while making the coding and debugging stages as painless as possible.

Kidstuff

Kidstuff, by Thomas R. Smith, is suitable for children as young as first or second grades. It is also appropriate as an easy introduction to programming for older children. *Kidstuff* operates on PET computers, and a version for Commodore 64 computers is being developed.

The commands of the *Kidstuff* language let children write programs to create pictures on the computer screen and play music. The language itself is a mix of turtle graphics-like commands (e.g., DF for draw forward, TR for turn right), modules like those in Logo, branches and loops similar to those of BASIC, a command to use any of the PET graphics symbols, a music command, and special features to aid debugging.

This sounds like a mish-mash, but it has been blended into a coherent teaching tool. A particularly good feature of the *Kidstuff* package is the manual, which contains 13 tutorial lessons, demonstration programs, and suggested projects. The *Kidstuff* language and manual make it possible for all teachers to introduce programming to their students. The manual can be an extremely valuable aid for teachers who are not themselves knowledgeable about programming.

The commands of the *Kidstuff* language are:

DF – draw forward
JF – jump forward without drawing
TR – turn right 90 degrees
TL – turn left 90 degrees
P – select a symbol for drawing (any letter, number, or PET graphics symbol can be used)
B# – play a note of a specified pitch and duration

There are also two control elements similar to GOTO branches and FOR/NEXT loops in BASIC, as well as two simple variables, X and Y.

In addition, *Kidstuff* lets you "teach" the computer new commands. For example, you can tell the computer how to draw a square of size 5:

```
TO SQUARE
BL        [begin a loop]
DF 5      [draw forward 5 steps]
TR        [turn right 90 degrees]
RL 4      [repeat the loop 4 times]
```

Once this is entered, SQUARE can be used just like any of the built-in commands. This capability, similar to the use of procedures in Logo, encourages modular programming.

Kidstuff has several features to facilitate debugging. First, syntax errors are caught as the program is entered, and friendly, clear error messages are given. It's much easier for children to deal with an error message which says "OOPS! THE COMPUTER DOESN'T UNDERSTAND" or "OOPS, LINE NUMBER ERROR" than messages such as "SYNTAX ERROR" or "ERROR 112" found in other languages.

Also, *Kidstuff* has a WALK option which tells the computer to follow the instructions in the program one at a time. When walking, the computer displays an instruction, follows it, and then waits for the child to press the SPACE BAR before going on to the next instruction. This option, similar to TRACE or STEP options in some versions of other languages, is very valuable for helping children analyze their programs and find bugs.

This simple language (I have described all of it) can introduce children to most of the fundamental concepts of programming. The only main concept missing is that there are no conditional (IF-THEN) commands.

Kidstuff is not a powerful language. It is very limited in the number of variables, loops, and new commands possible. However, these limits do not distract from its intended purpose. Once children find the limits of *Kidstuff* constraining, they are ready to go on to learn BASIC or Logo. Having mastered *Kidstuff* first, they will find it easier to learn other languages.

Karel The Robot

Karel the Robot, by Richard E. Pattis, is designed for high school and college students. It teaches concepts of structured programming and can serve as an excellent bridge to learning Pascal, a language now taught in many colleges and universities and becoming increasingly popular in high schools. There is a book about *Karel the Robot*'s language and a "simulator" for Apple II computers that lets you explore the language.

Karel the Robot's world consists of a grid of streets and avenues, walls which block Karel's paths, and beepers which Karel can pick up, carry, and place on street corners. Karel, like all well-behaved robots, obeys simple commands. These are:

MOVE – go forward 1 block
TURNLEFT – pivot 90 degrees to the left
PICKBEEPER – pick up a beeper
PUTBEEPER – put a beeper on a corner
TURNOFF – end the program

In addition, Karel's language contains control elements for repeating instructions (the Pascal ITERATE-TIMES and WHILE-DO commands), conditional tests (IF-THEN-ELSE), and grouping instructions into blocks (BEGIN/END). It also lets you define new instructions. These are some of the most important elements of Pascal, and Karel's language also uses Pascal-like syntax.

The *Karel the Robot* book contains six chapters which present Karel's world and language with example programs, suggested problems, and valuable information about good programming practices. The book is very well done and can be used without the simulator. However, the simulator adds a great deal.

The Karel simulator has a number of excellent features. After you enter your program, it is checked for syntax errors, and useful diagnostic messages are given. There is even a spelling check routine – for example, if you type "MIVE" the computer will display a message saying "I ASSUME YOU MEAN MOVE". Once your program is syntactically correct, you can create the world in which you want Karel to run your program.

You assign Karel a starting location and specify the locations of walls and beepers. You then have many options as to how to run the program. For example, you can select high, medium, or low speed, and you can have Karel leave a trail as he moves. Karel's world is stark, with Karel looking like an arrowhead on the screen. But watching Karel move clearly shows how your program operates.

A "monitor mode" option is an extremely valuable learning aid. In this mode, you can control exactly how Karel proceeds through your program. You can tell Karel how many steps to execute; he does so and then pauses for your next command. At any point you can tell Karel to run the program in reverse, display each command as it is executed, run the program until reaching a specified command, and use other options which make it easy to analyze and debug programs.

Karel the Robot is a well-designed, gentle introduction to programming, as well as a solid stepping-stone for people interested in learning Pascal.

Course disks, which contain solutions to all the problems in the Karel book, are available for $150. The *Karel the Robot* book, published by John Wiley and Sons, is also available separately.

Kidstuff
Thomas R. Smith
P.O. Box 345
Dedham, MA 02026
$59.95 cassette
$69.95 disk (including backup)

Karel The Robot
Cybertronics
999 Mount Kemble Ave.
Morristown, NJ 07960
$85

©

Micros With The Handicapped

Susan Semancik & C. Marshall Curtis

Developing A Communications Program

This is Part 3 of a series of columns to help the handicapped communicate. The program is for the Apple, PET, and VIC.

The following outline shows the progress we've made so far in developing a program to help nonverbal, motor impaired individuals in their communication needs:

I. Introduction (**COMPUTE!**, April 1982)
II. Menu setup (**COMPUTE!**, June 1982)
III. Selection process
    A. Menu selection
        1. Menu storage
            a. Subscripted variables
            b. DATA lists
            c. Screen values
            d. Peripheral devices
            e. Memory blocks
        2. Multiple menus
            a. Access
            b. Display

Selection Of Menu

Example 9 uses subscripted variables and adds lines to the programs of Example 8 (June, 1982) in order to display a chosen entry of the menu at the top of the screen. The user indicates an entry by giving its menu row number and column number (menu row numbers start at one and increase from top to bottom, and menu column numbers start at one and increase from left to right).

In general, the following changes to Example 8 will produce Example 9: change lines 5, 25, 70, and 130: and add lines 300, 310, and 360. Also, the menu is changed so it will fit on all three computers. Example 9 will work on the VIC if W is changed to 22 in line 20. The Apple computer requires the following changes.

```
10 TEXT: HOME: REM CLEAR TEXT SCREEN
75 P=S(C) + TP
95 IF BR=0 THEN TP=TP + W:IF TP>39 THEN
   TP=0:REM UPDATE TAB IF LINE ENDS WITH
   NO LF
300 VTAB 2: INPUT"ROW #, COLUMN #? "
   ;RN,CN: REM INPUT ON 2ND LINE
310 VTAB 1: PRINT M$(RN,CN): REM DESIRED
   MENU ENTRY PRINTED ON TOP LINE
```

Since most home computers use memory-mapped video, when you PRINT characters on the computer's display screen, these characters are stored as screen values in a block of memory. Usually, changing the values within this block of memory will change the screen contents faster than using PRINT commands will.

If you PRINT a menu to the screen from DATA statements, as is done in Example 9, you actually have the menu stored in memory twice: once within the program storage area of memory, and once in the video-mapped area of memory. If you also save the menu using subscripted variables, then you've increased the memory areas to three, since it is now also stored in the variables-storage area of memory. This can be disastrous if your computer has only a small amount of memory and/or your program or menu is large!

Entry Selection

We can eliminate subscripted variables by using the RESTORE statement to pick the words out from the program's DATA statements when we need them. This is implemented for the PET and VIC computers in Example 10, which lists the changes to be made to Example 9. Make the same changes to the Apple version of Example 9, along with the following change:

```
350 READ M$: VTAB 1: PRINT M$: REM DESIRED
   MENU ENTRY PRINTED ON TOP LINE
```

This program also allows an entry to be selected by its menu row and column numbers and displayed at the top of the screen. The advantage here is that no extra memory is required to store the entries as would be needed by using subscripted variables. The disadvantage is that no other DATA statements can be read in the program without careful checking on where the RESTORE and rereading have left the DATA pointers.

Note that previous non-menu DATA entries are bypassed in line 310 of Example 10. If there had been any non-menu DATA entries needed after the menu selection, we would have had to read through the rest of the menu to get to the right DATA statement after it. (Some computers don't have this problem, since their extended BASIC allows restoration to a particular DATA statement.)

An alternative to the use of the RESTORE statement and its possible DATA pointer problem is to pick the selected menu entries from the video-mapped area of memory. This will, however, cause loss of program mobility between different home computers, since this area of memory is not a standardized location. This can be seen by examining the differences between the versions of Example 11, which uses the PEEK statement to pick up the screen values of the selected entry, and uses the POKE statement to display the entry at the top of the screen. In particular, the VIC needs to add the following lines, the last of which is used to set the color register for text to be visible when POKEing the top line of the screen:

```
130 SP=7680: P= SP+(SR-1)*W: GOTO 300
312 CL=PEEK(646): FOR I=38400 TO 38422: POKE
    I,CL: NEXT I
```

The Apple also needs to add the following lines, the last of which is used to help account for the non-linear mapping of the screen:

```
130 SP=1024: GOTO 300
312 R=SR+RN-1+(RN-1)*BR: REM R=SCREEN
    ROW #
315 P1=SP+128*(R-1)-984*INT(R/8)+980*INT(R/24)
```

The DATA statements in Example 11 are used only once to initially display the menu. We should be able to save this memory space by eliminating the DATA statements and entering the menu directly to the screen from peripherals, such as tape recorders or disk units. This concept will be further explored in our next article.

Example 9: For the PET computer – displays a menu by rows from DATA statements, and uses subscripted variables to allow a user to select by menu row and column numbers an entry for display at the top of the screen.

```
5 REM EXAMPLE 9A) PET COMPUTER
10 PRINT CHR$(147);:REM CLEAR TEXT SCREEN
20 W=40:RM=6:BR=1:CM=4:BC=1:RI=2:SR=3:SC=1:RE
   M SET MENU PARAMETERS
25 DIM S(CM),L(CM),M$(RM,CM):S(1)=SC
30 DATA 3,3,5,8:REM COLUMN WIDTHS
35 IF CM=1 THEN 50
38 REM CALCULATE STARTING POSITION OF EACH CO
   LUMN
40 FOR I=2 TO CM:READ L(I-1):S(I)=S(I-1)+L(I-
   1)+BC:NEXT I:READ L(CM)
50 IF SR=1 THEN 70
60 FOR X=1 TO SR-1:PRINT:NEXT X:REM POSITION
   CURSOR TO 1ST ROW OF MENU
65 LP=S(CM)+L(CM)-1:IF LP>W THEN 200
70 TP=0:FOR R=1 TO RM:FOR C=1 TO CM:READ M$:M
   $(R,C)=M$
75 P=S(C)-1+TP
80 PRINT TAB(P);M$;:NEXT C
90 IF S(CM)+LEN(M$)-1<W THEN PRINT:TP=0:GOTO1
   00:REM WRAPAROUND ADVANCES A LINE
95 IF BR=0 THEN TP=TP+W:IF TP>87 THEN TP=0:RE
   M UPDATE TAB IF LINE ENDS W/NO LF
100 IF BR=0 THEN 120
110 FOR B=1 TO BR:PRINT:NEXT B:REM SKIP BLANK
    ROWS BETWN COLUMN ENTRIES
120 NEXT R
130 GOTO 300
139 REM ENTER DATA BY ROWS
140 DATA DR.,IS,COLD,INGEDS12
145 DATA I,AM,WHEN," AOTFR34"
150 DATA YOU,ARE,DRINK,.ULHCP56
155 DATA MOM,EAT,WANT,?MYWKB78
160 DATA DAD,NO,TIME,",VJQZX90"
165 DATA HOT,YES,SLEEP,";$%()'+-"
200 PRINT "MENU SIZE ERROR!":END
300 PRINT CHR$(19):INPUT "ROW #, COLUMN #"; RN
    ,CN:REM INPUT ON 2ND LINE
310 PRINT CHR$(19);M$(RN,CN):REM DESIRED MENU
    ENTRY PRINTED ON TOP LINE
360 GOTO 360:REM DISPLAY ISN'T DISTURBED UNTIL
     USER BREAKS PROGRAM
```

Example 10: For the PET computer – changes to Ex. 9, so RESTORE can be used instead of subscripted variables.

```
25 DIM S(CM),L(CM):S(1)=SC
70 TP=0:FOR R=1 TO RM:FOR C=1 TO CM:READ M$
310 RESTORE:FOR I=1 TO CM:READ N:NEXT I:REM BY
    PASS PREVIOUS DATA
315 IF RN=1 THEN 330:REM BYPASS PREVIOUS MENU
    ROWS
320 FOR R=1 TO RN-1:FOR C=1 TO CM:READ M$:NEXT
     C:NEXT R
330 IF CN=1 THEN 350:REM BYPASS PREVIOUS COLUM
    N ENTRIES
340 FOR C=1 TO CN-1:READ M$:NEXT C
350 READ M$:PRINT CHR$(19);M$:REM DESIRED MENU
     ENTRY PRINTED ON TOP LINE
```

Example 11: For the PET computer – changes to Ex. 10, so PEEK and POKE can be used instead of RESTORE.

```
129 REM SP=STARTING MEMORY AREA FOR SCREEN, P=
    STARTING SCREEN POSITION FOR MENU
130 SP=32768:P=SP+(SR-1)*W:GOTO 300
310 REM P1=STARTING SCREEN POSITION FOR DESIRE
    D ENTRY
315 P1=P+(RN-1)*W+(RN-1)*BR*W
320 P1=P1+S(CN)-1
330 REM P2=ENDING SCREEN POSITION FOR DESIRED
    ENTRY
340 P2=P1+L(CN)-1
350 J=0:FOR I=P1 TO P2:POKE SP+J,PEEK(I):J=J+1
    :NEXT I
```

©

Christmas Bird Count

Jean B. Rogers
Eugene, OR

Personal computers can make any hobby more rewarding. Here's how a PET contributed to the author's bird watching, along with some hints on effective preplanning when writing large programs.

Every year, during a two-week period near Christmas, thousands of bird watchers spend whole days surveying all the birds around them. This event, the annual Audubon Christmas Bird Count, provides large amounts of information about bird populations throughout North and Central America.

The first Christmas Bird Count (CBC) was held on Christmas Day, 1900, when 27 birders noted all the birds they saw during the day. Those birders covered 25 different areas, mostly in cities in the Northeastern USA. CBC's have been held every year since then; currently about 34,000 birders survey nearly 1360 different count areas each year. Results from these CBC's are submitted to the National Audubon Society and are published in its journal, *American Birds*.

In 1979, my teen-aged son and my husband decided to establish a CBC in the area near our home, Port Orford, Oregon. To initiate a CBC, a circular area 15 miles in diameter is chosen, separate from an existing CBC area. This circle is then subdivided into sections, and a group of people is assigned to scour each section, recording every bird identified by sight or sound. Each group tallies the birds according to the number of each species seen. After the count, the number of different species seen by each party of observers and the number of species seen by the total group are counted. For CBC's held in 1979, these totals varied from the Atlantic area of the Panama Canal Zone with 320 species, to Bethel, Alaska, with 4. A reasonable expectation for the Port Orford area is 100 to 120.

Additionally, the observers record the number of individual birds of each species seen on the count. For some species such as Screech Owl, only one individual might be found in the whole count area. Others, like the American Robin or the Common Murre, might be tallied in the thousands. The main data processing task related to a CBC thus is a tabulation of sums of species and individuals seen. A count report including this information is provided for each participant in the count as well as being sent to the National Audubon Society.

Since we wanted an easy-to-read, attractive report, and needed to do some simple numerical calculations, I concluded that this would be a very reasonable task for a microcomputer using BASIC. I had available a PET with 8K of memory and cassette for storage, and a CBM printer. We designed software that worked successfully for the 1979 Port Orford CBC and have used the same programs for CBC's since.

Designing The Project

I think that many amateur programmers have a bigger problem analyzing the project they've undertaken than they do coding it. Thus, I propose to explain how I attacked the problem rather than to provide the BASIC code I used. While the code might be useful to some people with projects very much like mine, the information on problem analysis will possibly be helpful to many people with a wide variety of interests.

The first step in working on the project was to sit down with my son and find out specifically what information he wanted on the output report, as well as approximately what he expected it to look like. The report would essentially consist of a list of the names of birds seen on the count, the number of individuals of that species seen by each of the parties (people assigned to a sub-area of the count circle), and the total seen by the whole group.

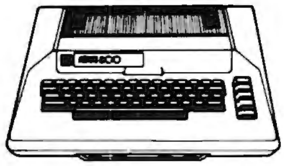
In discussing the report, we realized that, with little additional work, we could produce a field form for use on the count. This is a recording sheet listing the birds one might expect to see, with spaces for tallying the number of individuals of each species seen. Each party has one person designated as recorder who keeps track of the tallies. On the field form and on the final report, birds are listed in phylogenetic order. This is a standard order based on evolutionary progression, and is used in field guides, ornithological research, and scientific documents.

So the overall task was divided down into subtasks: build a bird list, tabulate the results of the count, print the report. A basic list of the birds one might expect to see in our area, then, was the first thing we would need. Having this list on a separate file stored on a tape would make it easily available for whatever future need we had of it.

The Master Bird List

The program to build the list and store it on the

ATARI ™



800 48K . . $619

400 16K . . $259

410 Recorder	$ 75
810 Disk Drive	$419
825 Printer	$579
830 Modem	$155
850 Interface	$165
481 Entertainer	$ 79
482 Educator	$119
483 Programmer	$ 55
484 Communicator	$299
853 16K Ram	$ 75
The Bookkeeper Kit	$169

ATARI Software

CX4104 Mailing List	$ 19
CX404 Word Processor	$105
CXL4007 Music Composer	$ 45
Programming 2 & 3	$ 22
Conversational Languages	$ 45
CX4018 Pilot	$ 59
CX405 Pilot	$ 99
CXL4003 Assembler Editor	$ 45
CX8126 Microsoft Basic	$ 67
CXL4022 Pac-Man	$ 33
CX8130 Caverns of Mars	$ 29
CXL4020 Centipede	$ 33
CXL4006 Super Breakout	$ 28
CXL4008 Space Invaders	$ 28
CXL4009 Computer Chess	$ 28
CXL4011 Star Raiders	$ 33
CXL4012 Missile Command	$ 28
CXL4013 Asteroids	$ 28
The Bookeeper	$105
Home Filing Manager	$ 36
Atari Speed Reading	$ 54
My First Alphabet	$ 26

Business & Utilities

Visicalc	$169
Mail Merge	$ 20
Data Perfect	$ 75
Letter Perfect	$105
Text Wizard	$ 65
Disk Detective	$ 20
Datasm 65 2.0	$ 59
File Manager 800 +	$ 65
Syn Assembler	$ 34
Page 6	$ 20
Atari World	$ 39
K-Dos	$ 59
Micropainter	$ 23
Color Print	$ 27
Lisp Interpreter	$ 79
Bishops Square	$ 20
Graphic Master	$ 27
Graphic Generator	$ 17
Basic Compiler	$ 65

Programming Techniques

Display Lists	$ 17
Horiz/Vert Scroll	$ 17
Page Flipping	$ 17
Basics of Animation	$ 17
Player Missile Graphics	$ 24
Sound	$ 17
Data Files	$ 24

For Fast Delivery, send certified or cashier checks, money orders, or direct bank wire transfers. Personal checks allow 2 to 3 weeks to clear. Prices reflect a cash discount only and are subject to change. Shipping—Software ($2.00 Minimum). Hardware—call. Foreign inquiries invited — add 15% for shipping. Nevada residents add sales tax

NEW LOWER PRICES

TOP SELLERS

Atari

Temple of Apshai	$ 27
Raster Blaster	$ 20
Apple Panic	$ 20
Crossfire	$ 20
Threshold	$ 27
Mousekattack	$ 23
Krazy Shootout	$ 34
Deadline	$ 34
Tumble Bugs	$ 20
Pool 1.5	$ 23
Crypts of Terror	$ 23
Richochet	$ 15
Empire of the Overmind	$ 23
Tanktics	$ 20
Match Racers	$ 20
Wiz & Princess	$ 22
Mission: Asteroid	$ 17
Ali Baba & the Forty Thieves	$ 22
The Shattered Alliance	$ 27
Bug Attack	$ 20
Canyon Climber	$ 20
Shooting Arcade	$ 20
Pacific Coast Highway	$ 20
Clowns & Balloons	$ 20
Ghost Hunter	$ 23
Preppie	$ 20
Rear Guard	$ 17
Lunar Lander	$ 17
War	$ 17
Star Warrior	$ 27
Invasion Orion	$ 17
Dragon's Eye	$ 20
Crush, Crumble & Chomp	$ 20
Jawbreaker	$ 20
Pathfinder	$ 23
Zork I	$ 27
Zork II	$ 27
Action Quest	$ 20
Softporn Adventure	$ 20
Deluxe Invaders	$ 23
Protector	$ 23
Dodge Racer	$ 23
Chicken	$ 23
Nautilus	$ 23
Alien Hell	$ 15
Mar Tesoro	$ 17
Galactic Chase	$ 20
Alien Swarm	$ 23
Intruder	$ 23
Lords of Karma	$ 15
B-1 Nuclear Bomber	$ 12
Rescue at Rigel	$ 20

COMPUTER OUTLET

*** SPECIALS OF THE MONTH ***

ELEPHANT DISKS (BOX)	$ 22
HAYES SMARTMODEM	$209
MOSAIC 32K RAM	$ 99
RAMDISK (128K)	$429
AMDEK COLOR I MONITOR	$309
PERCOM DOUBLE DENSITY DRIVE	$639
NEC 8023A PRINTER	$479
BASIC A +	$ 59
FLIP N' SORT DISKETTE BOX (Holds 50 Diskettes)	$ 21
FLIP-SORT CARTRIDGE BOX (Holds 10 Atari Computer Cartridges)	$ 21
AXIOM GP-100 GRAPHICS PRINTER	$299
AXIOM IMP-4 GRAPHICS PRINTER	$499
MOSAIC 64K RAM	$179
BIT 80 COLUMN BOARD	$289
ALL APX SOFTWARE	15% TO 20% OFF

Computer Outlet

Park Place — Upper Level
1095 E. Twain — (702) 796-0296
Las Vegas, Nevada 89109

Call Toll Free **800-634-6766**
We accept Major Credit Cards
Mon.-Fri. 8 A.M.-6 P. M.
Sat. 9 A.M.-5 P. M.

NEW
Atari

Automated Simulations

Upper Reaches of Apshai (D, C)	$ 15
Curse of Ra (D, C)	$ 15
King Arthur's Heir (D)	$ 20
Escape from Vulcan's Isle (D)	$ 20
Crypt of the Undead (D)	$ 20
The Nightmare (D)	$ 20
Danger in Drindisti (D, C)	$ 15
Armor Assault (D)	$ 27
Monster Maze (CT)	$ 27
Alien Garden (CT)	$ 27
Plattermania (CT)	$ 27

Broderbund

David's Midnight Magic (D)	$ 23
Track Attack (D)	$ 20
Star Blazer (D)	$ 22
Choplifter (D)	$ 23
Deadly Secrets (D)	$ 23
Stellar Shuttle (D, C)	$ 20
Genetic Drift (D, C)	$ 20
Labyrinth (D, C)	$ 20
Serpintine (D)	$ 23
Sea Fox (D)	$ 20

Datasoft

Spell Wizard (D)	$ 53
Sands of Egypt (D)	$ 27
O'Riley's Mine (D, C)	$ 23
Rosen's Brigade (D, C)	$ 23
Fathoms Forth (D)	$ 23

Gebelli

Doctor Goodcode's Cavern (D)	$ 20
Firebird (CT)	$ 34
Embargo (CT)	$ 34

Innovative Design

Pool 400 (CT)	$ 27
Speedway Blast (CT)	$ 27

JV Software

Ghost Encounters (D, C)	$ 20

K-Byte

K-razy Kritters (CT)	$ 34
K-Star Patrol (CT)	$ 34
K-Razy Antiks (CT)	$ 34

L & S Computerware

Crossword Magic (D)	$ 34

Lightning Software

Master Type	$

On-Line

Frogger (D)	$ 23
Ulysses and The Golden Fleece (D)	$ 23
Ultima I (D)	$ 27
Ultima II (D)	$ 39

Roklan Corp.

Gorf	(D) $27, (CT) $30
Wizard of Wor	(D) $27, (CT) $30
Anti Sub Patrol	(D) $20, (C) $15

Sentient

Cyborg (D)	$ 23
Gold Rush (D)	$ 23

Sirius

Space Eggs (D)	$ 20
Sneakers (D)	$ 20
Cyclod (D)	$ 20
Snake Byte (D)	$ 20
Bandits (D)	$ 23
Way Out (D)	$ 27
Fast Eddy (CT)	$ 20
Deadly Duck (CT)	$ 20
World War I (CT)	$ 23
Beanie Bopper (CT)	$ 23

Strategic Simulations

Battle of Shiloh (D, C)	$ 27
Tigers in the Snow (D, C)	$ 27
The Cosmic Balance (D)	$ 27

Big Five Software

Miner 2049er (CT)	$ 34

Bram Inc.

Attack at EP-CYG-4	(D) $22, (C) $20

cassette was very simple.

1. Open the cassette file for writing
2. While there are more birds to go onto the list
   2.1 Input a bird name
   2.2 If the name does not have typos
      2.2.1 Then write the name on the tape file
      2.2.2 Else request re-entering that bird name
3. Close the file

We'd need to be able to make changes in the list when the count results were being tabulated as some unexpected birds would appear, and other expected ones would not.

For creating the field form, however, this list would be used directly. We wanted the complete form to fit on one page for convenience in the field. By reading the whole list into an array in memory, then printing the list in two columns, one from the beginning of the array and one from the midpoint of the array, all the names fit on one sheet.

There was even space for eight unexpected birds to be noted on the bottom of the form. To divide up the tally space, a row of dashes was printed in front of each bird name. The procedure was this:

1. Initialize array space for names
2. Initialize dash string to correct number of dashes
3. Open name file
4. Read complete list into array
5. Close name file
6. Specify format to printer
7. While more copies of the form are needed
   7.1 For number of birds from one to half of total
      7.1.1 Print dash string; bird name (counter); dash string; bird name (counter + halfnumber)
   7.2 Print four lines of pairs of double-dash strings

By using the formatting capability of the CBM printer, it was easy to line up the strings in even rows. This could have been done by padding the name strings with blanks so they would be an even length, as I did later in this project, doing the report.

Processing The Results

After the day of the count we were ready to tabulate the data that had been collected. The primary subtasks of the tabulation and report writing process were these:

1. Get the bird list and edit it
2. Input the numbers of species seen by party
3. Calculate the cross totals and species counts
4. Print the report
5. Save the data for future use

Each of these would be divided further.

The list of names of expected birds was stored on a cassette tape, but some of these birds had not been found on the count day, while a few unexpected others did appear. Using a simple editing program, we read in the original list and wrote out a new list of all the birds sighted on that year's count. The procedure was this:

1. Initialize an array for the names
2. Open the master list file
3. Read names into the array
4. Close master list file
5. Open list file for this year
6. While not at end of list in array
   6.1 Print next name on list for user
   6.2 If a new name should be inserted before next name
      6.2.1 Then accept input of name to be inserted
         Write new name to year file
      6.2.2 Else if next name should be kept
            Then write name to year file
         Move to next name
7. Close the file with this year's list

This procedure deletes birds not seen by simply skipping over them and not writing them on the current year list file.

The next step required entering the data on sightings of each bird by each party. The input mechanism I chose for this portion of the project was the READ-DATA combination. With this method, the data is specified in the program in non-executable statements that look like this:

```
2001 DATA 5,0,4,14,6,3,9
```

They are read by an executable statement (READ) elsewhere in the program. I think of this method as attaching a data file onto the end of the program. When using a PET, there is a very good reason for doing this: the PET screen editor.

The built-in editor on Commodore computers is very flexible and easy to use, not only for correcting typos, but also for duplicating lines or parts of lines. I find that entering a lot of numerical data is hard to do accurately, even when using a number pad. If such data is being input interactively, the user must be asked to confirm each item for correctness, making data entry very boring.

Using the screen editor, however, makes it relatively painless to get a complete set of correct data via DATA lines within the program.

For this project, we needed the number of birds of each species seen by each party. I used one DATA statement for a set of three birds, with the line numbers of the statements keying back to the birds on the list. I then ran the program to combine the names from the cassette file with the data on the sightings.

1. Open name file
2. Initialize array for names
3. Read in names
4. Close name file
5. Open results file on cassette
6. For birds from 1 to end of list in array do
   6.1 Read a set of data from the sequential DATA statements
   6.2 Get the next name from the array
   6.3 Write the name plus the data to the results file
7. Close results file

By now you have noticed that I write intermediate steps of my processing out to cassette files frequently. This is not because I enjoy waiting for the tape read and writes. It is my insurance against radical loss. I am cautious enough about my machine and the perfection of my programs that I never want to get too far away from my last plateau. Additionally, by dividing the total project into chunks, each of the parts did not come up against the size limitation of 8K memory, while a program to do the complete project undoubtedly would have.

Creating The Report

The next step was to actually process the data. This cycle, I read each line of data including the name from the cassette file and processed the numbers in it. I then packed it into a string variable and put it temporarily in an array. It was necessary to do this in sections because the memory is insufficient to hold the complete set of data in the array.

This was still quite convenient, though, because we found that 25 lines of data, plus a heading, fit nicely on a page for the report. We processed it in units of this size, ending up with a report with five pages of results (see the chart).

After each set of 25 was processed, we printed the needed number of copies of that page of the report, then proceeded to the next. The last page was somewhat different because of the totals, but the general process was this:

1. Open input file
2. Initialize
   a) a string array of 25 elements
   b) an eight-element array to read the data into (seen)

Bird Count Results

	NUMBER SEEN BY PARTY						
	1	2	3	4	5	6	TOTAL
COMMON LOON	- Ø	3	Ø	Ø	Ø	1	4
ARCTIC LOON	- Ø	Ø	Ø	1	Ø	1	2
RED-THROATED LOON	- 1	Ø	Ø	4	3	1	9
LOON SP.	- Ø	40	Ø	15	Ø	Ø	55
RED-NECKED GREBE	- Ø	Ø	Ø	1	Ø	1	2
HORNED GREBE	- 15	2	Ø	5	Ø	2	24
EARED GREBE	- 2	Ø	Ø	1	Ø	Ø	3
WESTERN GREBE	- 6	2	Ø	4	Ø	2	14
PIED-BILLED GREBE	- Ø	7	Ø	2	1	14	24
DBL-CRSTD CORMORANT	- 2	12	1	2	1	1	19
BRANDT'S CORMORANT	- Ø	Ø	Ø	2	Ø	Ø	2
PELAGIC CORMORANT	- Ø	14	Ø	61	Ø	8	83
GREAT BLUE HERON	- 3	2	3	3	1	1	13
GREEN HERON	- Ø	Ø	Ø	Ø	Ø	1	1
GREAT EGRET	- Ø	Ø	Ø	Ø	1	1	2
CATTLE EGRET	- 1	Ø	Ø	Ø	2	Ø	3
WHISTLING SWAN	- Ø	Ø	Ø	Ø	1	Ø	1
GOOSE SP.	- Ø	20	Ø	Ø	Ø	Ø	20
MALLARD	- 28	Ø	37	Ø	Ø	Ø	65
GADWALL	- Ø	6	Ø	Ø	Ø	Ø	6
PINTAIL	- Ø	28	Ø	Ø	Ø	Ø	28
GREEN-WNGD TEAL	- 6	10	Ø	2	Ø	Ø	18
AMERICAN WIGEON	- 8	Ø	23	Ø	Ø	2	33
RING-NECKED DUCK	- 3	Ø	Ø	Ø	Ø	10	13
CANVASBACK	- 1	Ø	Ø	Ø	Ø	2	3
YELLOW-RMPD WARBLER	- 76	34	5	46	1	650	812
PALM WARBLER	- 1	Ø	Ø	Ø	Ø	Ø	1
HOUSE SPARROW	- 25	Ø	Ø	Ø	Ø	20	45
WESTERN MEADOWLARK	- 20	Ø	36	72	12	Ø	140
REDWINGED BLACKBIRD	- Ø	Ø	100	Ø	6	Ø	106
BREWER'S BLACKBIRD	- 3	Ø	5	Ø	13	40	61
HOUSE FINCH	- 1	Ø	8	Ø	Ø	6	15
PINE SISKIN	- 16	Ø	Ø	Ø	Ø	5	21
AMERICAN GOLDFINCH	- Ø	Ø	Ø	3	Ø	5	8
RED CROSSBILL	- Ø	Ø	13	Ø	Ø	Ø	13
RUFOUS-SIDE TOWHEE	- 7	2	5	2	1	1	18
SAVANNAH SPARROW	- Ø	Ø	Ø	12	Ø	Ø	12
OREGON JUNCO	- 43	13	70	50	36	53	265
WHITE-CRND SPARROW	- 53	Ø	38	21	13	10	135
GOLDN-CRND SPARROW	- 3	Ø	Ø	1	Ø	Ø	4
FOX SPARROW	- 9	4	11	21	Ø	4	49
LINCOLN'S SPARROW	- Ø	Ø	Ø	1	Ø	Ø	1
SONG SPARROW	- 17	11	4	35	1	9	77
LAPLAND LONGSPUR	- Ø	Ø	Ø	3	Ø	Ø	3
SPECIES SEEN	- 56	54	45	74	38	61	115

TOTAL INDIVIDUALS SEEN- 11944

c) an eight-element array to count species seen by party (count)

3. Create the heading strings
4. For the first hundred birds (four sets) do
   4.1 For 25 data lines do
      4.1.1 Read a data line (name and eight numbers into seen (party))
      4.1.2 For each of the eight parties
         4.1.2.1 If bird seen by the party (not 0) Then increment count (part) by 1
      4.1.3 Sum numbers seen across the eight parties
      4.1.4 Make strings of the numbers seen and the total
      4.1.5 Build a string of the name, number strings, total string
      4.1.6 Place this output string in the string array
      4.1.7 Accumulate grand total of numbers seen
   4.2 For the number of copies of the report needed
      4.2.1 Print heading
      4.2.2 Print the set of 25 output lines
   4.3 Write the set of 25 output lines to a file

The process was repeated in a similar manner for the last page. Here there were fewer data lines, and at the bottom of that page, the total number of species seen by each party and the grand total of individuals and of species seen were printed.

When building the output string, the name and number strings were padded with blanks, effectively formatting the printed output. BASIC's string functions make this quite simple, and storing in one string array again saves space in memory.

Using these programs, we have been able to get reports out to participants within a week of the count. We have been pleased with the quality and attractiveness of the reports, as well as appreciating the use of our personal computer to make another hobby, birding, even more enjoyable. ©

High Resolution Turtle Graphics

Connecting The Strobe Pen Plotter To Apple Turtle PILOT

David D Thornburg, Associate Editor

There comes a time when most users of turtle graphics wish they could get higher resolution pictures than those shown on the display screen. The easiest way to accomplish this is to connect the computer to a graphic pen plotter. Pen plotters have been available for many years, but it is only recently that their cost has dropped to the point that they are affordable to home computer users.

Of the various low-price plotters, the Strobe Model 100 has a price of under $800 (including Apple interface card and software), and has a resolution of 0.002 inches in both axes. It uses inexpensive fine-point pens from the corner drugstore, and plots on plain 8½ x 11 paper. With special pens, it can also plot directly onto plastic sheets for overhead transparencies.

While Strobe provides several application packages for various business and other graphic applications, the plotter can also be interfaced to any program written in Applesoft BASIC. In order to use the plotter with your own programs, you must first load the printer driver program (supplied). Since this program resides just above memory location 35071 and is executed with the Applesoft CALL command, I have not found a way to use this plotter directly from Logo. Anyone who has solved this problem is invited to write about it!

Modify PILOT

Devout turtlers need not feel depressed, however, since the Turtle PILOT language by Alan Poole (published in the September 1982 issue of **COMPUTE!**) is written in Applesoft.

This language system consists of two programs – an editor for creating PILOT listings, and a translator that converts the PILOT program to Applesoft and appends the necessary BASIC utilities needed to make everything work properly. To interface the plotter to the language, one needs only to modify two subroutines and add one new subroutine to the translator program. To keep these programs clear, I will show only the changes that are to be made in the program published in Poole's original article. If you used different line numbers in your version, in order to see where you should put them you will have to compare these changes with the original listing.

The modifications to the translator perform three tasks:

1. We must load the plotter driver routine and initialize the system. Since the routine starting at line 50000 is used at the beginning of every translated program, this is where we will add these tasks.

2. We must add plotter commands after the screen drawing commands so that our plotted image will appear at the same time it is being drawn on the screen. The screen drawing routine begins at line 55000, so this is where we will make these changes.

3. Finally, we need to add a routine that scales the plot commands for the paper size and plotter resolution, sets the pen in the up or down position as appropriate, and ships this assemblage of data to the plotter for execution. We will create this routine starting at line 56000.

Because all the changes are in that portion of the translator appended to each translated program, only one tiny change needs to be made in the PILOT programs themselves. As implemented, the command G:GOTO x,y will only be executed when the next G:DRAW command is given. If you are moving the turtle to a new loction X,Y with the pen up, you can execute this on the plotter with the sequence:

```
G:PEN UP;GOTO X,Y;DRAW 0;PEN DOWN
```

The function of the DRAW 0 command is to force the plotter to carry this motion out before setting the pen down.

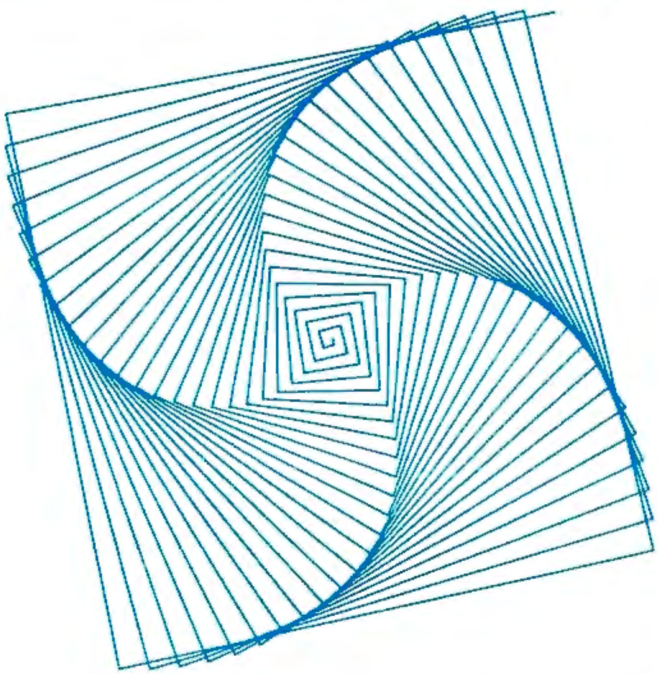
Except for the small inconvenience of adding the extra DRAW 0 commands after each GOTO, any of your existing PILOT turtle graphics programs will run on the plotter as soon as they have been re-translated. I recommend using the original translator for making sure the picture fits on the screen and otherwise does what you want. Once this is done, you can use the modified translator (called, for example, TRANSPLOT) to generate the BASIC program that will both draw pictures on the display and plot them on the plotter at the same time.

To try out the plotter, I entered the following PILOT program:

```
*SQUIRAL
```

```
G: CREAL
C:A=0
*LABEL
G:DRAW A
G:TURN 91
C:A=A+1
J(A<100):*LABEL
E:
```

When this was translated and run, I was able to get a beautiful squiral pattern that was devoid of the jaggies one gets with a raster display screen.

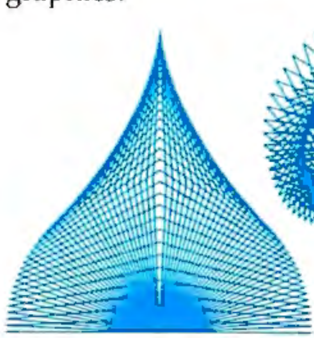


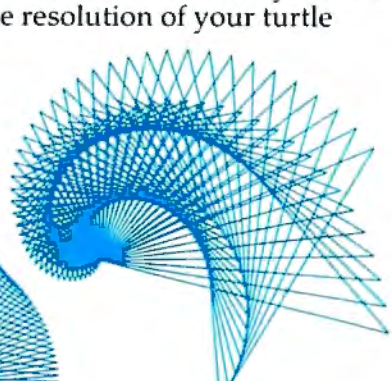
As you create pictures of your own, you will want to change pen colors every so often in the middle of a drawing. An easy way to do this is to use the following procedure when you want to change colors:

```
*QUERY
T:CHANGE PEN AND PRESS RETURN
A:
E:
```

This will stop the execution of the program while you change pens. When you are ready to start plotting again, just press RETURN.

The following figures are but a small indication of the pleasures that await those of you who want to increase the resolution of your turtle graphics.





The changes to be made in the *Translator* program of Apple Turtle PILOT include:

1. Set up procedure:

```
50000 PRINT CHR$(4);"BLOAD PLOT1.8"
50002 HIMEM: 35071
50004 CALL 35081
50006 DIM Q$(25),QS(31)
50008 QP=1:QX=0:QY=0:GOSUB 56000:QP=0
```

2. Modify drawing routine:

```
55004 GOSUB 56000
55045 IF QP=1 THEN GOSUB 56000
55060 HPLOT TO QX+139.0005, -QY+80.0005:
      GOSUB 56000
55070 RETURN
```

3. Add plotter routine:

```
56000 XI=20*(QX+137.5):YI=20*(QY+106.25):
      P%=QP+2
56010 IF XI<0 THEN XI=XI+65536
56020 IX%=XI/256
56030 POKE 35085,IX%
56040 IX%=XI-IX%*256
56050 POKE 35084,IX%
56060 IF YI<0 THEN YI=YI+65536
56070 IY%=YI/256
56080 POKE 35087,IY%
56090 IY%=YI-IY%*256
56100 POKE 35086,IY%
56110 IF P%<0 THEN P%=P%+255
56120 POKE 35088,P%
56130 CALL 35072
56140 RETURN
```

Note: If you are extremely picky about plotting accuracy, add the line:

```
56005 XI=XI*1.0007506:YI=YI*1.0198781
```

Any disk that contains programs generated with *Transplot* also needs to have a copy of the Strobe program *Plot*1.8. To copy this program to your disk, place any Strobe disk in your Apple and enter:

```
BLOAD PLOT1.8
```

Next, insert your program disk and enter:

```
BSAVE PLOT1.8,A$8900,L$6E0
```

©

Apple Educational Games

Sheila Cory, Chapel Hill, NC

If you are either a teacher or a parent of young children and have access to an Apple II+ computer with 48K and a disk drive, there is some software available that you should know about. Produced by The Learning Company, it's specifically designed for preschool and elementary-school youngsters.

This review covers three packages of programs. The first, *Juggles' Rainbow*, is designed for children aged three to six. The second, *Bumble Games*, is for ages four to ten, and the third, *Bumble Plot*, eight to thirteen. All three packages are well designed, and the sequence of the material progresses logically.

Juggles' Rainbow

Juggles' Rainbow is a welcome addition to the small amount of good software for the preschool, kindergarten, and first grade set. Frequently, teachers of very young children are left out when computers are discussed in faculty meetings or workshops, and feel that there's not much that can be done with the computer for children who don't yet have reading skills. It takes great sensitivity to the particular qualities of children of this age to produce software that is interesting, challenging without being too difficult, and educationally sound. *Juggles' Rainbow* shows this sensitivity.

Juggles' Rainbow consists of three programs for children, and one program for teachers or parents. The children's programs are Juggles' Rainbow (the name is used for the entire package and for one of the programs within the package), Juggles' Butterfly, and Juggles' Windmill. The adults' program, called The Big Question Mark, allows the setting of options such as whether sound should be included in the program, whether the child should be given picture and word or just word clues, and gives instructions for dividing the keyboard into halves and quarters for some of the exercises.

Juggles' Rainbow is designed to reinforce the teaching of the concepts of *above* and *below*. The program divides the keyboard into an upper and a lower section with a blue strip of cardboard that is provided with the diskette. A blue line appears on the screen. Children find that when they depress a key below the keyboard divider, a colorful vertical line appears below the blue screen line, and when a key is depressed above the keyboard divider, a colorful line appears above the screen line.

The next segment of this program prompts the user to depress keys above and below the keyboard divider to color in outlined bars above and below the blue screen line. The third segment allows the child to apply his skill with *above* and *below* to create a colorful rainbow.

Juggles' Butterfly reinforces the concepts of *left* and *right*. Again the keyboard is divided with a provided blue strip, but this time the division is in a vertical direction, creating a left and right section of the keyboard. The program works basically the same way as Juggles' Rainbow, but the final segment allows the child to create a marvelous butterfly by applying color to the right and left sides of the butterfly body as keys to the right or left of the keyboard divider are depressed.

Juggles' Windmill takes the learning one step further by having the child depress keys above (or below) the horizontal keyboard divider and to the left (or right) of the vertical divider. The culmination of this activity is the creation of a windmill that would delight a very young child.

Luring our four-year-old visitor, Christopher, away from his LEGO project to try out these programs was difficult, but they quickly absorbed him. This was not only his first opportunity to use the programs, but was also his first time using a computer. A good deal of adult guidance was needed to help him figure out what he was supposed to do, and to extend the learning. This program could make ideal use of a classroom volunteer or older child whose role would be to talk through the concepts, exclaim over the results, and guide the discoveries made using the computer.

One problem Christopher had was keeping the cardboard keyboard dividers in place. I recommend that a piece of heavy blue yarn be used instead of the cardboard. The yarn could be placed between the second and third rows of the keyboard, rather than over the third row as suggested in the manual, and the yarn could be securely taped in place at each end. A similar procedure could be used for the vertical keyboard divider. Christopher's interest in the activity lasted about ten minutes, giving him time to get through Juggles' Rainbow and begin Juggles' Butterfly. His enjoyment of the activity was evident when he asked me if he could play the rainbow game again before he went home.

Bumble Games

Bumble Games introduces the delightful Bumble, who is the central character in all of the programs in *Bumble Games* and *Bumble Plot*. The learning objective in *Bumble Games* is to teach the graphing of positive numbers. Some of the concepts covered are also covered in the MECC (Minnesota Educational Computing Consortium) game of Hurkle. The *Bumble Games* diskette contains six programs, each one progressively more sophisticated. The sequence is excellent, extending the learning by a small degree with each successive game.

The program Find Your Number begins the sequence by giving practice in finding a number between zero and five that Bumble has secretly chosen. The child is shown a horizontal or a vertical number line with the numbers zero to five on it, and makes a guess. Bumble responds with a left or right arrow in the case of a horizontal number line, and an up or down arrow in the case of a vertical number line, indicating whether the next number guessed should be more or less than the present guess. The horizontal and vertical number lines begin preparing children for an X and Y axis that they'll see in a later program. When the number is guessed, the child gets a colorful display of the number, accompanied by tones representing the number. If two children want to play this game, Bumble will select two numbers.

Find the Bumble introduces a four-by-four grid, cleverly differentiating the X and Y axes by labeling one with letters and one with numbers. Bumble hides in one of the boxes formed by the grid, and the child must find Bumble by naming the coordinates of his box. Bumble is very helpful, telling the child to pick bigger or smaller numbers for the Y axis, and letters to the left or to the right for the X axis. When Bumble is found, his friendly, bigger-than-life image appears on the screen.

Butterfly Hunt has Bumble out searching for his lost butterfly. This game works very much like Find the Bumble, but a slightly larger grid prepares the child for the next game, Visit From Space.

Visit From Space introduces the idea that the intersection of two lines in a grid can be named by using a number on the bottom of the grid and one at the side. In this game, both X and Y axes are labeled with numbers. The object of the game is for the child to find Bumble's cousin who has flown in from outer space and is hiding in his spaceship somewhere on the grid. Very clear graphic and written clues help the child learn to locate exactly the intersection he wants to guess. When the spaceship is finally found, it zooms across the screen, making appropriate outer-space noises!

Tic Tac Toc is a game for two players, similar to the more conventional tic tac toe. The idea is for the child to get four markers in a horizontal, vertical, or diagonal line before his or her opponent does. The game screen consists of a five-by-five grid, and a marker is placed by naming the coordinates of the desired position.

The board is somewhat confusing to the beginning player; it would be helpful for the teacher to make a similar board on a transparency and use the overhead projector to play the game a few times with the whole class before children begin to play the game on the computer. The game does give excellent practice in naming points on a grid. It is just different enough from tic tac toe to be interesting.

Bumble Dots extends the grid to ten-by-ten. In this game, Bumble helps the child draw dot-to-dot pictures. A dot appears on the grid, and the child is asked to name it. When the first dot is successfully named, a second dot appears, and when that is successfully named, a line is drawn to connect the dots. This procedure continues until a whole picture is drawn.

The child can also make his own picture by naming coordinates for Bumble to connect. Bumble first asks the child how many dots will be in his picture. Since this is difficult for a child to ascertain in advance, it would be helpful if the teacher had the children first draw a picture using three to nine dots on a piece of graph paper, and then bring that picture to the computer when their turn comes. Children would then be all set to answer Bumble's question about the number of dots needed for the picture. The Learning Company, in developing this program, recognized the fact that generations of children have loved dot-to-dot pictures, and that a natural progression of learning can take place by tapping into this love.

Bumble Plot

Bumble Plot extends the learning about grids to include negative numbers. It consists of five programs, again carefully sequenced to take the child comfortably through the steps culminating with naming points on a ten-by-ten grid where negative numbers are used and the 0,0 point is in the middle. The sequence starts with Trap and Guess, where the child tries to trap Bumble's secret number on a minus three to plus three horizontal or vertical number line. Bumblebug has Bumble hopping around on a grid; the object is to set traps for him to jump into! In Hidden Treasure, the child searches for invisible treasures on a ten-by-ten grid with negative numbers. I found a ship's anchor, a diamond ring, a friendly octopus, and a golden crown! Children would enjoy a worksheet where they could show what they found and where they found it when they played the game. These worksheets could be displayed on the bulletin board above the computer.

Bumble Art is similar to Bumble Dots, but contains negative numbers in the grid. The most action-packed game of the series is Roadblock. The object of this game is to build roadblocks to surround the bank robber before he gets away. This, of course, all takes place on a minus-five by plus-five grid, providing wonderful practice in the skills that have been developed through all of the other games.

All three of the packages reviewed here share some very positive qualities. They all contain excellent graphics; they use sound appropriately to enhance the learning or entertainment value of the program, and sound can be turned off if it provides a distraction in the classroom. The programs are very user friendly, take all kinds of input without bombing, give the user excellent prompts, and have very carefully formatted screen displays.

Manuals are well illustrated and appealing. Each one gives instructions on how to load the diskette and a little information about each program. It would have been very useful to have included suggestions for teachers about things to talk about before each program, and appropriate worksheets for follow-up activities.

The company will send you a set of activity cards for free when you send back the owner registration card. This card also entitles you to purchase a backup diskette for $12. No teacher should ever use software in the classroom without having a back-up diskette.

I suspect that schools that purchase software from The Learning Company will have a new little character joining Snoopy and The Cat In The Hat in adorning their bulletin boards. Bumble has great personal appeal and represents software that is both educationally sound and fun to use.

Juggles' Rainbow ($45)
Bumble Games ($60)
Bumble Plot ($60)
The Learning Co.
4370 Alpine Road
Portola Valley, CA 94025

PROMQUEEN (VIC-20 Hardware)

Harvey B. Herman, Associate Editor

The hardware reviewed here will be of interest to a select group of **COMPUTE!** readers. If you own a VIC and have the need to "burn EPROMs," you should consider this cartridge. On the other hand, if you are completely befuddled by the previous sentence, go to the next article; save $200.

I was excited when I received the PROMQUEEN for review, as it was just what I needed. Several pieces of computer-related equipment which I use daily contain EPROMs (Erasable Programmable Read Only Memory chips). What would I do if one failed and I had no way to replace it? The PROMQUEEN promised to solve this potential problem, even for one like myself, who had never programmed an EPROM before.

It is misleading to think of the PROMQUEEN exclusively as hardware. What is visible, of course, is a cartridge (hardware), which plugs into the memory expansion port of the VIC. But it also comes with essential software (actually firmware on EPROM) without which the hardware would be useless. A 25-page instruction manual is included as well.

The major function of this product is to allow the user to conveniently burn EPROMs. That is, data is to be stored into an erased EPROM so that it will be there the next day even after power has been turned off. This data can be copied from a previously programmed EPROM or typed in from scratch using the monitor program provided.

I was a little apprehensive on my maiden EPROM burning session. First, an EPROM had to be erased. No problem here. I used a shortwave ultraviolet mineral lamp (2537 A wavelength). The shortest erase time I tried was 40 minutes. If you don't have one of these lying around, there are several advertised units which should be satisfactory.

Next, I attempted a copy of the EPROM Hexkit program which comes with the PROMQUEEN. The instructions were somewhat confusing. It was not immediately clear that there is both RAM and ROM memory in the package and that you have the option of moving these around by switches. The ROM memory is only there if an EPROM is plugged into the external socket. However, I was using a preliminary manual. The manufacturer promises that an improved manual will be available shortly.

The burning of an EPROM is actually easy once you know what you are doing. The procedure is:

1. Insert the Hexkit EPROM in the zero insertion force socket (ZIF).
2. Set the switches correctly.
3. Transfer the program to the VIC RAM with a SYS call.
4. Insert the EPROM to be copied.
5. Transfer the EPROM data to the PROMQUEEN RAM using the transfer function of Hexkit.
6. Insert the erased EPROM (2716, 2732 or 2732A – a 2532 will not work).
7. Use the burn function of Hexkit. The software first checks for a properly erased EPROM and later verifies the burn. After several false starts, I had successfully burned my first EPROM.

I have not described all of the features of the PROMQUEEN. The Hexkit program has many other features in addition to the burn function. It also uses color effectively: red screen when burning, for example. The hardware has lights (LEDs) which minimize the chance of error. It can also be used to emulate a ROM when working with other computers.

As with most equipment, I can cite good and bad features. The cartridge is well-constructed and easy to use once you know how. However, the version I tested included confusing directions and it is priced at more than twice what a similar system for the PET goes for. Nevertheless, if you need an EPROM burner and already own a VIC, this could be the best way to get one.

PROMQUEEN
Gloucester Computer Bus Co.
6 Brooks Road
Gloucester, MA 01930
$199

Preppie! For Atari

Mike Kinnamon, Stillwater, OK

Look out, *Star Raiders*! Move over, *Pac-Man*! *Preppie!* is here. It is time to return from patrolling the deep fringes of space and reduce your gluttonous intake of caloric maze candies. Now you can work off those extra pounds and breathe clean fresh air right here on Earth at your local golf course retrieving golf balls.

Preppie!, an arcade game for one or two players, embodies all the arcade game characteristics that take a player beyond mere enjoyment to truly enthusiastic excitement. The author, Russ Wetmore, is to be highly commended for the thoroughness and detail of his programming efforts. This program easily ranks among the best games to appear for the Atari computer to date.

Superb Graphics

Preppie! fully exploits the Atari graphics capabilities. I have seen no other game use as many different colors as this one. The detail given the objects is superb. The golf carts have steering wheels and bumpers. The treads of the bulldozers rotate. The alligators have wrinkled skin and sharp, white teeth. The logs display growth rings and peeling bark. The frog extends his limbs when leaping. Preppie's knees flex when he jumps. He even has the obligatory knit emblem on his shirt. The blades of the lawnmowers rotate. Even the title page and scoreboard are unique. Add all these details together, and you get a graphics display that commands everyone's attention.

As if graphics weren't enough, there is some fine music as well. As your Preppie moves, a bell rings to indicate passage from one line to the next and increases in your score. Should your Preppie be mauled by a marauding mower or poisoned by the infamous frog, a short funeral march is played. Falling into the water elicits yet another sound, as does retrieving or returning a golf ball from the rough.

Deadly Water Hazards

Enough aesthetics. How do you play *Preppie!*? The object of the game is to maneuver your Preppie across the fairway and the treacherous river into the rough, retrieving golf balls and returning them safely to the greens. You begin play by selecting one or two players. If two are playing, you have the option of using one or two joysticks. Now press the START key and begin the fun.

You start with three Preppies. Your journey is fraught with many dangers. You must negotiate a path that avoids contact with speeding golf carts, razor-sharp lawnmowers, killer bulldozers, and a poisonous frog. Then you must ford the river using moving boats, logs, and alligators as stepping stones. Should you survive to this point, you will find yourself in the rough, where most of the golf balls will appear. Pick up a golf ball, but be careful you don't fall into a water hole.

Now you must make your way back to the green via the same mobile hazards as before. Upon retrieving all the golf balls on the green, you will be advanced to the next higher level and will increase your score according to the amount of time increments remaining on the timer bar. Should the timer expire before you complete your mission, you lose a Preppie and must repeat the level you are on.

There are ten levels of play in *Preppie!*. You may start the game at level one or choose a higher level up to nine by pressing the appropriate number on your keyboard before hitting the START key. Each successively higher level increases the speed of the hazards. At certain levels you will be confronted with new hazards or an increase in the number of golf balls, up to four. Level ten can be reached only after you have successfully completed level nine. An extra Preppie is awarded when you reach a score of 8000.

Be sure to take the time to read the short story at the front of the instruction book. It gives you an uproariously entertaining and irreverent view of what a Preppie is.

The most surprising aspect of *Preppie!* is that Adventure International is the marketer of the program. AI, famous for its excellent text and graphics adventure programs, makes its first excursion into the world of arcade gaming with *Preppie!*.

The disk version contains a high-resolution picture not found on the tape version at the opening of the game. Disk users are offered a chance to purchase a back-up copy for $3.99.

If you are looking for a game that will please all ages, *Preppie!* is it. I have used this game in my school classrooms with kids ranging in age from five to thirteen, and they all loved to play it. It has replaced *Pac-Man* as their favorite game. Get *Preppie!*. It will entertain the kids and impress your friends.

Preppie!
Adventure International
Box 3435
Longwood, FL 32750
16K Tape
32K Disk
$29.95

Player ZX81

A Tune-Playing Program For The Sinclair/Timex

Arthur B. Hunkins, School of Music, University of North Carolina, Greensboro

Player ZX81 is a 1K tune-playing program available in versions for all Sinclair/Timexes (including 4K ROM). Although each cassette is specific to a single version, the six pages of instructions cover them all. For anyone interested in coding tunes into the Sinclair and playing them back – over and over again if you wish – the $6.95 spent on this program is a bargain.

This is especially true considering the fact that you also get *81 Space Muse-AK*, a program that randomly generates and plays 127 pitches and rhythms, generates another set, etc. In short, it's guaranteed to drive you and your neighbors batty – a kind of mindless musical autopilot.

The procedure in *Player ZX81* is to code a duration number (all durations are relative) and pitch value (from the handy pitch table) for each note. You can have, even on a 1K machine, up to 127 notes. The limitations are these: 1) The single tone color available is a square wave; 2) Only pitches from the B above middle C on up are available (i.e., *high* pitches); 3) You can't easily do rests, but I'll describe a fix for this presently.

The external hardware required is a high-gain amplifier/speaker. A Radio Shack mini- or telephone amplifier/speaker at $10-$12 will do the job nicely. Or you can use a regular hi-fi amplifier, connecting to its phono input.

Versatile Modifications

Several program modifications lend further versatility.

All modifications start with two steps: 1) Omit the recommended protecting of high memory while making the modification; and 2) Immediately after loading, POKE 16544,28 and POKE 16600,28 – this renders the BASIC code accessible.

One thing you may want to do is to relocate the note table – according to how much memory you have and what you may want to add to the program. First, POKE 16549 with the same value you'll POKE into 16389, namely the page number for the beginning of the table. The lowest possible number is 67, the top page of 1K. Each additional K of memory is four pages; for example, the top page of a 2K Timex is 71. Whatever starting page you choose, change the memory location in statements 105 and 120 to 256*page number, and to this value +1 in line 117.

If you want a single play, instead of infinite repeat, delete statement 150. Or insert a PAUSE statement, a do-nothing FOR/NEXT loop, between 140 and 150 in order to space the repeats. Or again, use a statement 145 IF INKEY$ = " " THEN GOTO 145, to wait until you press a key.

If you have more than 1K memory, you may wish to use it to code longer songs. What you have to change here is line 95. The principle is this: a page of memory holds 127 notes; two pages hold 255. Subtract one from these numbers, and multiply by two to get the loop value to plug in line 95. Given available memory, there is no limit to the number of notes you may specify.

I mentioned the problem of rests. Here we have to modify the machine language routine itself, by changing statement 10 (in the process you can lop off the final zero in the statement, if you like). Be sure to follow these instructions *precisely*; any mistakes or omissions will cause a system crash. The idea is to reserve the frequency value of 255 as a flag for the rest specified as the duration value. In other words, every time a frequency of 255 is specified, you get a rest instead.

Bring statement 10 down to the bottom of the screen for editing. Locate the <= character. Add the following sequence of three characters immediately *before* <=: a lower left quarter square graphic, a C, and an upper right quarter square graphic. (Graphics characters are shifted, and must be preceded and followed by pressing the graphics key.) Then, after the RETURN that follows <=, insert a left half square graphic. In addition, change the following three characters: LOAD to FOR, DIM to FAST, and the last PEEK to INT.

INT is a function, and is prepared by pressing the function key. FOR and FAST are keywords, and are registered by pressing THEN, the keyword, and finally deleting THEN. Double check the code, and SAVE the program before RUNning it. If all is well, a frequency of 255 should now produce a (clickless) rest!

Finally, I recommend substituting the following list of "Pitch Nos." for those given by the author. Note that names duplicate Mr. Maples', but add two pitches on the top end: 250, 235, 222, 210, 198, 186, 176, 166, 157, 148, 139, 131, 124, 117, 110, 104,, 98, 93 or 92, 87, 82, 77, 73, 69, 65, 61 (B2), and 58 (C3). These values have been checked with a frequency counter.

Player ZX81 is a most useful program; the above modifications can make it even more so. Anyone interested in a "voice" for the Sinclair/Timex will find this a real value.

Player ZX81
Wm. Maples
688 Moore St.
Lakewood, CO 80215
$6.95

©

PET/CBM Standard Terminal Communications Package

Harvey B. Herman, Associate Editor

"Why did you buy a personal computer?" I have been asked this question several dozen times. **COMPUTE!** readers, as a group, should be able to reply with a dozen answers. One of my answers, "I wanted to use it as a terminal to communicate with other computers," would probably be a popular response.

How so, you ask? Well, the basic ingredients of a terminal are present in any small computer, e.g., a keyboard and a display screen. Thus, by paying a little more for terminal software and hardware, the user has the best of both worlds. That is, the computer can be used standalone for games or word processing, but also for accessing bulletin boards and large data bases over the telephone line.

Where does one find a good terminal program? There are several choices for PET/CBM computers, ranging from gratis (only a copying fee) up to $300 or more. However, the free program, attractive as it may sound, does require construction of hardware, which may be beyond many people's abilities.

The *Standard Terminal Communications Package*, reviewed here, is priced intermediately between those two extremes. It comes with all necessary hardware and software, ready to run on any Commodore system (except the 2001 PET with Original ROMs). The hardware, based on a 6850 ACIA chip, plugs easily into an empty ROM socket; clip on a few wires, connect your modem, and you're ready to run the terminal software supplied on disk. (Specify what computer and disk drive in your initial order to receive the proper version.)

The initial program loaded is written in BASIC, and is used to load, partially configure, and run the machine language terminal program. For example, you are prompted for the printer device number and the communication format (device four and eight bits-no parity, in my case).

Fifteen Different Options

The terminal program also offers a full configuration menu. You can specify full- or half-duplex, set a timer, toggle a printer on or off, etc. I counted 15 options. A status line at the top of the screen informs the user of the current state of the program, just as a much higher-priced, dedicated terminal would.

This package has some very attractive features. The best one is the ease with which it uploads and downloads BASIC programs. I have previously published in **COMPUTE!** two "how-to" articles on this subject. This program simplifies that process to the point where anyone, even a complete novice, can do it. So much for my arcane knowledge!

There are a number of minor points which I didn't like:

1. No end-of-line bell.
2. No option for line feed with an ASCII printer.
3. Does not always tokenize "IF" during downloading.
4. The manual did not make it clear that the modem must be off when configuring.
5. The delete key sends a backspace (change $23DE from $08 to $7F, if desired).

Let there be no mistake – this is a very worthwhile package. It is reasonably priced and is offered by a very reputable company which promises updates and program maintenance. Do not underestimate the importance of this service, particularly if you don't have the time or expertise to do it yourself. My wife and I have been using *STCP* regularly since it was received for review. Unlike most people, we have a choice of terminal software, and the fact that we continue using this program is our highest recommendation.

Standard Terminal Communications Package
Eastern House Software
3239 Linda Drive
Winston-Salem, NC 27106
$129.95

©

A Financial Wizard For Atari

Tina Halcomb, Carrollton, TX

If you want to use a finance system, but don't want to spend several days trying to learn how to use one, then *A Financial Wizard* by Computari of Dallas, Texas may be just what you need.

A Financial Wizard is an autoboot program that requires an Atari 400 or 800, an 810 disk drive, and at least 24K of memory. A printer is optional. It works with the Epson with Graftrax, NEC, Prowriter or Centronics 739.

The illustrated manual that comes with this program is clear, direct, and very thorough. It won't take long to get the system set up and working for you. Procedures appear in the manual in the order in which they should be followed, so you can just learn as you go. On the back cover of the manual you will find a (tear-out) command reference card.

To prevent costly errors that will result in data damage and/or loss, error warnings appear throughout the manual and on the screen during program execution.

It appears that this finance system was designed to achieve the best and most comfortable working relationship between the user and the program.

The Check Entry routine (item #1, main menu) is the most attractive feature of this finance system. Data prompts are very clear, and the category item names are displayed at all times during data entry for your convenience. It keeps track of the number of the last check entered and displays it for you. The balance of your checking account is kept current as checks are entered.

Also, if you make a mistake while recording your checks, you can correct it immediately. You are given the ability to split one check into more than one expense category. If you write a check at your bank for cash and want to account for everything it was used for, then you will be allowed to itemize it using this feature; and you will want to use it if you are striving for accuracy in your accounting. Scanning your entries is made possible by pressing START. You can see records very quickly this way. The correction capability is offered during scanning also.

Menu item #2 – Budget Entry allows you to set up a projected budget on a monthly basis. The category names provided by the program are generally used by almost all of us. However, if Spot is sick more often than you are, you could change the "Medical" category to "Vet" with the category change routine in the Utilities program.

The same applies to all categories. If your budgeted allowance for one category is the same for more than one month, you can replicate a budget from one month to the next by entering a "/" next to the category letter. An entire budget plan can also be replicated when you start to set up a new month by entering a "C" and the number of the month you want to copy from in the salary block.

Superior File Searching

The file search capabilities of this program are superior. You are offered seven ways to look up the checks. You may search by Name of Payee, Category Name, Sub-Category Name, a range of Check #'s, a range of Months, a range of Days, and a range of Amounts.

The Tabulation section figures what percentage of your salary is spent on each category. This can be very revealing. The Bargraphs clearly display what you have spent with respect to your planned expenses. A thick colored bar represents your expenses and a thin contrasting line overlays this bar to show your budgeted amounts. You have a choice of seeing either one month's expenses in all categories or one category shown over a 12-month period.

The Checkbalancer routine follows the standard procedure shown on the back side of your bank statement. Marking cancelled checks has been made extremely simple. The checks are addressed sequentially and displayed on the screen for you. To mark a check that has cleared your bank, you type an asterisk next to it. If you accidentally cancel a check that has not cleared the bank, you can reverse it by typing an "X".

The Checkwriter routine will print your checks for you, with custom checks available through Abacus Software.

The Utilities program, item #8, consists of seven utilities that you will need to maintain accurate records. With this program you will transfer records to a permanent storage disk at the end of a financial period and prepare the disk for the new year.

This system is disk intensive. All data is saved automatically and immediately following all routines that either enter data or modify it.

Overall, this is an excellent finance system – entertaining, accurate, and fun to use.

A Financial Wizard
Available From:
On-Line Computer Center
10944A N. May Avenue
Oklahoma City, OK 73120
(405)751-2701
$59.95

(An earlier version of *A Financial Wizard*, known as *Personal Finance for the Atari*, may be upgraded for $10. Users should return their master disk and a check or money order to On-Line Computer Center.) ©

Automate Your Atari

Joseph J Wrobel, Rochester, NY

The Atari Disk Operating System (DOS) supports the use of a file named AUTORUN.SYS that has a very special characteristic. At system start-up, the DOS loads and runs this file automatically if it exists on the mounted diskette. This allows you to arrange for your Atari to come up smart.

The Potential

The AUTORUN.SYS file could contain a machine language program that loads and runs. It could also contain just a short program to do some routine operations like setting the screen margins or color before passing control to BASIC. However, the major use I've seen for AUTORUN. SYS is to direct the system to load and run a BASIC program. Not only does this type of operation save you some time and effort, but it also allows an unskilled operator, like a student, to turn on the machine and interact with an application program without getting into the details of the LOAD or RUN instructions.

The Problem

So far, so good. Why doesn't everyone use the AUTORUN.SYS file? Apparently the major obstacle to its more widespread use is the fact that it is a machine language routine. Thus, it requires knowledge of 6502 machine language and, for complex operations, some knowledge of the intricacies of the Atari Operating System to create a functional AUTORUN.SYS file. Unless someone were to come up with a program to do it for you.

Automate (Program 1) is just such a program. If you key in this program correctly and run it, Automate will help you create your own personal AUTORUN.SYS file, and it won't hurt a bit. The program starts by asking you to input the series of commands you wish to be executed at start-up. You enter the commands exactly as you would if the machine came up in its normal ready state. The only limit on the number of commands is that the total number of characters entered may not exceed 196 (including the Atari end-of-line character added each time you hit RETURN). The program keeps track of the number of characters entered and will prevent you from exceeding this limit. After you've entered the final command in the sequence, the program will create an AUTORUN.SYS file on the mounted diskette. Note that any previous AUTORUN.SYS file will be overwritten by this operation.

The next time you boot up from the diskette bearing the AUTORUN.SYS file, the AUTORUN.SYS program will be run. This will cause the commands you entered to be executed in the order they were entered (although they will not be displayed), then control will be returned to the system. The commands, of course, must be compatible with the cartridge in use (BASIC, Assembler Editor, etc.) or an error will result. If at any time you wish to boot up from a diskette and circumvent the AUTORUN.SYS file, just hold the OPTION key down until system initialization is complete. The AUTORUN.SYS file created by Automate checks that key and, if it finds it depressed, the command list will not be executed.

A BASIC Example

To demonstrate the use of the program, a single command BASIC example will be presented. Let us suppose there exists a BASIC program entitled BEGIN which you would like to run automatically at start up. Using AUTOMATE, you enter (as Command #) the statement:

```
GR.0:?"Autoboot in progress.":RUN"D:BEGIN"
```

then press RETURN. Assuming you entered the command correctly, you respond to the question:

```
Is that correct (Y/N)?
```

by pressing Y. When the program asks if there are:

```
More commands (Y/N)?
```

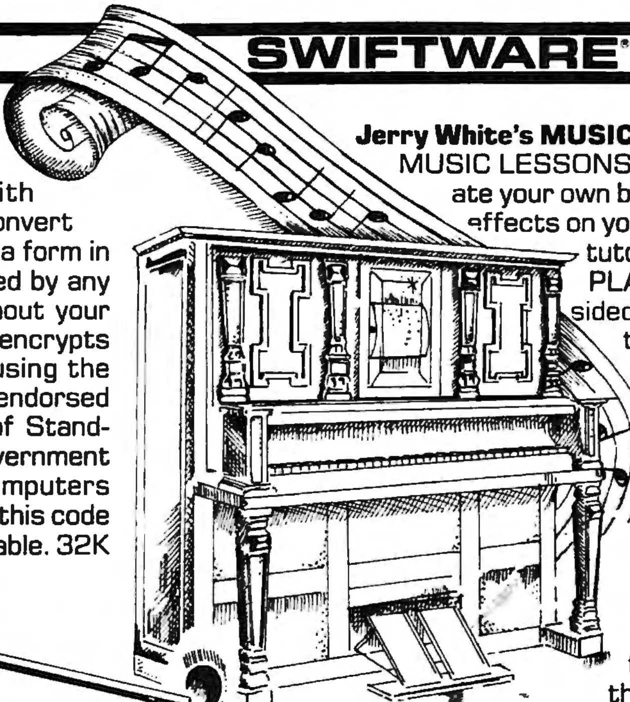
respond by pressing N. The program then creates the AUTORUN.SYS file and displays READY when it's done. If you now turn off your computer and switch it on again, you will find that it "comes up" running program BEGIN. How simple can you get?

Description Of Operation

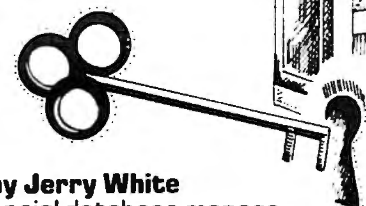
This section is for those who are not satisfied with just running the program, but are also interested in knowing how it works. Let's first take another look at Program 1. Automate consists of three major sections. The first section (lines 50 through 130) are for documentation and initialization. The program employs two key numeric variables: I, which counts the number of commands entered, and L, which counts the total number of characters in the command list. The second program section (lines 140 through 350) INPUTs the commands one at a time. As each command is entered, the program allows for error correction, checks command list size, packs the command into B$ and tacks on an Atari end-of-line (EOL) character, namely CHR$(155). The third section of the program (lines 360 through 600) actually creates the AUTORUN.SYS file.

Before this third section is discussed, I direct your attention to Program 2. This is the assembly

SWIFTWARE®



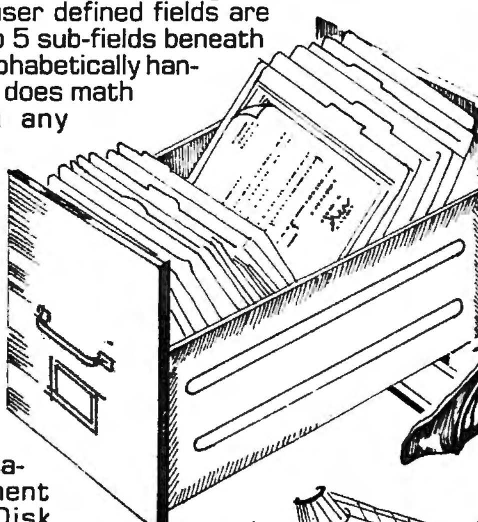
CRYPTO 800
Protect your valuable programs and data! With CRYPTO 800 you can convert your programs or data to a form in which they can not be used by any unauthorized parties without your secret key. CRYPTO 800 encrypts and decrypts your files using the data encryption standard endorsed by the National Bureau of Standards for use by many government agencies. The fastest computers would take years to break this code making it virtually uncrackable. 32K Disk. $39.95



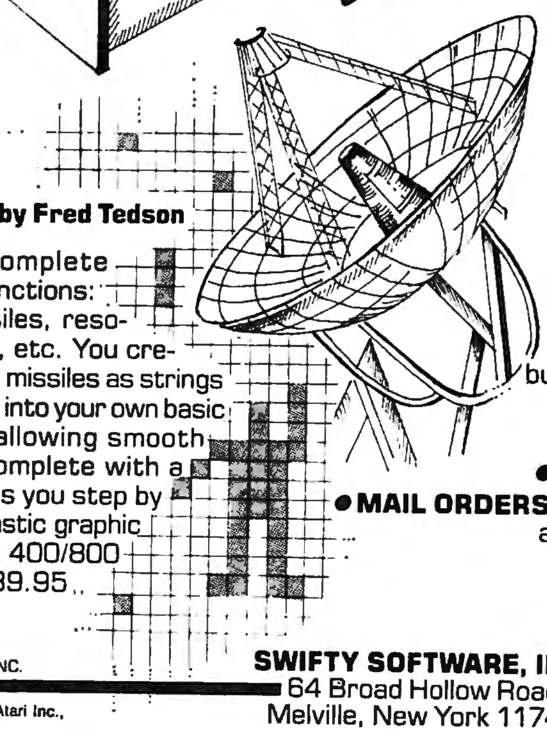
Jerry White's MUSIC LESSONS
MUSIC LESSONS has everything you need to know to create your own beautiful music and a wide range of sound effects on your ATARI computer. This comprehensive tutorial contains 13 separate programs and PLAYER PIANO on 2 cassettes or 1 double sided disk and includes extensive documentation complete with program listings. 32K DISK 16K CASSETTE (24K required for PLAYER PIANO). $29.95

SPACE SHUTTLE by Paul Kindl
Join the crew of the *Space Shuttle* as they prepare to take the next step into the world of space travel. Take control of the world's first reuseable spaceship, the *Space Shuttle*, and in an accurate full graphic simulation, place yourself in the cockpit. Pilot the *Space Shuttle* through take-off with booster stage separations, orbit, descent down the glide path and landing to touchdown—complete with a chase plane and scrolling runway visible through the cockpit windscreen. You assume command throughout all phases of the mission aided by complete instrumentation. 32K Diskette. $29.95

FILE • IT 2 + by Jerry White
A powerful financial database management system. 6 user defined fields are created with up to 5 sub-fields beneath each main field. Alphabetically handles data and also does math computation on any selected fields. Data files are stored on separate disks with full field and subfield sorting with file merging. Supports up to 4 drives including the 128K Axlon Ramdisk. The "alternative" to more costly database management systems. 24K Disk. $49.95



HAUNTED HILL by George Richardson
Fight bats and ghosts in the dark of the cemetery. This exciting, all machine language game has arcade quality graphics and speed. Requires Joystick.
$24.95 16K Cassette/$29.95 16K Disk.



P/M 800 by Fred Tedson
The ULTIMATE Player/Missile Editor
P/M 800 gives you complete control of all graphic functions: 5 players, colors, missiles, resolution, priority settings, etc. You create and save players and missiles as strings that can be incorporated into your own basic assembler programs allowing smooth arcade type action. Complete with a users tutorial that takes you step by step, exploring the fantastic graphic capabilities of the ATARI 400/800 computer. 32K Disk. $39.95

DATALINK by Tony Dobre
Top rated by national magazine reviews, purchased by NASA, this ultra-sophisticated menu-driven multioption smart terminal communications package supports uploading/downloading in full-duplex or simplex modes. Compatible with all the commercial services and bulletin boards such as the Source and Compuserve, etc. 24K Disk. $39.95

- **AVAILABLE AT SELECT COMPUTER STORES**
- **MAIL ORDERS:** Send check or money order plus $2.50 shipping and handling. N.Y. Residents add 7¼% sales tax.
- **TELEPHONE ORDERS:** (516) 549-9141

Send for FREE Catalog • Dealer Inquires Invited

SWIFTY SOFTWARE, INC.
64 Broad Hollow Road
Melville, New York 11747

MasterCard — VISA

Great Games and Books

For Your Atari 400/800

THE PROGRAM STORE™

YOUR ATARI COMPUTER

By Poole, McNiff & Cook from Osborne/McGraw-Hill

Comprehensive, all-in-one guide for Atari 400/800 computers that is helpful to the beginning and advanced programmer. Complete operating instructions; detailed BASIC programming, including a handy alphabetical glossary of BASIC statements and functions; advanced BASIC; computer graphics; and tips on hardware, peripherals and compatible software.

Softcover, $16.95

COMPUTE!'S SECOND BOOK OF ATARI

From Compute! Books

An **all new** collection of **never-before** published articles on the Atari for beginner to advanced users. Topics include: Utilities, Programming, Graphics, Applications and Beyond BASIC.

Spiral-bound, $12.95

GAMES & RECREATIONS

From Reston

This new book is packed with games and ideas on how to create your own. You'll discover the special Atari capabilities of the keyboard, graphics, sound and color. The book provides an entertaining way to learn more about general programming, too.

Part I: Learning Through Games. Part II: Games for the Atari. Part III: The Atari Special. Plus seven appendices.

Softcover, $14.95 **Hardcover, $19.95**

MASTER MEMORY MAP

From Santa Cruz Software

If you are serious about programming the Atari, this booklet could become one of your most important tools. A highly detailed map of the Atari's memory, it details thousands of locations and routines. Using this booklet makes easier the use of display list, player/missile, and interrupt graphics. The "Miscellaneous Notes" section contains a wealth of knowledge picked up by Santa Cruz in their explorations of the Atari. Also included are notes on the new GTIA graphics chip.

Softcover, $6.95

VISICALC HOME AND OFFICE COMPANION

By Castlewitz and Chisausky from Osborne/McGraw Hill

Fifty VisiCalc models for home and office, including: investments, inventory, sales forecasts, payroll, personal net worth, home budget planning, family insurance needs. Each comes with model listings, sample printed reports and narrative.

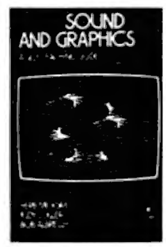
Softcover, $15.95

6502 ASSEMBLY LANGUAGE SUBROUTINES

By Leventhal and Saville from Osborne/McGraw-Hill

If you're interested in using assembly language quickly, this book is ideal. It describes general 6502 programming methods, provides code for more than 40 subroutines to help you improve your programming skills, debug or revise an existing program; add instructions and addressing modes.

Softcover, $15.95

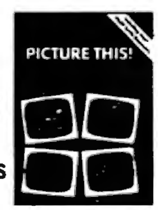


ATARI SOUND AND GRAPHICS

By Moore, Lower and Albrecht from John Wiley

This self-paced, self-teaching guide will have you seeing and hearing things on your Atari in no time—even if you're a complete beginner. You'll learn to compose and play melodies, draw cartoons, create sound effects and games. Each section teaches something new in BASIC, the most commonly used computer language.

Softcover, $9.95



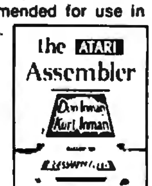
PICTURE THIS!

An Introduction to Computer Graphics for Kids of All Ages

By David D. Thornburg from Addison-Wesley

This book promises to become the "modern replacement for coloring books and crayons". It's a learn-by-doing manual that uses PILOT, a simpler language than BASIC, and Turtle Geometry to teach kids to create pictures in full color from simple lines to complex angles and curves. Recommended for use in conjunction with PILOT Cartridge.

Spiral-bound, $14.95



ATARI ASSEMBLER

By Don & Kurt Inman from Reston

While the Atari Assembler Cartridge comes with an operating manual, it assumes that you already know assembly language. If you're new to the Atari or its 6502 processor, this book is a must. The Inmans guide you through the rudiments of this fascinating type of programming in clear, easy steps. Includes full listing and description of 6502 mnemonics and addressing modes. Recommended for use in conjunction with Assembler Cartridge.

Softcover, $12.95

INSIDE ATARI DOS

By Bill Wilkinson from Compute

The comprehensive manual on the disk File Manager System (FMS), commonly known as Atari DOS 2.0S. Contains the only complete and official listing for the system, plus a full description of: the external view, charts & tables, various interfaces and functions of individual subroutines.

Spiral-bound, $19.95

DE RE ATARI

From APX

Translated from Latin, the title of this book is "All About Atari" and it means what it says! Used in combination with Atari's Technical Reference Manual, advanced programmers will be able to learn to exploit the many hardware and operating system features that make the Atari 400/800 so tremendously versatile. Includes a useful discussion of the new GTIA chip. Once you know Atari BASIC and assembler, this book is a must.

Loose leaf (binder not supplied), $19.95

COMPUTERS FOR KIDS

Atari Edition By Sally Larsen from Creative

The BASIC programming manual written for kids, from flowcharts to color graphics, including the sure-to-please program, "Scare Mom with an Elephant." Detailed instructions and sketches plus glossary of statements and commands. With lesson plans and tips for parents and teachers.

Softcover, $4.95

STIMULATING SIMULATIONS,

Atari Version, 2nd Edition

By C.W. Engel from Hayden

A handbook of 12 simulation games including Art Auction, Starship Alpha, Monster Chase and Devil's Dungeon—each complete with listing, sample run, instructions and program documentation, including flowchart and ideas for variations.

Softcover, $5.95

THE PROGRAM STORE • Dept. 10-01-3 • Box 9582 • 4200 Wisconsin Avenue, NW • Washington, D.C. 20016

Item ______ Tape/Disk/Book ______ Price ______

Postage $2.00

Total ______

☐ CHECK ☐ VISA

☐ MASTERCARD

Computer ______

Name ______

Address ______

City ______ State ______ Zip ______

Card# ______ Exp ______

listing for the core of the AUTORUN.SYS program. What this machine language program does, in a nutshell, is to temporarily take over the task of supplying screen editor data by substituting a new device handler table and "get character" routine for the default ones provided by the operating system. At system start-up while the AUTORUN.SYS program is active, it intercepts all the keyboard entry requests and feeds out, one character at a time, the commands which you have entered. When it has sent out the last character of the last command in the list, it re-installs the default screen editor handler table, and the system takes over from there.

Returning to the section of the BASIC program which creates the AUTORUN.SYS file, you will find that it consists primarily of three loops. Loop one (lines 490 through 510) PUTs the core program and its associated six byte header into the file as READ from the DATA statements in lines 430 through 480.

Note that in line 500 of Automate, two numbers are changed from the values shown in the DATA statements before putting them into the AUTORUN.SYS file. The first is a byte in the AUTORUN.SYS file header which gives the end of the program when loaded in memory. This is the sum of the core program length and the number of bytes in the command list. Automate also alters the value of the immediate argument of the CPY instruction in line 370 of Program 2. This byte is set equal to the total number of characters (including EOL's) in the command list. Loop two (lines 530 through 550) PUTs in the command list which resides in B$. Finally, loop three (lines 580 through 590) adds a twelve byte postscript to the file which provides the system with the initialization and run locations for the routine.

The BASIC program here provides an easy way to create a useful AUTORUN.SYS file. There are dozens of ways this file can be used. It doesn't necessarily have to be a serious application. For example, it's sort of fun just to start up my machine, listen to it go through its disk machinations, then see it automatically display the personalized greeting:

READY WHEN YOU ARE, J.W.!

Program 1.

```
50 I=0:L=0:MAX=196
60 DIM A$(MAX),B$(MAX),R$(1)
70 OPEN #1,4,0,"E:":OPEN #2,4,0,"K:"
80 ? "This program helps you to creat
   e"
90 ? " a personalized AUTORUN.SYS fil
   e"
100 ? "  which, following the disk bo
   ot"
110 ? "{3 SPACES}process, automatical
   ly issues"
120 ? "{4 SPACES}a set of commands th
   at you"
130 ? "{5 SPACES}specify."
140 I=I+1
150 ? :? "Please enter command #";I;"
   ."
160 ? :INPUT #1;A$
170 POKE 766,1:? :? "Command #";I;":"
   ;A$:POKE 766,0
180 ? :? "Is that correct (Y/N)? ";:G
   ET #2,X:? :R$=CHR$(X)
190 IF R$="Y" OR R$="y" THEN 220
200 IF R$="N" OR R$="n" THEN 150
210 GOTO 170
220 X=L+LEN(A$)+1-MAX
230 IF X<=0 THEN 260
240 ? :? "Command #";I;" is ";X;" cha
   racter(s)"
250 ? "too long.":I=I-1:GOTO 270
260 B$(L+1)=A$:L=LEN(B$):B$(L+1)=CHR$
   (155):L=L+1
270 ? :? "Current command list:"
280 POKE 766,1:? :? B$:POKE 766,0
290 IF L>=MAX-1 THEN ? "Command list
   is full.":? :GOTO 370
300 ? "Command list can hold ";MAX-L-
   1;" more"
310 ? " character(s)."
320 ? :? "More commands (Y/N)? ";:GET
    #2,X:R$=CHR$(X)
330 IF R$="Y" OR R$="y" THEN 140
340 IF R$="N" OR R$="n" THEN 360
350 GOTO 300
360 ? CHR$(125);
370 ? "Mount diskette which is to bea
   r"
380 ? " the AUTORUN.SYS file, then"
390 ? "  press RETURN. ";:GET #2,X:CL
   OSE #1:CLOSE #2
400 ? CHR$(125);:? "Writing AUTORUN.S
   YS file."
410 OPEN #1,8,0,"D:AUTORUN.SYS"
420 REM PUT OUT THE HEADER AND THE CO
   RE MACHINE LANGUAGE PROGRAM
430 DATA 255,255,0,6,59,6
440 DATA 173,31,208,41,4,240,10,169,1
   8,141,33,3
450 DATA 169,6,141,34,3,96,251,243,51
   ,246,33,6
460 DATA 163,246,51,246,60,246,76,228
   ,243,0,238,33
470 DATA 6,172,33,6,192,0,208,10,169,
   0,141,33
480 DATA 3,169,228,141,34,3,185,59,6,
   160,1,96
490 FOR I=1 TO 66:READ X
500 IF I=5 OR I=48 THEN X=X+L
510 PUT #1,X:NEXT I
520 REM ADD THE COMMAND LIST
530 FOR I=1 TO L
540 X=ASC(B$(I,I))
550 PUT #1,X:NEXT I
560 REM APPEND INITIALIZE AND RUN VEC
   TORS
570 DATA 226,2,227,2,0,6,224,2,225,2,
   17,6
580 FOR I=1 TO 12:READ X
590 PUT #1,X:NEXT I
600 CLOSE #1:? CHR$(125);:END
```

Program 2.

```
D01F          0100 CONSOL  =    $D01F
0320          0110 DEVTAB  =    $0320
E400          0120 OLDDHT  =    $E400
```

LEARN TO TYPE OR GET BLOWN TO BITS.

MasterType™ and Atari™ make typing a blast.

Now anyone can learn to type and have hours of fun at the same time. With MasterType. We've combined the fast-action of blow 'em up video games with the best instructional program available. You either learn to type or the aliens ZAP your spaceship into oblivion. The results? Highly motivated and enjoyable learning.

MasterType earns a ten-gun salute.

MasterType is fast becoming the best selling educational software product designed for use with the Atari 800 home computer.

It's no wonder, when you consider what InfoWorld magazine had to say:

"MasterType is an excellent instructional typing game. We had fun reviewing it, and we highly recommend it to those who want to learn typing in an unconventional but motivating way."

InfoWorld also went on to rate MasterType as "excellent" in all four of its categories.

MasterType teaches your fingers to fly.

Face it. If you're like most people, the most inefficient components on your computer are probably your fingers. But you can change all that with MasterType. In just a few easy lessons your fingers will be doing things you thought them incapable of.

MasterType. With 18 explosive learning levels. You'll either learn to type or get blown to pieces.

$39.95. Requires disk drive 32k and basic.

MasterType

Lightning Software ©

P.O. Box 11725, Palo Alto, CA 94306
(415) 327-3280

```
          0130 ;
0000      0140          *=     $0600
0600 AD1FD0 0150 INIT   LDA    CONSOL
          ;Load the console switch
   register
0603 2904  0160         AND    #$04
          ;and check for the OPTION
    key.
0605 F00A  0170         BEQ    RUN
          ;If it's pressed, branch
   to the RTS.
0607 A912  0180         LDA    #NEWDHT&
   $00FF    ;Otherwise, install the v
   ector
0609 8D2103 0190        STA    DEVTAB+1
          ;to the new device handle
   r table
060C A906  0200         LDA    #NEWDHT/
  256     ;in the appropriate place
  in the
060E 8D2203 0210        STA    DEVTAB+2
         ;device table and
0611 60    0220 RUN     RTS
          ;return.
          0230 ;
0612 FBF3  0240 NEWDHT .WORD $F3FB
          ;This is the replacement
0614 33F6  0250        .WORD $F633
          ;screen editor handler
0616 2106  0260        .WORD GET-1
          ;vector table. All the
0618 A3F6  0270        .WORD $F6A3
          ;vectors have their defau
   lt
061A 33F6  0280        .WORD $F633
         ;values except for the
061C 3CF6  0290        .WORD $F63C
         ;GET routine, which
061E 4C    0300        .BYTE $4C
         ;points to the replacement
061F E4F3  0310        .WORD $F3E4
         ;routine below.
           0320 ;
0621 00    0330 COUNTR .BYTE 0
         ;character counter
           0340 ;
0622 EE2106 0350 GET    INC  COUNTR
         ;Increment the character
0625 AC2106 0360        LDY  COUNTR
         ;counter.  Compare it with
0628 C000  0370         CPY  #ENDLST-
    BEGLST  ;the command list length.
062A D00A  0380         BNE  CONT
         ;If not equal, branch to C
   ONT.
062C A900  0390         LDA  #OLDDHT&
   $00FF   ;Otherwise, reinstate the
062E 8D2103 0400        STA  DEVTAB+1
         ;default screen editor han
   dler
0631 A9E4  0410         LDA  #OLDDHT/
    256    ;table vector at the prop
    er
0633 8D2203 0420        STA  DEVTAB+2
         ;spot in the device table.
0636 B93B06 0430 CONT   LDA  BEGLST-1
   ,Y      ;Fetch the next character
0639 A001  0440         LDY  #1
         ;from the command list and
063B 60    0450         RTS
         ;return.
           0460 ;
           0470 BEGLST
           0480 ;The command list go
    es here.
063C       0490 ENDLST .END
```

All About Commodore's WAIT Instruction

Louis F Sander, Pittsburgh

WAIT is one of Commodore BASIC's most mysterious instructions – seldom seen in programs, rarely mentioned in magazines, and nearly impossible to understand in manuals. But it's available for VIC-20, PET/CBM, and 64 users. To find out how helpful it can be for all kinds of applications (program debugging, single-stepping, even a superior form of the common pause GET K$: IF K$ = "" THEN), read on.

WAIT allows a BASIC program to communicate with hardware and with certain software external to itself. It causes PET to suspend all apparent activity on receipt of a signal from the keyboard, an external device, or the computer's internal timers. PET's normal activity resumes when the signal is removed. Thus, WAIT provides a simple means of pausing until a key is pressed, an interval ends, or contacts open or close. We'll soon get to some useful examples.

When executed, WAIT examines a selected memory location and halts the program if the location contains a specified "trigger value." The program continues if, or as soon as, any other value appears in the selected location. Optionally, WAIT can be made to ignore some of the bits in the location it is testing.

In other words, WAIT halts a program if, and for as long as, selected bits in a chosen location have one specific pattern. Note carefully: the program waits *if* a specific pattern exists, not *for* a specific pattern to appear.

WAIT's format is:

```
WAIT ADDR, MASK, TRIG
```

ADDR, MASK, and TRIG can be any numeric constants, expressions, or variables in the range 0-65535 for ADDR, and 0-255 for MASK and TRIG. TRIG and its leading comma may be left out of the statement if desired, in which case TRIG defaults to zero.

Technically speaking, the WAIT statement reads the status of memory location ADDR, exclusive ORs it with TRIG, then ANDs the result with MASK, repeating these steps until a nonzero result is obtained. *Practically* speaking, few human minds can *follow* such logic, let alone comprehend its effect on their programs. If you prefer simplicity, think of WAIT as saying this: "Pause if the MASK bits in the contents of ADDR are the same as those in TRIG. Otherwise, continue." But let's illustrate some of its specific uses.

ADDR is the address of the memory location to be tested. WAIT halts the program if ADDR contains a preselected trigger value, resuming execution if and when ADDR's contents change. It follows that ADDR must be a location whose contents can change independently of the program, or there will be no way to resume program execution. Relatively few memory locations meet this criterion – mainly they are associated with the keyboard, the user and IEEE ports, and the computer's internal timers. Table 1 is a partial listing of such locations.

MASK determines whether WAIT tests all, or only some, of the bits in ADDR. If a given bit in MASK is set to one, the corresponding bit in ADDR will be tested. Otherwise, the bit will be ignored. If the entire contents of ADDR are to be tested, MASK must equal 255; any lower number will cause WAIT to ignore one or more bits. The various powers of two are often used in MASK, to monitor a single bit for a one or a zero. Zero is a legal value for MASK, but should never be used, since it always causes an endless halt. (Any number and zero equals zero.)

TRIG is the value that triggers a halt. If WAIT is executed when ADDR contains TRIG, the program will stop until TRIG is replaced by another value. Of course, if MASK is blocking out one or more bits, any number whose unblocked bits are identical to those in TRIG will have the same effect as TRIG, and will cause the program to halt. TRIG's default value is zero, so when TRIG is omitted from the WAIT statement, a halt occurs whenever all the unblocked bits are zero.

WAIT has three other notable properties. First, just as PRINT can be abbreviated as "?", WAIT can be abbreviated as "W shifted A". You can use this property to save keystrokes and line space. Second, the STOP key will not terminate a WAIT. That can only be done by satisfying the logical conditions in the argument; if a programming error has made this impossible, you must reset your machine to recover. So as soon as you put a WAIT statement into a program, save a copy on tape or disk; that will save *you* if you've made an error. Finally, WAIT does not affect the jiffy clock – TI and TI$ continue during WAITs, even though the computer and the STOP key are ostensibly dead. So by using the memory locations of the jiffy clock, you can precisely control WAIT's pauses.

Real World Applications

Table 2 lists some of WAIT's uses, along with the arguments used to implement them. To demonstrate these applications, the following examples can be inserted as line 25 of this little program:

```
10 TI$="000000"
20 PRINT TI
30 GOTO 20
```

Lines 20 and 30 cause a continuous screen printout while the program is running, making it easy to observe the effects of the WAIT in line 25. (The following examples use ADDResses from Original ROMs; if you have PET Upgrade, 4.0, or a VIC or a 64, use Table 1 to find the right ADDResses for your machine.)

WAIT 59410,255,251 stops the program when SPACE is depressed, and continues execution when it's released. No other key can make the program pause if these arguments are used. Different TRIGs, of course, will activate different "59410 keys."

WAIT 59410,255,255 stops the program *until* SPACE or one of the other "59410 keys" is depressed. Only these keys can change the contents of 59410, and any one of them will do it, thus ending the WAIT.

WAIT 516,255 is a simple way to pause until a key is pressed. Since 516 contains a zero until SHIFT is pressed, this line halts the program until you press SHIFT. Notice that the STOP key has no effect unless SHIFT is pressed and the program is running. Also notice that WAIT 516,1 would have the same effect, using fewer bytes.

WAIT 59411,8,8 waits for a button on the tape drive to be pressed. While all the buttons are up, 59411's eight-bit is set, and the program halts. Depressing PLAY or any other recorder button clears the eight-bit, resuming execution of the program.

WAIT 59411,8 halts the program when the eight-bit is cleared, resuming when it's set. So, unlike the last example, this one stops when a button is *down*. Together, these two examples show how to use a one or a zero in any bit position to stop your program – just block out all the other bits and use TRIG to look for a one or a zero in the position of interest. This technique can be used to wait for a peripheral to signal that it is ready to proceed, assuming that the signal comes

Table 1: Some Useful Memory Locations

PET Orig. ROMs	PET Upgrade or 4.0 ROMs	VIC-20	64	Contents
512	141	162	162	Increments every jiffy (1/60 second).
513	142	161	161	Increments every 256 jiffies (4.2 seconds).
514	143	160	160	Increments every 65536 jiffies (18.2 minutes). Zeroing TI$ zeros all three clock locations.
515	151	197 (Note: returns keyboard matrix rather than ASCII value)	197	Unique value for the key pressed at the current jiffy. No key=255. Other values differ with ROMs.
516	152	653	653	Status of SHIFT key. Up=0, Down=1.
525	158	198	198	Number of characters in the keyboard buffer (0 to 9).
59410	59410	N/A	N/A	Senses certain keys. The keys vary with keyboards and ROMs, but these are the most common: For BASIC 1.0 and 2.0: RVS = 254; '[' key = 253; SPACE = 251; '<' key = 247; STOP = 239; '.' key = 191; '=' key = 127 For BASIC 4.0: '← ' key = 254; '3' key=253; '6' key = 251; '9' key = 247; STOP = 239; ':' key = 223 Note that pressing a key zeros a bit in 59410. Pressing multiple keys zeros multiple bits.
59411	59411	37151 (Button down=62 all buttons up=126)		The eight-bit of this location tells the state of the buttons on TAPE #1. Button down = 0, all buttons up = 1.
59471	59471	37136		Parallel User Port.

by way of a line going high or low.

WAIT 514,128 pauses until the jiffy counter hits 128. See Table 2 for other valid MASKs for this purpose (WAIT ADDR,T).

WAIT 525,1 : POKE 525,0 waits for any key to be pressed. This is simpler than the more common

```
25 GET A$ : IF A$ = "" THEN 25
```

and it allows other statements to follow it on the same program line. The POKE is there to clear the keyboard buffer and can be omitted if there are no subsequent GETs or INPUTs in your program.

WAIT 525,2 : POKE 525,0 waits for *two* keys to be hit. There is no easy way to wait for three.

WAIT 513,255,PEEK(513) waits for the 4.2 second timer to increment. The screen display will prove that this happens every 256 jiffies.

WAIT 516,1,PEEK(516) illustrates an interesting technique. Whether SHIFT is up or down,

WAITing On The VIC-20 And Commodore 64

Doug Ferguson, Elida, Ohio

Joysticks Can WAIT

One of my pet peeves involves a game that uses a joystick for virtually all movement, but when it's time to indicate whether to play again, I have to put aside the joystick and hit a function key, type Y for YES, or hit the space bar. Why not use the joystick?

End-of-the-program questions are well suited for the WAIT command. To replay or not to replay is hardly a "menu" of choices. With WAIT, the computer "waits" for the replay signal. Even if the player wants to quit, he can always RUN/STOP-RESTORE or turn off the power.

The most suitable replay signal is the fire button, as in this VIC-20 example:

```
6000 PRINT "YOU WIN!!": PRINT" PRESS
     FIRE-BUTTON TO PLAY AGAIN"
6005 WAIT 37137,32: REM IN CASE BUTTON
     IS ALSO USED IN THE GAME ITSELF
6010 WAIT 37137,32,32
6020 RUN
```

Here is a table showing the specific test values not only for the VIC-20 but also for the Commodore 64:

		COMMODORE 64	
	VIC-20	Joystick 1	Joystick 2
FIRE	WAIT 37137,32,32	145,16,16	56464,16,16
LEFT	WAIT 37137,32,32	145,4,4	56464,4,4
DOWN	WAIT 37137,16,16	145,2,2	56464,2,2
UP	WAIT 37137,4,4	145,1,1	56464,1,1
*RIGHT	WAIT 37152,128,128	145,8,8	56464,8,8
ANY (except RIGHT on the VIC)	WAIT 37137,62,62	145,31,31	56464,31,31

*POKE 37154,127 before and POKE 37154,255 after the WAIT statement on the VIC.

This table assumes you want to test if the joystick is *pressed* a certain way. If you want to test that a certain position is *not pressed*, just leave off the last number (as in line 6005).

Tracing With WAIT

Another way to use WAIT is in FOR/NEXT loops in either program or direct mode. For example, to examine the contents of the ROM memory containing BASIC, type in the following program:

```
100 FOR X = 12 * 4096 TO X + 81
    91: PRINT X,PEEK(X)
110 WAIT 197,64
120 NEXT
```

or the direct statement

```
FOR X = 12 * 4096 TO X + 8191:
    PRINT X,PEEK(X):WAIT 197,64:
    NEXT
```

(In both examples, substitute 10*4096 for the Commodore 64.)

A list of memory addresses and contents will begin to scroll by. To stop printing, press any key (except RESTORE, SHIFT, CTRL, or the Commodore key). Printing resumes when the key is released. If the WAIT is changed to WAIT 653,1,1, the SHIFT key alone becomes the control key. This has the advantage of providing a "hands off" pause by using the SHIFT LOCK key.

It is also possible to single-step (go through a program line by line) using the WAIT command. Simply change the WAIT to

```
WAIT 197,64: WAIT 197,64,64
```

for "any key" control or

```
WAIT 653,1,1: WAIT 653,1
```

for SHIFT key control, although the SHIFT LOCK is of no consequence when single-stepping.

Escape from examining memory by hitting the RUN/STOP key.

NOW. The only real limitation for your VIC 20 is imagination.

The ARFON MICRO VIC 20 EXPANSION CHASSIS allows you to fully expand the VIC 20 memory, plug in interfaces, other computer peripherals, cartridges for expanded Basic language functions, programming utilities and even ROM cartidges of your own design to turn the VIC 20 into a sophisticated computer control system. In fact, with your VIC installed in the AFRON MICRO VIC 20 EXPANSION CHASSIS the only real limitation to the uses you can find for the VIC 20 is imagination.

VIC 20 SOFTWARE FROM ARFON MICRO, U.S.

TOTL TIME MANAGER is a set of two programs which allow you to create personal or business schedules, calendars of events, and checklists of activities. Organize reports by: person, project, or activity. Sort reports by: beginning or ending dates, or activity number. Print 56 different bar chart formulas. $25.

RESEARCH ASSISTANT 2.0 is a set of programs which allow you to keep track of reference data and create keyword cross reference lists. Keep data on reference sources: author, title, bibliography. Keep reference notes: page(s), text, up to 12 keyword cross reference, print data and or cross reference lists. $25.

TOTL LABEL 2.0 is a very useful mailing list and label program. TOTL LABEL 2.0 features easy editing, add or delete labels, define your own labels (width, length, number of printed lines), sort alphabetically or numerically and more. $20.

TOTL TEXT 2.0 is a complete word processing program which allows you to create and format professional looking documents. There is no limit to the length of a document. TOTL TEXT features: page numbering, paragraph control, right and left margin control, VIC printer expanded characters, skip to top of page, single, double, or triple spacing, centered title lines, tab position control, character and line spacing, upper and/or lower case, use of graphic characters, full screen editing, full cursor control while editing, scroll up and down through text, add, change and delete characters, insert and delete blocks of text. $25.

TOTL TEXT 2.5 has all the features of TOTL TEXT 2.0, plus up to 4 heading lines per page, footing line every page, footnotes, keyboard imput for form letters (up to 6 80-column lines), special characters for printer, right justify, and 3K additional working memory for editing. $35.

ALL TOTL 2.0 series programs work with tape and/or disk and require VIC 20, 8K expansion, cassette deck and/or disk drive, VIC printer or RS-232 printer.

BALDOR'S CASTLE is a fast-moving real time adventure—can you fight off 11 different types of monsters with just your bare hands, or will you need bow and arrow, magic sword, potions, and more to steal Baldor's gold? Game cartridge by Martin Kennedy. (More fun with joystick.)

ARFON VIC EXPANSION CHASSIS comes complete with aluminum cover/monitor shelf.

TOTL TEXT AND TOTL TIME MANAGER NOW AVAILABLE FOR THE COMMODORE 64

Ask your VIC Dealer, or Contact
ARFON MICROELECTRONICS, U.S.
111 Rena Drive, Lafayette, La. 70503
(318) 988-2478

this line waits for it to change. (Careful – if you changed ADDR to accommodate your ROMs, the PEEK must be changed to match it.)

WAIT 515,255,PEEK(515) does the same for any other key.

WAIT 59471,1,1 waits for the PA0 line on the user port to go low. Don't try this or the following examples unless you've configured the port for inputs and can control the lines.

WAIT 59471,1 waits for PA0 to go high.

WAIT 59471,1,PEEK(59471) waits for PA0 to change state.

WAIT 59471,3,2 waits if PA0 is low and PA1 is high. Otherwise, the program continues to run.

Of course, there are hundreds of other ways to use WAIT. If you understand the ones we've looked at here, you're ready to find and exploit the others. Here's one of them, to check your understanding: If X means we don't care whether a bit is 0 or 1, and if we want our program to pause as long as XXXX0101 appears in location 59471, but to continue on any other value, the proper statement is WAIT 59471,15,5. If you understand exactly why 15 and 5 are the proper arguments, you really *do* know all about WAIT.

Table 2: Some Useful Applications

ADDR is the memory location to be tested.
CONT is ADDR's contents when tested.

Argument	Effect
WAIT ADDR,255,N	For N of 0-255, waits while CONT = N. Continues when CONT does not equal N.
WAIT ADDR,255,255	Waits as long as all bits in CONT are ones. Continues when any bit goes to zero.
WAIT ADDR,255	Waits as long as all bits in CONT are zeros. Continues when any bit goes to one.
WAIT ADDR,B,B	Where B is 1,2,4,8,16,32,64 or 128, waits while CONT's B-bit is one. Continues when the B-bit is zero.
WAIT ADDR,B	Where B is as above, waits while CONT's B-bit is zero. Continues when the B-bit is one.
WAIT ADDR,T	Where a counter in ADDR cycles from 0 to 255, and where T is 128, 192, 224, 240, 248, 252, or 254, waits while CONT is less than T. Continues as soon as CONT = T.

Apple Machine Language Memory Aid

K Lourash, Decatur, IL

"ML Helper" is a utility developed to assist fledgling Apple machine language programmers in studying 6502 object code when the original source code is not available, and also in adapting that code to their particular needs and systems. This program also works as is on OSI and can easily be modified for any Microsoft BASIC.

Options are offered in this program to list and modify zero page usage, to list and modify absolute addressing references, and to relocate the code under examination. Although written in Microsoft floating-point BASIC, this utility is readily converted to the other popular dialects. In fact, while my system is OSI, the listing is for Apple simply to involve a wider audience.

You may save ML Helper without REMarks. If you do, notice that line 31 may be incorporated into line 29, and line 35 into line 33, for increased program optimization. However, do not tamper with the "NEXT A" statement of line 51, since ML Helper will exit a loop without completing it; a simple "NEXT" there is insufficient.

In the interest of brevity, I chose to do no error checking of input from the keyboard. Thus it's easy to become careless and obtain seemingly inexplicable program performance. Also, when using hexadecimal notation, I assumed you won't prefix an address with the "$" symbol. Furthermore, leading zeros are harmless, but quite unnecessary. No relocate is foolproof. Hence, ML Helper does not resolve the indirect JMP or the technique of jumping with an RTS once the stack has been prepared. In other words, jump tables and data blocks are moved unchanged.

Disassemble And Relocate

When up and running, ML Helper emulates a disassembler, examining the address range you've specified for valid 6502 operation codes. When they are found, the program logic proceeds to list or modify the zero page references, to list external absolute references, to modify absolute references, or to move code and modify addresses for a successful relocate, whichever option is operative.

Bytes determined to be invalid instruction code sequences are assumed to form data tables. A data table finder, as such, is always active and can actually become an unspecified sixth option to locate unknown data table areas.

At this point I set an arbitrary criterion – namely, that wherever there occurs a block of six or fewer consecutive bytes of executable code, the data table finder should, nonetheless, report that block of code as part of a data table area. If this standard proves unsuitable for your requirements, then change the "A-7" expression in line 350. The absence of data tables is reported as an address range of 0-0 ($0000-0000 hexadecimal).

Menu item 4 may not be immediately clear. The "EXTERNAL" references that ML Helper will list are those absolute addresses referencing memory outside the body of the program module being examined. Displaying all absolute addressing usage produces a counterproductive volume of screen clutter which I thought best to avoid.

Menu item 5, by which you elect to change absolute references, is not similarly restricted. If during a run it appears that interesting data might scroll away, then Apple users are reminded to invoke the CTRL S Stop-List feature of their system; others may have to rely on CTRL C or divert all output to hard copy. Have fun exploring uncharted machine language programs with ML Helper pointing the way.

```
0 DATA 232,200,202,136,72,104,24,56,96,170,1
   68,138,152,234,10,74,42,106,186
1 DATA 154,64,120,88,184,248,216,8,40,0,208,
   240,144,176,48,16,80,112,169,162
2 DATA 160,201,224,192,105,233,41,9,73,165,1
   66,164,133,134,132,230,198,197,228
3 DATA 196,101,229,36,37,5
10 DATA 69,38,102,6,70,181,182,180,149,150,14
   8,246,214,213,117
11 DATA 245,53,21,85,54,118,22,86,177,145,209
   ,113,241,49,17,81,161,129,193,97
12 DATA 225,33,1,65,32,76,108,44,173,174,172,
   141,142,140,238,206,205,236,204
13 DATA 109,237,45,13,77,46,110,14,78,189,190
20 DATA 188,157,254,222,221,125,253,61,29,93,
   62,126,30,94,185,153,217,121,249
21 DATA 57,25,89: GOTO 530
30 REM *** LIST ADDRESSES ***
40 IF A(Z) >  = S THEN  IF A(Z) <  = E THEN
   RETURN
50 IF Z = 0 GOTO 80
60 FOR X = 0 TO Z - 1: IF A(X) = A(Z) THEN  R
   ETURN
68 S(T) = VAL (H$):E(T) = VAL (E$)
70 NEXT
80 PRINT "ADDR REF'D:  ";: IF H THEN D = A(Z)
   : GOSUB 220: PRINT "$"H$: GOTO 100
90 PRINT A(Z)
```

```
100 Z = Z + 1 - (Z > 29): RETURN
110 REM *** ZERO PAGE CHANGE ***
120 FOR I = 0 TO X: IF C(I) = A(Z) THEN POKE A
    + 1,D(I)
130 NEXT : RETURN
140 REM *** RELOCATE ***
150 IF A(Z) < TS OR A(Z) > TE THEN  RETURN
160 I =  PEEK (A + 1) + T3: IF I > 255 THEN I ~
    = I - N:T4 = T4 + 1
170 POKE A + 1,I: POKE A + 2, PEEK (A + 2) + T
    4: RETURN
180 REM *** CHANGE ABSOLUTE ADDR ***
190 FOR I = 0 TO X: IF C(I) = A(Z) THEN K =  I
    NT (D(I) / N)
195 POKE A + 1, D(I) - N * K: POKE A + 2,K
200 NEXT : RETURN
210 REM *** DEC-HEX ***
220 H$ = "":F = 4096: FOR J = H TO 4:K =  INT ~
    (D / F):D = D - K * F
225 H$ = H$ + MID$ (G$,K + H,H):F = F / 16: NE
    XT : RETURN
230 REM *** HEX-DEC ***
240 D = 0:F = H: FOR J = LEN (H$) TO H STEP - ~
    H:M = ASC ( MID$ (H$,J,H )) - 48
245 D = D + F * (M - 7 * (M > 9)):F = 16 * F: ~
    NEXT : RETURN
250 REM *** PRINT DATA TABLES ***
260 PRINT "DATA TABLE:  ";: IF H THEN D = T1: ~
    GOSUB 220: PRINT "$"H$"-";:D=T2
265 GOSUB 220: PRINT H$: RETURN
270 PRINT T1"-"T2: RETURN
280 REM *** MAIN ROUTINE ***
290 FOR A = S TO E
300 REM *** SKIP DATA TABLES ***
310 FOR I = 0 TO T: IF S(I) THEN IF A > = S(I)
     THEN A = E(I) + 1:S(I) = 0
320 NEXT : FOR I = 0 TO 150: READ M: IF PEEK (
    A) = M GOTO 390
330 NEXT
340 REM *** PRINT DATA TABLES ***
350 IF A - 7 > T2 THEN IF T1 THEN GOSUB 260:T1
     = A
360 IF T1 = 0 THEN T1 = A
370 T2 = A: GOTO 510
380 REM *** 1-BYTE IGNORE ***
390 IF I < 29 GOTO 510
400 REM *** 2-BYTE IGNORE ***
410 IF I < 48 GOTO 500
420 REM *** ZERO PAGE ***
430 IF I > 102 OR C > 2 GOTO 470
440 IF C < 3 THEN A(Z) = PEEK (A + 1): ON C GO
    SUB 50,120
450 GOTO 500
460 REM *** 3-BYTE ***
470 IF I < 103 GOTO 500
480 IF C > 2 THEN A(Z)=PEEK (A + 1) + PEEK (A ~
    + 2) * N: ON C-2 GOSUB 150,40,190
490 A = A + 1
500 A = A + 1
510 RESTORE : NEXT A: GOSUB 260: END
520 REM *** END OF MAIN ROUTINE ***
530 PRINT "1= LIST ZERO PAGE REFERENCES":PRINT
    "2= CHANGE ZERO PAGE REFERENCES"
531 PRINT "3= RELOCATE": PRINT "4= LIST EXTERN
    AL ABSOLUTE REFERENCES"
532 PRINT "5= CHANGE ABSOLUTE REFERENCES": PRI
    NT: PRINT "CHOOSE ONE: ";: GET H$
533 PRINT H$:C = VAL (H$):PRINT :PRINT "WANT H
    EX  NUMBERS, Y/N? ";: GET H$:PRINT H$
540 PRINT :H = H$ = "Y":N = 256:G$ = "01234567
    89ABCDEF": DIM A(30)
541 INPUT "INPUT START,END ADDRESSES: ";H$,E$:
     PRINT
542 IF H THEN GOSUB 240:S = D:H$ = E$: GOSUB 2
    40:E = D: GOTO 560
550 S = VAL (H$):E = VAL (E$)
560 IF C < > 3 GOTO 660
570 INPUT "INPUT TARGET ADDRESS: ";H$: PRINT :
     IF H THEN GOSUB 240:TS=D:GOTO600
580 TS = VAL (H$)
590 REM *** CALCULATE OFFSET ***
600 TE = TS + E - S:I = ABS (TS - S):T4 = INT ~
    (I / N):T3 = I - T4 * N
605 IF TS < S THEN T3 = - T3: T4 = - T4
610 REM *** MOVE ROUTINE ***
620 IF T3 > 0 THEN K = TE: FOR I = E TO S STEP
     - 1: POKE K, PEEK (I):K=K-1
625 NEXT : GOTO 650
630 K = TS: FOR I = S TO E: POKE K, PEEK (I): ~
    K = K + 1: NEXT
640 REM *** SWAP TS & S, TE & E ***
650 K = TS:TS = S:S = K:K = TE:TE = E:E = K
660 PRINT "LIST UP TO 11 KNOWN DATA TABLES IN ~
    THE  PROGRAM.  TYPE 0,0 WHEN DONE.":P
    RINT
670 PRINT "DATA TABLE "T" START,END:  ";: INPU
    T "";H$,E$
675 IF H THEN GOSUB 240:S(T) = D:H$ = E$: GOSU
    B 240:E(T) = D: GOTO 690
680 S(T) = VAL (H$):E(T) = VAL (E$)
690 IF E(T) THEN I = T3 + T4 * N:S(T) = S(T) +
     I:E(T) = E(T) + I:T = T + 1
695 IF T < 11 GOTO 670
700 IF C < > 2 THEN IF C < > 5 THEN PRINT : GO
    TO 290
710 PRINT :PRINT "LIST UP TO 11 ADDRESSES TO B
    E CHANGED. TYPE 0,0 WHEN DONE.":PRINT
720 PRINT "#"X". OLD,NEW ADDRESSES:  ";: INPUT
     "";H$,E$
725 IF H THEN GOSUB 240:C(X) = D:H$ = E$: GOSU
    B 240:D(X) = D: GOTO 740
730 C(X) = VAL (H$):D(X) = VAL (E$)
740 IF C(X) = D(X) OR X = 10 THEN PRINT : GOTO
     290
750 X = X + 1: GOTO 720
```

Supermon64

Jim Butterfield, Associate Editor

Supermon64 is your gateway to machine language programming on the Commodore 64. Supermon, in several versions, has been popular over the years as a major programming tool for Commodore users. With this adaptation for the 64, a good book on 6502 programming, and patience, you can learn to write programs or subroutines which are capable of running at extraordinary speeds. You can learn machine language programming.

Supermon64 itself is in machine language, but you can type it in without knowing what it means. Using the Tiny Peeker/Poker (Program 1), or via the built-in monitor of a PET, type it in and SAVE it. The fastest way to check for errors is to type in Program 3 on a regular PET (or use the portioning techniques described in the article to make room for the checksum program the same way you made room for the Tiny Peeker). Then load Supermon64 into the PET. It will come in above your BASIC. Then RUN the checksum and it will report the location of any errors.

The easiest way to type in Supermon64 is by using a monitor. Unfortunately, you won't have a monitor until Supermon64 is typed in. This leads to a chicken-and-egg situation. It's no good my suggesting that you should borrow a friend's Supermon64 to type this in; if he's got it, you can just make a copy and save a lot of work. There's gotta be another way.

The Commodore 64 has lots of memory. We can waste some of it to create an easy method which will allow us to enter this program. Maybe we'll even learn something along the way.

You may have noticed that when you power up the 64, it tells you that you have 38911 bytes free. For the moment, I'm not going to tell you what happened to the rest of the 64K (it's there, but BASIC won't use it). The baffling thing is that if you ask the machine to PRINT FRE(0), it will tell you that it has -26627 bytes free. Don't be confused: this is a bug and you can get the right number by adding 65536. Whip out your trusty calculator, do the addition, and do you get 38911? Of course not, but it's close enough (another story for another time).

Now: we're going to trim some memory away from the computer, so as to give ourselves space to stage Supermon64. First, we must prepare the new space so that it will be able to hold a BASIC program. Type POKE 8192,0 and the space is ready. Next, we are going to move BASIC to this new area. Type POKE 44,32 followed by NEW and the deed is done. At this point we seem to have a perfectly normal Commodore 64 machine. Everything will work as before. But, if you ask PRINT FRE(0) you'll find that your free space has dropped to 32765 bytes. We have sectioned off the space where we will plant Supermon64. [*You can use this same technique, then type in Program 3 and run the checksum on your program – Ed.*]

Now we are ready for a simple input program. Enter the following:

Program 1. Tiny Peeker/Poker.

```
100 PRINT "TINY PEEKER/POKER"
110 X$="*":INPUT X$:IF X$="*" THEN END
120 GOSUB 500
130 IF E GOTO 280
140 A=V
150 IF J>LEN(X$) GOTO 300
160 FOR I=0 TO 7
170 P=J:GOSUB 550
180 C(I)=V
190 IF E GOTO 280
200 NEXT I
210 T=0
220 FOR I=0 TO 7
230 POKE A+I,C(I)
240 T=T+C(I)
250 NEXT I
260 PRINT "CHECKSUM=";T
270 GOTO 110
280 PRINT MID$(X$,1,J);"??":GOTO 110
300 T=0
310 FOR I=0 TO 7
320 V=PEEK(A+I)
330 T=T+V
340 V=V/16
350 PRINT " ";
360 FOR J=1 TO 2
370 V%=V
380 V=(V-V%)*16
390 IF V%>9 THEN V%=V%+7
400 PRINT CHR$(V%+48);
410 NEXT J
420 NEXT I
430 PRINT "/";T
440 GOTO 110
500 P=1
```

64 ** ALL NEW!!! ** 64

— — — * * * — — —

SOFTWARE FOR COMMODORE 64

WORD-PAC $74.95
- Print up to 99 pages of text.
- Automatic tabbing/Centering/Underlining.
- Copy Lines/Merge/Plus More!
- Coded in Machine language.

CALC-PAC $74.95
- Interface-Compatible with WORD-PAC & DATA-PAC.
- Coded in our own Unique Spread-Sheet language.
- User-Friendly Mathematical Applications.

DATA-PAC $39.95
- Interface-Compatible with WORD-PAC & CALC-PAC.
- User defined Formats/Search & Sorts.
- Printer compatable.

EDITOR-PAC $69.95
- Complete Programmer's Editor.
- Auto-Number/Renumber including goto & gosub.
- Program Merge/Global Search and Replace.
- Plus Much More!

ASSEMBLER-PAC $59.95
- Programmers take note!
- Mnemonic format to Machine Language.
- Link Modules/External references, More!

HOME-ACCOUNTANT $29.95
- Checkbook with reconciliation routine.
- Hard-Copy listing option.
- Search and Review/Chart of Accounts.
- Income and Expense.

ANNOUNCING . . .

The PCS/8064 Upgrade Module for the 64

On power-up the PCS/8064 provides:
- 80-column video output.
- WORD-PAC word processing.
- CALC-PAC spread sheet mathematics.
- DATA-PAC data base system.
- Exit to BASIC.
- All Applications Interface-Compatible.

Check local dealers or Call Pacific Coast Software for retail pricing.

Plus Full line of Games/Home Software for 64 Free Catalog Offer.

PACIFIC COAST SOFTWARE

3220 S. Brea Canyon Rd.
Diamond Bar, CA 91765
(714) 594-8210

218 S. Main/Box 147
LeSueur, MN 56058
(612) 665-6724

Mid-Eastern Distribution:

PERIPHERALS PLUS .... (215) 687-8540
155 E. Lancaster Ave. - Wayne, Penn. 19087

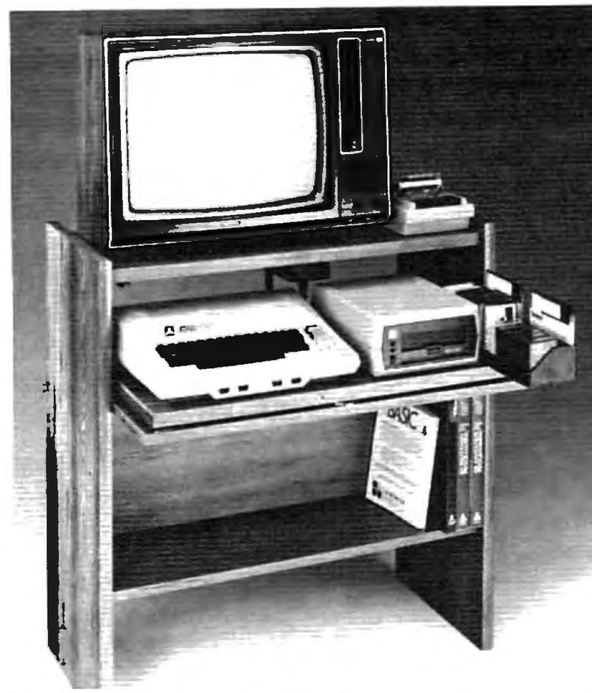
New England Distribution:

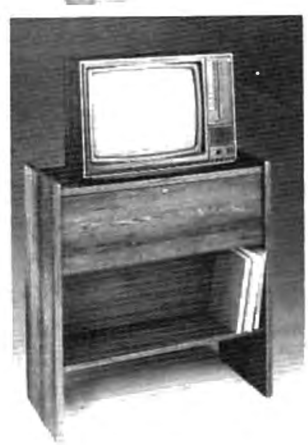
OMICRON ....(617) 769-6867
1416 Providence Highway - Norwood, Mass. 02062

Dealer Inquires Encouraged.

ARE YOU A SMART BUYER?

For **$89.95** this is a smart buy if you're looking for a place to store your computer, peripherals, and accessories without spending a fortune.





The CS 1632 computer storage cabinets compact yet functional design fits almost anywhere while housing your computer monitor, joysticks, software, books and peripherals all for only $89.95.
The slide out shelf puts the computer at the right height and position for easy comfortable operation.
The fold up locking door keeps unwanted fingers off the key board when not in use.
To store joysticks just turn them upside down and slide them into the inverted storage rack.
Twist tabs on the back of center panel allow for neat concealed grouping of wires, while power packs rest hidden behind center panel on shelf.
The slide out software tray has room for 14 cartridges or cassettes and up to 30 diskettes. Most brands of software will fit between the adjustable partitions with a convenient hook for the spare key at rear.
Stand fits Atari 400 & 800, Commodore 64 & VIC 20, Ti 99/4A and TRS-80.
Cabinet dimensions overall 36" high x 33-7/8" wide x 16" deep.
Cabinet comes unassembled. Assembly requires only a screwdriver, hammer, and a few minutes of your time.
Choice in simulated woodgrain, of warm golden oak or rich natural walnut finish.

To order CS1632, send $89.95 to:

BÖEGNER INDUSTRIES CORPORATION

P.O. Box 528
Lake Oswego, OR 97034
Phone orders call, (503) 636-6888

Name ____________________
Address ____________________
City ____________ State ________ Zip ________

☐ Golden oak finish ☐ Natural walnut finish

☐ My personal check, cashiers check or money order is enclosed.
☐ Bill my VISA # ____________ Exp. Date ______
☐ Bill my Mastercard # ____________ Exp. Date ______

Card Holders Signature ____________________

Immediate shipment if in stock. If personal check is sent, allow additional 2 weeks. Prices subject to change. Shipment subject to availability. Cabinet shipped unassembled in 2 cartons. Ships UPS frt. collect FOB Portland, Oregon.

```
510 L=4
520 GOTO 600
550 P=J
560 L=2
600 E=0
610 V=0
620 FOR J=P TO LEN(X$)
630 X=ASC(MID$(X$,J))
640 IF X=32 THEN NEXT J
650 IF J>LEN(X$) GOTO 790
660 P=J
670 FOR J=P TO LEN(X$)
680 X=ASC(MID$(X$,J))
690 IF X<>32 THEN NEXT J
700 IF J-P<>L GOTO 790
710 FOR K=P TO J-1
720 X=ASC(MID$(X$,K))
730 IF X<58 THEN X=X-48
740 IF X>64 THEN X=X-55
750 IF X<0 OR X>15 GOTO 790
760 V=V*16+X
770 NEXT K
780 RETURN
790 E=-1
800 RETURN
```

This program is a very tiny monitor. It will allow you to enter information into memory, eight bytes at a time. To do this: wait for the question mark, and then type in monitor-format the address and contents:

```
? 0800 00 1A 08 64 00 99 22 93
```

The program will return a checksum value to you, which you can use to insure that you have entered the information correctly. To view memory, type in only the address: the contents will be displayed.

Completing The Job

When you have finished entering all that data, you can make Supermon64 happen quite easily. Three last POKE commmands and a CLR:

```
POKE 44,8
POKE 45,232
POKE 46,17
CLR
```

You have Supermon64. Save it with a conventional BASIC SAVE before you do anything else.

Now you may RUN it – and learn how to use it.

Supermon64 Summary

Commodore Monitor Instructions:

G GO RUN
L LOAD FROM TAPE OR DISK
M MEMORY DISPLAY
R REGISTER DISPLAY
S SAVE TO TAPE OR DISK
X EXIT TO BASIC

Supermon64 Additional Instructions:

A SIMPLE ASSEMBLER
D DISASSEMBLER
F FILL MEMORY
H HUNT MEMORY
P PRINTING DISASSEMBLER
T TRANSFER MEMORY

• Simple assembler

```
.A 2000 LDA #$12
.A 2002 STA $8000,X
.A 2005 (RETURN)
```

In the above example the user started assembly at 2000 hex. The first instruction was load a register with immediate 12 hex. In the second line the user did not need to type the A and address. The simple assembler prompts with the next address. To exit the assembler type a return after the address prompt. Syntax is the same as the disassembler output.

• Disassembler

```
.D 2000
(SCREEN CLEARS)
2000 A9 12        LDA #$12
2002 9D 00 80     STA $8000,X
2005 AA           TAX
2006 AA           TAX
```

(Full page of instructions)

Disassembles 22 instructions starting at 2000 hex. The three bytes following the address may be modified. Use the CRSR keys to move to and modify the bytes. Hit return and the bytes in memory will be changed. Supermon64 will then disassemble that page again.

• Printing disassembler

```
.P 2000,2040
2000 A9 12            LDA #$12
2002 9D 00 80         STA $8000,X
2005 AA               TAX

      ....

203F A2 00            LDX #$00
```

To engage printer, set up beforehand:

```
OPEN 4,4:CMD4
```

• Fill memory

```
.F 1000 1100 FF
```

Fills the memory from 1000 hex to 1100 hex with the byte FF hex.

• Go run

```
.G
```

Go to the address in the PC register display and begin RUN code. All the registers will be replaced with the displayed values.

```
.G 1000
```

Program 2. Supermon64.

```
0800 00 1A 04 64 00 99 22 93
0808 12 1D 1D 1D 1D 53 55 50
0810 45 52 20 36 34 2D 4D 4F
0818 4E 00 31 04 6E 00 99 22
0820 11 20 20 20 20 20 20 20
0828 20 20 20 20 20 20 20 20
0830 00 4B 04 78 00 99 22 11
0838 20 2E 2E 4A 49 4D 20 42
0840 55 54 54 45 52 46 49 45
0848 4C 44 00 66 04 82 00 9E
0850 28 C2 28 34 33 29 AA 32
0858 35 36 AC C2 28 34 34 29
0860 AA 31 32 37 29 00 00 00
0868 AA AA AA AA AA AA AA AA
0870 AA AA AA AA AA AA AA AA
0878 AA AA AA AA AA AA AA AA

0880 A5 2D 85 22 A5 2E 85 23
0888 A5 37 85 24 A5 38 85 25
0890 A0 00 A5 22 D0 02 C6 23
0898 C6 22 B1 22 D0 3C A5 22
08A0 D0 02 C6 23 C6 22 B1 22
08A8 F0 21 85 26 A5 22 D0 02
08B0 C6 23 C6 22 B1 22 18 65
08B8 24 AA A5 26 65 25 48 A5
08C0 37 D0 02 C6 38 C6 37 68
08C8 91 37 8A 48 A5 37 D0 02
08D0 C6 38 C6 37 68 91 37 18
08D8 90 B6 C9 4F D0 ED A5 37
08E0 85 33 A5 38 85 34 6C 37
08E8 00 4F 4F 4F 4F AD E6 FF
08F0 00 8D 16 03 AD E7 FF 00
08F8 8D 17 03 A9 80 20 90 FF

0900 00 00 D8 68 8D 3E 02 68
0908 8D 3D 02 68 8D 3C 02 68
0910 8D 3B 02 68 AA 68 A8 38
0918 8A E9 02 8D 3A 02 98 E9
0920 00 00 8D 39 02 BA 8E 3F
0928 02 20 57 FD 00 A2 42 A9
0930 2A 20 57 FA 00 A9 52 D0
0938 34 E6 C1 D0 06 E6 C2 D0
0940 02 E6 26 60 20 CF FF C9
0948 0D D0 F8 68 68 A9 90 20
0950 D2 FF A9 00 00 85 26 A2
0958 0D A9 2E 20 57 FA 00 A9
0960 05 20 D2 FF 20 3E F8 00
0968 C9 2E F0 F9 C9 20 F0 F5
0970 A2 0E DD B7 FF 00 D0 0C
0978 8A 0A AA BD C7 FF 00 48

0980 BD C6 FF 00 48 60 CA 10
0988 EC 4C ED FA 00 A5 C1 8D
0990 3A 02 A5 C2 8D 39 02 60
0998 A9 08 85 1D A0 00 00 20
09A0 54 FD 00 B1 C1 20 48 FA
09A8 00 20 33 F8 00 C6 1D D0
09B0 F1 60 20 88 FA 00 90 0B
09B8 A2 00 00 81 C1 C1 C1 F0
09C0 03 4C ED FA 00 20 33 F8
09C8 00 C6 1D 60 A9 3B 85 C1
09D0 A9 02 85 C2 A9 05 60 98
09D8 48 20 57 FD 00 68 A2 2E
09E0 4C 57 FA 00 A9 90 20 D2
09E8 FF A2 00 00 BD EA FF 00
09F0 20 D2 FF E8 E0 16 D0 F5
09F8 A0 3B 20 C2 F8 00 AD 39

0A00 02 20 48 FA 00 AD 3A 02
0A08 20 48 FA 00 20 B7 F8 00
0A10 20 8D F8 00 F0 5C 20 3E
0A18 F8 00 20 79 FA 00 90 33
0A20 20 69 FA 00 20 3E F8 00
0A28 20 79 FA 00 90 28 20 69
0A30 FA 00 A9 90 20 D2 FF 20
0A38 E1 FF F0 3C A6 26 D0 38
0A40 A5 C3 C5 C1 A5 C4 E5 C2
0A48 90 2E A0 3A 20 C2 F8 00
0A50 20 41 FA 00 20 8B F8 00
0A58 F0 E0 4C ED FA 00 20 79
0A60 FA 00 90 03 20 80 F8 00
0A68 20 B7 F8 00 D0 07 20 79
0A70 FA 00 90 EB A9 08 85 1D
0A78 20 3E F8 00 20 A1 F8 00

0A80 D0 F8 4C 47 F8 00 20 CF
0A88 FF C9 0D F0 0C C9 20 D0
0A90 D1 20 79 FA 00 90 03 20
0A98 80 F8 00 A9 90 20 D2 FF
0AA0 AE 3F 02 9A 78 AD 39 02
0AA8 48 AD 3A 02 48 AD 3B 02
0AB0 48 AD 3C 02 AE 3D 02 AC
0AB8 3E 02 40 A9 90 20 D2 FF
0AC0 AE 3F 02 9A 6C 02 A0 A0
0AC8 01 84 BA 84 B9 88 84 B7
0AD0 84 90 84 93 A9 40 85 BB
0AD8 A9 02 85 BC 20 CF FF C9
0AE0 20 F0 F9 C9 0D F0 38 C9
0AE8 22 D0 14 20 CF FF C9 22
0AF0 F0 10 C9 0D F0 29 91 BB
0AF8 E6 B7 C8 C0 10 D0 EC 4C

0B00 ED FA 00 20 CF FF C9 0D
0B08 F0 16 C9 2C D0 DC 20 88
0B10 FA 00 29 0F F0 E9 C9 03
0B18 F0 E5 85 BA 20 CF FF C9
0B20 0D 60 6C 30 03 6C 32 03
0B28 20 96 F9 00 D0 D4 A9 90
0B30 20 D2 FF A9 00 00 20 EF
0B38 F9 00 A5 90 29 10 D0 C4
0B40 4C 47 F8 00 20 96 F9 00
0B48 C9 2C D0 BA 20 79 FA 00
0B50 20 69 FA 00 20 CF FF C9
0B58 2C D0 AD 20 79 FA 00 A5
0B60 C1 85 AE A5 C2 85 AF 20
0B68 69 FA 00 20 CF FF C9 0D
0B70 D0 98 A9 90 20 D2 FF 20
0B78 F2 F9 00 4C 47 F8 00 A5

0B80 C2 20 48 FA 00 A5 C1 48
0B88 4A 4A 4A 4A 20 60 FA 00
0B90 AA 68 29 0F 20 60 FA 00
0B98 48 8A 20 D2 FF 68 4C D2
0BA0 FF 09 30 C9 3A 90 02 69
0BA8 06 60 A2 02 B5 C0 48 B5
0BB0 C2 95 C0 68 95 C2 CA D0
0BB8 F3 60 20 88 FA 00 90 02
0BC0 85 C2 20 88 FA 00 90 02
0BC8 85 C1 60 A9 00 00 85 2A
0BD0 20 3E F8 00 C9 20 D0 09
0BD8 20 3E F8 00 C9 20 D0 0E
0BE0 18 60 20 AF FA 00 0A 0A
0BE8 0A 0A 85 2A 20 3E F8 00
0BF0 20 AF FA 00 05 2A 38 60
0BF8 C9 3A 90 02 69 08 29 0F

0C00 60 A2 02 2C A2 00 00 B4
0C08 C1 D0 08 B4 C2 D0 02 E6
0C10 26 D6 C2 D6 C1 60 20 3E
0C18 F8 00 C9 20 F0 F9 60 A9
0C20 00 00 8D 00 00 01 20 CC
0C28 FA 00 20 8F FA 00 20 7C
0C30 FA 00 90 09 60 20 3E F8
0C38 00 20 79 FA 00 B0 DE AE
0C40 3F 02 9A A9 90 20 D2 FF
0C48 A9 3F 20 D2 FF 4C 47 F8
0C50 00 20 54 FD 00 CA D0 FA
0C58 60 E6 C3 D0 02 E6 C4 60
0C60 A2 02 B5 C0 48 B5 27 95
0C68 C0 68 95 27 CA D0 F3 60
0C70 A5 C3 A4 C4 38 E9 02 B0
0C78 0E 88 90 0B A5 28 A4 29

0C80 4C 33 FB 00 A5 C3 A4 C4
0C88 38 E5 C1 85 1E 98 E5 C2
0C90 A8 05 1E 60 20 D4 FA 00
0C98 20 69 FA 00 20 E5 FA 00
0CA0 20 0C FB 00 20 E5 FA 00
0CA8 20 2F FB 00 20 69 FA 00
0CB0 90 15 A6 26 D0 64 20 28
0CB8 FB 00 90 5F A1 C1 81 C3
0CC0 20 05 FB 00 20 33 F8 00
0CC8 D0 EB 20 28 FB 00 18 A5
0CD0 1E 65 C3 85 C3 98 65 C4
0CD8 85 C4 20 0C FB 00 A6 26
0CE0 D0 3D A1 C1 81 C3 20 28
0CE8 FB 00 B0 34 20 B8 FA 00
0CF0 20 BB FA 00 4C 7D FB 00
0CF8 20 D4 FA 00 20 69 FA 00

0D00 20 E5 FA 00 20 69 FA 00
0D08 20 3E F8 00 20 88 FA 00
0D10 90 14 85 1D A6 26 D0 11
0D18 20 2F FB 00 90 0C A5 1D
0D20 81 C1 20 33 F8 00 D0 EE
0D28 4C ED FA 00 4C 47 F8 00
0D30 20 D4 FA 00 20 69 FA 00
0D38 20 E5 FA 00 20 69 FA 00
0D40 20 3E F8 00 A2 00 00 20
0D48 3E F8 00 C9 27 D0 14 20
0D50 3E F8 00 9D 10 02 E8 20
0D58 CF FF C9 0D F0 22 E0 20
0D60 D0 F1 F0 1C 8E 00 00 01
0D68 20 8F FA 00 90 C6 9D 10
0D70 02 E8 20 CF FF C9 0D F0
0D78 09 20 88 FA 00 90 B6 E0

0D80 20 D0 EC 86 1C A9 90 20
0D88 D2 FF 20 57 FD 00 A2 00
0D90 00 A0 00 00 B1 C1 DD 10
0D98 02 D0 0C C8 E8 E4 1C D0
0DA0 F3 20 41 FA 00 20 54 FD
0DA8 00 20 33 F8 00 A6 26 D0
0DB0 8D 20 2F FB 00 B0 DD 4C
0DB8 47 F8 00 20 D4 FA 00 85
0DC0 20 A5 C2 85 21 A2 00 00
0DC8 86 28 A9 93 20 D2 FF A9
0DD0 90 20 D2 FF A9 16 85 1D
0DD8 20 6A FC 00 20 CA FC 00
0DE0 85 C1 84 C2 C6 1D D0 F2
0DE8 A9 91 20 D2 FF 4C 47 F8
0DF0 00 A0 2C 20 C2 F8 00 20
0DF8 54 FD 00 20 41 FA 00 20
```

```
0E00 54 FD 00 A2 00 00 A1 C1
0E08 20 D9 FC 00 48 20 1F FD
0E10 00 68 20 35 FD 00 A2 06
0E18 E0 03 D0 12 A4 1F F0 0E
0E20 A5 2A C9 E8 B1 C1 B0 1C
0E28 20 C2 FC 00 88 D0 F2 06
0E30 2A 90 0E BD 2A FF 00 20
0E38 A5 FD 00 BD 30 FF 00 F0
0E40 03 20 A5 FD 00 CA D0 D5
0E48 60 20 CD FC 00 AA E8 D0
0E50 01 C8 98 20 C2 FC 00 8A
0E58 86 1C 20 48 FA 00 A6 1C
0E60 60 A5 1F 38 A4 C2 AA 10
0E68 01 88 65 C1 90 01 C8 60
0E70 A8 4A 90 0B 4A B0 17 C9
0E78 22 F0 13 29 07 09 80 4A

0E80 AA BD D9 FE 00 B0 04 4A
0E88 4A 4A 4A 29 0F D0 04 A0
0E90 80 A9 00 00 AA BD 1D FF
0E98 00 85 2A 29 03 85 1F 98
0EA0 29 8F AA 98 A0 03 E0 8A
0EA8 F0 0B 4A 90 08 4A 4A 09
0EB0 20 88 D0 FA C8 88 D0 F2
0EB8 60 B1 C1 20 C2 FC 00 A2
0EC0 01 20 FE FA 00 C4 1F C8
0EC8 90 F1 A2 03 C0 04 90 F2
0ED0 60 A8 B9 37 FF 00 85 28
0ED8 B9 77 FF 00 85 29 A9 00
0EE0 00 A0 05 06 29 26 28 2A
0EE8 88 D0 F8 69 3F 20 D2 FF
0EF0 CA D0 EC A9 20 2C A9 0D
0EF8 4C D2 FF 20 D4 FA 00 20

0F00 69 FA 00 20 E5 FA 00 20
0F08 69 FA 00 A2 00 00 86 28
0F10 A9 90 20 D2 FF 20 57 FD
0F18 00 20 72 FC 00 20 CA FC
0F20 00 85 C1 84 C2 20 E1 FF
0F28 F0 05 20 2F FB 00 B0 E9
0F30 4C 47 F8 00 20 D4 FA 00
0F38 A9 03 85 1D 20 3E F8 00
0F40 20 A1 F8 00 D0 F8 A5 20
0F48 85 C1 A5 21 85 C2 4C 46
0F50 FC 00 C5 28 F0 03 20 D2
0F58 FF 60 20 D4 FA 00 20 69
0F60 FA 00 8E 11 02 A2 03 20
0F68 CC FA 00 48 CA D0 F9 A2
0F70 03 68 38 E9 3F A0 05 4A
0F78 6E 11 02 6E 10 02 88 D0

0F80 F6 CA D0 ED A2 02 20 CF
0F88 FF C9 0D F0 1E C9 20 F0
0F90 F5 20 D0 FE 00 B0 0F 20
0F98 9C FA 00 A4 C1 84 C2 85
0FA0 C1 A9 30 9D 10 02 E8 9D
0FA8 10 02 E8 D0 DB 86 28 A2
0FB0 00 00 86 26 F0 04 E6 26
0FB8 F0 75 A2 00 00 86 1D A5
0FC0 26 20 D9 FC 00 A6 2A 86
0FC8 29 AA BC 37 FF 00 BD 77
0FD0 FF 00 20 B9 FE 00 D0 E3
0FD8 A2 06 E0 03 D0 19 A4 1F
0FE0 F0 15 A5 2A C9 E8 A9 30
0FE8 B0 21 20 BF FE 00 D0 CC
0FF0 20 C1 FE 00 D0 C7 88 D0
0FF8 EB 06 2A 90 0B BC 30 FF

1000 00 BD 2A FF 00 20 B9 FE
1008 00 D0 B5 CA D0 D1 F0 0A
1010 20 B8 FE 00 D0 AB 20 B8
1018 FE 00 D0 A6 A5 28 C5 1D
1020 D0 A0 20 69 FA 00 A4 1F
1028 F0 28 A5 29 C9 9D D0 1A
1030 20 1C FB 00 90 0A 98 D0
1038 04 A5 1E 10 0A 4C ED FA
1040 00 C8 D0 FA A5 1E 10 F6
1048 A4 1F D0 03 B9 C2 00 00
1050 91 C1 88 D0 F8 A5 26 91
1058 C1 20 CA FC 00 85 C1 84
1060 C2 A9 90 20 D2 FF A0 41
1068 20 C2 F8 00 20 54 FD 00
1070 20 41 FA 00 20 54 FD 00
1078 A9 05 20 D2 FF 4C B0 FD

1080 00 A8 20 BF FE 00 D0 11
1088 98 F0 0E 86 1C A6 1D DD
1090 10 02 08 E8 86 1D A6 1C
1098 28 60 C9 30 90 03 C9 47
10A0 60 38 60 40 02 45 03 D0
10A8 08 40 09 30 22 45 33 D0
10B0 08 40 09 40 02 45 33 D0
10B8 08 40 09 40 02 45 B3 D0
10C0 08 40 09 00 00 22 44 33
10C8 D0 8C 44 00 00 11 22 44
10D0 33 D0 8C 44 9A 10 22 44
10D8 33 D0 08 40 09 10 22 44
10E0 33 D0 08 40 09 62 13 78
10E8 A9 00 00 21 81 82 00 00
10F0 00 00 59 4D 91 92 86 4A
10F8 85 9D 2C 29 2C 23 28 24

1100 59 00 00 58 24 24 00 00
1108 1C 8A 1C 23 5D 8B 1B A1
1110 9D 8A 1D 23 9D 8B 1D A1
1118 00 00 29 19 AE 69 A8 19
1120 23 24 53 1B 23 24 53 19
1128 A1 00 00 1A 5B 5B A5 69
1130 24 24 AE AE A8 AD 29 00
1138 00 7C 00 00 15 9C 6D 9C
1140 A5 69 29 53 84 13 34 11
1148 A5 69 23 A0 D8 62 5A 48
1150 26 62 94 88 54 44 C8 54
1158 68 44 E8 94 00 00 B4 08
1160 84 74 B4 28 6E 74 F4 CC
1168 4A 72 F2 A4 8A 00 00 AA
1170 A2 A2 74 74 74 72 44 68
1178 B2 32 B2 00 00 22 00 00

1180 1A 1A 26 26 72 72 88 C8
1188 C4 CA 26 48 44 44 A2 C8
1190 3A 3B 52 4D 47 58 4C 53
1198 54 46 48 44 50 2C 41 42
11A0 F9 00 35 F9 00 CC F8 00
11A8 F7 F8 00 56 F9 00 89 F9
11B0 00 F4 F9 00 0C FA 00 3E
11B8 FB 00 92 FB 00 C0 FB 00
11C0 38 FC 00 5B FD 00 8A FD
11C8 00 AC FD 00 46 F8 00 FF
11D0 F7 00 ED F7 00 0D 20 20
11D8 20 50 43 20 20 53 52 20
11E0 41 43 20 58 52 20 59 52
11E8 20 53 50 AA AA AA AA AA
```

Go to address 1000 hex and begin running code.

- **Hunt memory**

```
.H CØØØ DØØØ 'READ
```

Hunt through memory from C000 hex to D000 hex for the ASCII string read and print the address where it is found. A maximum of 32 characters may be used.

```
.H CØØØ DØØØ 2Ø D2 FF
```

Hunt memory from C000 hex to D000 hex for the sequence of bytes 20 D2 FF and print the address. A maximum of 32 bytes may be used.

- **Load**

```
.L
```

Load any program from cassette #1.

```
.L "RAM TEST"
```

Load from cassette #1 the program named RAM TEST.

```
.L "RAM TEST",Ø8
```

Load from disk (device 8) the program named RAM TEST. This command leaves BASIC pointers unchanged.

- **Memory display**

```
.M ØØØØ ØØ8Ø
.  ØØØØ ØØ Ø1 Ø2 Ø3 Ø4 Ø5 Ø6 Ø7
.  ØØØ8 Ø8 Ø9 ØA ØB ØC ØD ØE ØF
```

Display memory from 0000 hex to 0080 hex. The bytes following the .: can be altered by typing over them, then typing a return.

- **Register display**

```
.R

     PC   IRQ  SR AC XR YR SP
    ØØØØ E62E Ø1 Ø2 Ø3 Ø4 Ø5
```

Displays the register values saved when Supermon64 was entered. The values may be changed with the edit followed by a return.

- **Save**

```
.S "PROGRAM NAME",Ø1,Ø8ØØ,ØC8Ø
```

SAVE to cassette #1 memory from 0800 hex up to but not including 0C80 hex and name it PROGRAM NAME.

```
.S "Ø:PROGRAM NAME",Ø8,12ØØ,1F5Ø
```

SAVE to disk drive #0 memory from 1200 hex up to but not including 1F50 hex and name it PROGRAM NAME.

- **Transfer memory**

```
.T 1ØØØ 11ØØ 5ØØØ
```

Transfer memory in the range 1000 hex to 1100 hex and start storing it at address 5000 hex.

- **Exit to BASIC**

```
.X
```

Return to BASIC ready mode. The stack value SAVEd when entered will be restored. Care should be taken that this value is the same as when the monitor was entered. A CLR in BASIC will fix any stack problems.

Program 3. Supermon64 Checksum.

```
1ØØ REM SUPERMON64 CHECKSUM PROGRAM
11Ø DATA 1Ø17Ø,13676,154Ø4,14997,15136,
    16221,16696,12816,16228,14554
12Ø DATA14677,15Ø39,14551,151Ø4,15522,
    16414,15914,8958,11945 :S=2Ø48
13Ø FORB=1TO19:READX:FORI=STOS+128:N=P
    EEK(I):Y=Y+N
14Ø NEXTI:IFY<>XTHENPRINT"ERROR IN
    BLOCK #"B:G OTO16Ø
15Ø PRINT"BLOCK #"B" IS CORRECT"
16Ø S=I:Y=Ø:NEXTB:REM CHECK LAST SHORT
    BLOCK BY HAND
```

INSIGHT: Atari

Bill Wilkinson

No gossip to start with this month. Instead, let's start right off into a whole series of interesting tidbits (and even a few tidbytes).

Which Is It? GTIA Or CTIA?

Several articles have been written on how to tell whether you have a GTIA or CTIA in your system. Most of them suggest that you use a GRAPHICS 9 statement and observe the screen (it turns black with a GTIA, remains blue with a CTIA).

But suppose you want to write a program that takes advantage of all the capabilities of the GTIA. What does the poor user with only a CTIA do? If you are commercially clever, you will have your program sense which chip is in use and adapt itself accordingly. This program will enable you to do just that:

```
100 GRAPHICS 0:REM ALWAYS USE THIS MODE
110 PRINT "NOW TESTING FOR CTIA VERSUS GTIA"
120 PRINT "================================"
130 POKE 559,58:POKE 53277,2:REM ENABLE PLAYERS
140 POKE 54279,240:REM USE ROM FOR PLAYER DATA
145 POKE 53248,80:REM CENTERED PLAYER
150 POKE 53278,0:REM CLEAR COLLISION REGISTERS
160 POKE 623,65:REM ENABLE GTIA, IF IT EXISTS
170 POKE 20,0
180 IF PEEK(20)<2 THEN 180
190 POKE 623,1:REM DISABLE GTIA
200 POKE 559,34:POKE 53277,0:REM TURN OFF
PLAYERS
210 FOR A=53261 TO 53265:POKE A,0:NEXT A:REM
(AND PLAYER DATA)
220 IF PEEK(53252) THEN PRINT "SORRY, ONLY A
CTIA":GOTO 240
230 PRINT "AHA! A GTIA."
240 END
```

First of all, to give credit where it is due, I should mention that I was inspired to try this by a remark I read in one of Craig Chamberlain's articles in the *M.A.C.E.* newsletter (Michigan Atari Computer Enthusiasts, in the Detroit area). Portions of that article were also reprinted recently in **COMPUTE!**.

But let's now discuss the program. First, we must explain why and how it works: there is no way to inquire of the chip itself which it is. Even the operating system does not know which is installed. *But* .... (There *had* to be a "but," or there wouldn't be this article.)

There are a few subtle differences between how the two chips view players and missiles. In particular, the GTIA doesn't believe that players can collide with "printed" characters, so it never reports such a collision. The CTIA, though, considers a character to be just another kind of COLORed (and SETCOLORed) display.

The first thing our listing does is insure that we have some mode zero characters on the screen (lines 100 - 120). Then we enable the player DMA and the players themselves (line 130). And we tell the chips that the player data memory is smack in the middle of the ROMs! (Why? To insure that lots of data bits will be on, forcing lots of collisions between the player and the playfield screen characters.)

With line 145, we place the player somewhere left of the center of the screen, insuring that it will collide with our printed message. Then, after clearing the collision registers (to insure that the later results will be valid), we enable the special modes of the GTIA (lines 150 and 160).

We wait for at least one full screen scan (lines 170 and 180), to be sure that the collision will "take" (if it's going to). Then we turn everything back off again (lines 190 - 210).

Finally, we inspect the collision register for player zero. If a collision did occur, it must be because the older CTIA was installed. If no collision occurred, we presume that we have a GTIA.

All of this is a little complicated, but I sincerely hope that some of you game developers out there will start designing some good GTIA-based games, now that you can have them modify themselves for the CTIA owner.

A Few Abbrev'd REMs. Period.

In his article on "The Atari Wedge" (in the November 1982 **COMPUTE!**), Charles Brannon mentions that BASIC treats a line beginning with a period as a REMark, claiming that it is a lucky fluke. Well, it really isn't a fluke. It's just one of those things that got designed into Atari BASIC and then forgotten about.

The rule for using abbreviations in Atari BASIC (and BASIC A+, naturally) is fairly simple: when a statement begins with an abbreviation (any alphabetic characters followed by a period), BASIC searches the keyword name table for the first statement name which matches the abbreviation, *starting at the first character of the abbreviation and ending at the period*.

This means, for example, that "L." will match "LIST" only because LIST is the first word in the keyword name table that begins with an "L". If "LET" had been placed before "LIST" in this table, then "L." would have been interpreted as a LET statement. Boy, aren't we lucky that LIST comes before LET!

Luck had nothing to do with it. The order of those keywords was carefully chosen to provide the maximum usability of the shortest abbreviations. (Actually, I now believe that there are a few variations in the order that might be more useful; but remember that the order was set by intuition, not experience, since the language didn't then actually exist.)

Anyway, Atari had asked for a very short abbreviation for REMark statements (e.g., "!", as is used by most Microsoft BASICs). But what could be shorter than a single period? It's even easier to use than "!" (no shift key needed). How to produce that result? Trivial! Place REM as the first statement name in the keyword table.

So try it sometime. Why type in three characters ("REM") when one will do? Of course, because of the tokenizing nature of Atari BASIC, any abbreviated statement(s) are LISTed in their full form. So "." will be LISTed as "REM".

And a P.S. for those of you into BASIC internals: note that this implies that the token value for REM must be zero, since the token values relate directly to the order of the names in the keyword table.

Page 6 Preached Again

I kind of promised myself that I would get down off my soap box this month and quit ranting and raving. But I couldn't go one whole column without a little preaching, could I?

Stay out of page 6! I can't believe it! It seems that every other article and/or utility program and/or device driver that I run across wants to place itself in page 6 (memory locations $600 to $6FF, 1536 to 1791 decimal). *It won't work*!

How can I possibly install a printer driver in page 6 and then put my player vertical move routine there and my disk block input and output and .... Ah, come on, folks. Give us a break.

If you are writing a complete "system" (a game, or data base program, or whatever), then you are naturally free to configure memory as you wish, including doing whatever you want to page 6. But if you are going to publish a utility in a magazine or include a device driver with your printer interface board or do anything that others might use or modify, *please* don't make it fixed-assembled in page 6. *Please*.

Besides, it is *not* true that BASIC leaves all of page 6 alone. If you do an INPUT from disk (or cassette or anything other than the screen), and if the data you input exceeds 128 bytes, BASIC will use at least a portion of page 6 as its buffer. (However, it is probably – not surely, just probably – safe to use memory from $680 to $6FF.)

A little history: If you examine your Atari BASIC reference, you will find that there are two memory usage tables. One claims that all of page 6 is available for the user. The other claims that only the upper half is available. In general, you should believe the latter. *It is not a design flaw nor an error* that BASIC sometimes uses the bottom half of page 6. It is necessary and documented.

I think it was someone at Atari (my rumor sources say Chris Crawford, but this is unconfirmed) who began using all of page 6 for assembly language routines. And, as I stated above, there is really nothing wrong with doing so within a "closed" environment (where you write *all* the software, both BASIC and assembler). Just don't do it for public consumption.

So what should you do, instead? The best solution is to write self-relocatable code and load it wherever there is free memory (e.g., in a BASIC string). (Showing how to write self-relocatable code might be an instructive article, in and of itself. Any takers?)

The second best solution is to perform my favorite trick: place your code at LOMEM and move LOMEM up. Even here, though, it is best to use relocatable code, so you can run under a variety of operating system configurations and varying heights of LOMEM (as I documented in last month's column).

And, last but not least, I have some good, practical (and a little bit selfish) reasons for avoiding page 6: BASIC A+ uses a good portion of it ($610 through $642, actually). Does that make us a villain? Perhaps a little, to the article writers. But we aren't that terrible: I understand that Microsoft BASIC uses *all* of page 6. And who knows what other languages and operating systems and peripheral devices and whatever will also use page 6? Why complicate both your and others' lives by putting your routines there also?

Some FORTH-Right Comments

I received a very well written and thought-out letter from Steven Weston, of Del Mar, California, regarding the benchmarks I reported in my September 1982 column. Mr. Weston shares the predilections of some others, considering FORTH to have been slighted in that column (and in the following one, I presume).

First, I should like to report that he translated the BASIC benchmark to FORTH and obtained a time of a little under 118 seconds. Which is interesting, since ValFORTH (the version he used) makes use of the Atari floating point routines, I believe. So why should it be slower than Atari BASIC? If I were guessing (which means I'm about

WIN $5,000 Plus Royalties!*

For the best Talking Game

For the Atari® or Apple® II+

Using the VOICE BOX™ Speech Synthesizer

Now you can make your Atari® 400/800 or Apple® II games and other programs come alive with the VOICE BOX by the Alien Group — the first low-cost, smart speech synthesizer with unlimited vocabulary.

Add jokes to your programs. Insults. Compliments. Help messages. Stories. Alien voices. Animal roars. Have your computer talk to the fire department or police in emergencies. To kids. Or blind people. Teach touch typing with immediate spoken feedback. Or just about any other subject — the fun way. Or help a speech-impaired friend communicate . . . the possibilities are limitless.

The VOICE BOX plugs into your Atari's serial port. And talks directly through your TV set. Or into any Apple II slot. No power supply or special interfaces needed.

Just select from its simple screen menu. A dictionary with thousands of common words (on diskette or cassette) automatically translates your text into speech. It's that easy.

But don't let its friendliness fool you. The VOICE BOX has all 64 phonemes (basic sounds, like "ah") built in. So you can precisely create any word or sound you can imagine. And store it all on diskette or tape. Names or foreign language words, for example. Or wierd non-human languages.

Let me entertain you — The VOICE BOX is creative too. It will crack you and your friends up with non-stop random, grammatically correct sentences, using words you specify. It also has an amusing talking face with lip-sync animation — a real crowd-stopper. Best of all, you can call the VOICE BOX from any BASIC program and make your program really hum —literally!

Singing Apples? — Apple owners get all these capabilities too — as a plug-in card plus diskette. Or there's a deluxe version with the dictionary in ROM (no diskettes to bother with), speaker, and ability to "sing" (hey we're not making this up folks) in any key. (Both Apple versions require 32K or more. Applesoft and DOS 3.3).

Don't confuse the VOICE BOX with "dumb" speech synthesizers that can't learn new words. Or software-based ones with lower speech quality — and an annoying tendency to blank out the display when they talk. The VOICE BOX is a true breakthrough in speech synthesis. Small wonder thousands of Atari and Apple owners have already bought the VOICE BOX.

The VOICE BOX is available now at leading computer stores throughout the world. Or direct from the Alien Group, with 10-day money back guarantee if you're not completely satisfied.

VOICE BOX For Atari. $169.00
16K and 32K versions included
(Specify diskette or cassette).

VOICE BOX for Apple II.+ $139.00.
(Requires speaker.)

VOICE BOX for Apple II+. $215.00
(Includes dictionary in ROM and singing capability. Comes with speaker.)

Enclose check or money order.

CHECK YOUR LOCAL COMPUTER STORE FOR TALKING "VOICE BOX" VERSIONS OF YOUR FAVORITE GAMES FROM LEADING GAME COMPANIES

*Win a $5,000 prize — plus royalties — for the best Atari 400/800 or Apple II+ game using the VOICE BOX. Deadline: May 30, 1983. Write for contest details.

Please mail to: The Alien Group, Department PT- 6 , 27 W. 23 St., N.Y., N.Y. 10010
Or call in order to (212) 741-1770

to take a flyer), I would presume that the floating point words for ValFORTH are written in FORTH words, instead of being written as low-level (assembly language) words. The very operation of stacking and unstacking the floating point numbers must then be relatively slow and painstaking.

If this is indeed true, then my comment is a positive one: the FORTH user indeed has the choice of implementing "commands" (words) either way, with other FORTH words or with assembly language. This flexibility is poorly supported by most other languages. (Although many C compiler implementations come close to having such accessible assembly code. C/65 functions, for example, need very little overhead in the assembly language code to "unstack" their parameters.) Want a faster FORTH instead of a smaller one? Recode some routines in assembly language.

What Benchmarks Really Test

Before going on to the second point of Mr. Weston's letter, I should like to note that I feel that perhaps he (and many other readers) missed part of the point of the benchmarks: I was really trying to show how useless any one benchmark is, since it is so easy to dream up benchmarks which show off the best features of a given language. I would be hard pressed to construct even a set of ten benchmarks which would adequately compare languages.

And even if I thought I succeeded, how much is the human interface to a language worth? PILOT is still the easiest language on the Atari to learn and interface to. *By definition*, it therefore outbenchmarks every other language for beginners. But would anyone seriously propose using PILOT for generating prime numbers? I think not. Benchmarks are usually worth the paper they are printed on and no more.

So now to Mr. Weston's second point. I quote: "...the bottom line on languages is to use that language which is best suited to the task. [With Atari BASIC] the lack of integer based math is a serious deficiency which can preclude its use by professional software authors." He goes on to ask why I don't provide a "toolbox" of integer math routines to be interfaced to Atari BASIC "instead of defending an inadequate situation."

Well. Kudos and jibes all in one it seems. Anyway, he is absolutely right: pick the language that fits the job instead of making the job fit the language. You will remember, I hope, that in a recent column I mentioned that I collect languages like some people collect games. I keep hoping to find one that will be useful to me.

But now let me disagree a little on a couple of points. And I do so because I have received too many comments in this same vein. (1) Integer math is *not* needed by all "professional software authors." The person writing a financial package needs integer math about as much as the game writer needs floating point. If you need integer math, choose a language which supports it. (2) BASIC is, unfortunately, a non-extensible language. Sure, we could put integer math routines in memory somewhere and use them from BASIC. But BASIC would still insist on thinking of its variables and constants as floating point, and the conversion time (from floating point to integer to floating point, *ad nauseam*) would wipe out all speed advantages gained. (3) I don't think Atari BASIC is an "inadequate situation." Sure, I think there are other solutions. Why else would our company produce languages such as BASIC A+ and C/65 (and probably more to come)? But "inadequate"? I think not, if it is used for and how it was meant to be used. (If anything is inadequate, it is the 6502 microprocessor, which does not lend itself to the implementation of powerful language compilers.)

But, if you are a beginner, don't let anyone (including me) pressure you into trying to learn a new language before you are ready. It is true that you are not going to write "Super Invading Packers with Tronic Fighters" with Atari BASIC. But just look at what you *can* write! Ten years ago, a computer fanatic would have sacrificed his left thumb for what we now take for granted. Seven short years ago, the "hot" computer game that everybody was rewriting (to make it fit in their expanded memory 8K byte gigantic machine) was Wumpus. Today, I seldom see a published program that doesn't make Wumpus look like something out of the dark ages. Hang in there, folks, you ain't seen nuthin' yet.

The New BASIC Standard?

Well, I finally got time to take a long, hard look at the new ANSI BASIC specification. Whew! I think the tower of Babel must have seemed organized by comparison. Even ADA and PL/1 look like closely designed languages compared to ANSI BASIC. I think that the rule in designing it was "If someone wants it, let's put it in."

You certainly won't see any microcomputer interpreter implementations of it in the near future. I estimate it would take over 80K bytes of Z-80 code to do it (which translates to maybe 100K to 120K of 6502 code). It is definitely designed to be compiled, not interpreted, and then only by big machines.

The error descriptions alone would take a few kilobytes (and they are required!). And what do lines like the following mean?

Use the handy reader service cards in the back of the magazine for information on products advertised in **COMPUTE!**

MAT PRINT #N, SAME, IF THERE EXIT FOR: A$; B$, C
ASK #3: access outin, organization org$, rectype internal, pointer p$

I am disappointed. I had hoped that the committee would distill the best of the various BASICs and come up with a somewhat enhanced version of the original ANSI standard BASIC. Instead, they seem to have distilled out the biggest features of the biggest BASICs they can find. And who will use the standard? Not the micros. (At least not in the near future. I understand that Microsoft's representative on the committee dropped out. From frustration? I would have.) Not those who need to contract with the government. (Soon, you will *have* to use ADA if you work with the defense department and various allied agencies.) Not the big business computer users. (They can't afford to go from COBOL, a clumsy but eminently maintainable language, to a BASIC as kludged up by the committee, with a lack of the data structures that made COBOL successful.)

I guess I believed that the only BASIC users that would be left in a few years would be the hobbyists and the time-sharing companies. Now, I think the only ANSI BASIC users will be the time-sharing companies. Maybe.

As much as I disagree with much of what Microsoft has done, I would rather have seen Microsoft BASIC (version 5, on the CP/M machines) become the standard than the hodgepodge the ANSI committee has selected. ANSI, on a scale of 10, I give you a 2.

The New Atari Computers

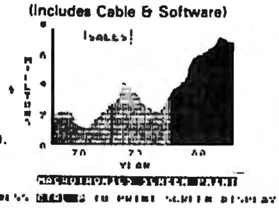
Perhaps by the time you read this, the new Atari computers will be on display at the Consumer Electronics Show (early January, in Las Vegas). Don't expect any real surprises. I expect to hear of a 64K machine (with no software to take advantage of the extra 16K). And probably a low-end 16K machine.

Obviously, Atari needs to get in there and fight with Commodore, both on price and features. Price is easy. Features? Well, if Commodore follows through as they claim they will, it could be a tough fight. And I think the 400 replacement might outstrip the VIC-20. I guess I should note that I am not as much of an Atari loyalist as this paragraph makes me sound. It's just that I like a good, competitive race. The consumer is bound to win.

Oh, yes, one more thing. No more right-hand cartridge slot in the new machines. And no memory board slots at all. Ouch? I don't know. I hope there will be a good way to expand the new machines, but we will all have to wait to see what it is.

Basically BASIC

All this talk about benchmarks and ANSI BASIC has made me regain interest in a project I thought of doing a while back. So, starting next month, we will begin writing a BASIC interpreter right here in this column. And we will write it in BASIC. Interested? I am.



ComputAbility™ presents the Newest in Software Products for Atari

ATARI

SUPER SPECIALS

GORF $29.95 cartridge	KICKBACK $34.95 cartridge	JUMBO JET PILOT $34.95 cartridge
WIZARD OF WOR $29.95 cartridge	SOCCER $34.95 cartridge	SUBMARINE COMMANDER $34.95 cartridge

Prices effective January 1 through January 31, 1983

Buy any 3 programs and receive a *FREE** SLIK STIK

ATARI
CONVERSATIONAL LANGUAGES · T 43.95
INVITATION TO PROGRAMMING 2 & 3 · T ... 21.95
MUSIC COMPOSER · C 32.95
MY FIRST ALPHABET · D ... 28.95
TOUCH TYPING · T 18.95
WORD PROCESSOR · D ... 112.95
BOOKKEEPER · D 112.95
HOME FILING MANAGER · D 37.95
MAILING LIST · T 18.95
ASTEROIDS · C 26.95
CAVERNS OF MARS · D 28.95
COMPUTER CHESS · C 26.95
MISSILE COMMAND · C 26.95
SPACE INVADERS · C 26.95
SUPER BREAKOUT · C 26.95
STAR RAIDERS · C 32.95
ASSEMBLY EDITOR · C 44.95
BASIC · C 44.95
MACROASSEMBLER · D 65.95
MICROSOFT BASIC · D 65.95
PILOT (HOME PACKAGE) · C .58.95

ADVENTURE INTERNATIONAL
REAR GUARD · D 20.95
PREPPIE · D/T 23.95
S.A.G.A. ADVENTURES · D . 23.95

ANALOG
RACE IN SPACE · D/T 20.95
CARNIVAL · D/T 20.95
SUNDAY DRIVER · D/T 23.95
CRASH DIVE! · D/T 23.95

BIG FIVE
COAL MINER 2049'ER · Cart . 39.95

APX
ALL ITEMS - 20% OFF RETAIL. WE CARRY THE ENTIRE APX CATALOG!

AUTOMATED SIMULATIONS
TEMPLE OF APSHAI · D/T ... 31.95
UPPER REACHES OF APSHAI · D/T 16.95
CRUSH, CRUMBLE & CHOMP · D/T 23.95
RICOCHET · D/T 16.95
CRYPT OF THE UNDEAD · D 23.95
ARMOR ASSAULT · D 31.95
DRAGON'S EYE · D 23.95
ESCAPE FROM VULCAN'S ISLE · D 23.95
KING ARTHUR'S HEIR · D ... 23.95
MONSTER MAZE · Cart 31.95
PLATTERMANIA · Cart 31.95
THE NIGHTMARE · D 23.95

DATASOFT
SHOOTING ARCADE · D/T . 23.95
PACIFIC COAST HIGHWAY · D/T 23.95
CLOWNS & BALLOONS · D/T 23.95
MICROPAINTER · D 27.95
CANYON CLIMBER · D/T ... 23.95
FATHOMS FORTY · D 27.95
O'RILEY'S MINE · D/T 27.95
ROSEN'S BRIGADE · D/T ... 27.95
SANDS OF EGYPT · D 31.95
SPELL WIZARD · D 63.95

BRODERBUND
APPLE PANIC · D/T 23.95
STELLAR SHUTTLE · D/T ... 23.95
DAVID'S MIDNIGHT MAGIC · D 27.95
STAR BLAZER · D 25.50
TRACK ATTACK · D 23.95
LABYRINTH · D/T 23.95
SERPENTINE · D 23.95
DUELING DIGITS · D 23.95
DEADLY SECRETS · D 27.95
CHOPLIFTER · D 27.95
GENETIC DRIFT · D/T 23.95
SEAFOX · D 23.95

COSMI
AZTEC CHALLENGER · D ... 16.95
AZTEC CHALLENGER · T ... 11.95
CRYPTS OF PLUMBOUS · D 13.95
CRYPTS OF PLUMBOUS · T .. 8.95
GALACTIC AVENGER · D ... 13.95
GALACTIC AVENGER · T 8.95
SPIDER INVASION · D 16.95
SPIDER INVASION · T 11.95

FIRST STAR
ASTRO CHASE · D/T 23.95

ON-LINE
JAWBREAKER · D/T 23.95
ULTIMA I · D 31.95
CROSSFIRE · D/T 23.95
THRESHOLD · D 31.95
ULTIMA II · D 44.95
MOUSKATTACK · D 27.95
FROGGER · D/T 27.95

***COUPON PROGRAM**
The purchase of each program (with the exception of Super Specials, Atari and APX) will earn you 1 COMPUTABILITY DIVIDEND COUPON. Save 3 coupons and redeem them for your choice of Slik Stik, a Lefty Adaptor, or an Extension Cable. You pay only a $2.50 shipping & handling charge.

I.D.S.I.
POOL 1.5 · D 27.95
POOL 400 · Cart 31.95
SPEEDWAY BLAST · Cart ... 31.95

ROKLAN
DELUXE INVADERS · Cart .. 31.95
GORF · Cart 35.95
WIZARD OF WOR · Cart 35.95

SENTIENT
CYBORG · D 27.95
GOLD RUSH · D/T 27.95

SIRIUS
SNAKE BYTE · D 23.95
SPACE EGGS · D 23.95
CYCLOD · D 23.95
SNEAKERS · D 23.95
BANDITS · D 27.95
WAY OUT · D 31.95

SYNAPSE
PROTECTOR · D/T 23.95
SLIME · D/T 23.95
SHAMUS · D/T 23.95
FORT APOCALYPSE · D/T .. 23.95
CLAIM JUMPER · D/T 27.95
NAUTILUS · D/T 23.95
PICNIC PARANOIA · D/T 27.95
RAPTILIAN · D/T 27.95

D - Disk T - Cassette C - Cartridge

WE CARRY HUNDREDS OF ITEMS FOR ATARI 400/800, ASK FOR OUR FREE CATALOG.

ATARI is a trademark of ATARI, Inc.

Starfighter
The Ultimate Joystick
- 2 Year Warranty
- More Accurate
- Easier to Hold

$16.95

SLIK STIK
- 90 Day Warranty
- Easy Ball Top Control.

Suncom

$9.95

LEFTY JOYSTICK ADAPTOR
Adapts to *any* Atari Joystick.
Moves fire button to top Right. $9.95

EXTENSION CABLE (5 ft.)
Adapts to *any* Atari controller. $6.95

ALL JOYSTICKS WORK WITH Atari VCS, Sears Telegame, Commodore VIC 20, Atari 400 800 (All products have registered trademarks)

30 Day Money Back Guarantee on all Suncom Products

PERCOM D/D DISK DRIVE $579.00

MOSAIC 32K RAM BOARD $97.95

Mastercard/VISA Order Toll Free 800-558-0003

In Wisc. Call 414/351-2007

ORDERING INFORMATION
To order by mail send money order, certified check or personal check (allow 14 days to clear) to COMPUTABILITY. Include $2.00 shipping on software orders and $2.50 shipping on hardware orders (FREE OR PURCHASED). Mastercard & VISA please include card number and expiration date. WI residents please add 5% sales tax. Outside of continental U.S.A. please add 15% shipping (U.S. Funds only). Prices subject to change without notice.

Order Hours
Monday · Friday
12 pm · 9 pm C.S.T.
ComputAbility
P.O. Box 17882
Milwaukee, WI 53217

Michael Day

Computers And Communication

Like the brains of animals, computers have evolved from dedicated, single-mindedness into general-purpose information processors. This month's column takes us through a brief survey of the changes in the capabilities of micro-, mini-, and mainframe computers, concluding with how telecommunications are generally used with each type of computer.

When computers were first introduced, they were designed and *wired* to perform a particular job. When the job was done, either it was performed again with new information inserted, or the computer was *rebuilt* to run a different job. A patch board reprogrammed the computer by reconfiguring the way the hardware was put together. Such reprogramming was necessary because there was just not enough memory available to hold both the program and data at the same time. (1K of memory was a lot back then.)

Batch Processing

Later, as the available memory size increased (to a whole 4K), software programmable machines were built. This improvement allowed the machine to be automatically "built" by the same mechanism that was used to load the data into the machine. Since most of the time spent on the computer was in "building" it for the particular job at hand, this improvement also permitted an interesting approach to processing, called *batch processing*.

Batch processing involves loading the program and data into the machine from a mass storage device (usually a tape drive) and running the program. The results are then saved (printed, put back on the tape or on punch cards). The program and data are then purged from the system, and a new program/data job is loaded into the machine. Batch processing helped increase the popularity of these very expensive machines. But they required intensive use to make them worth the cost.

Using The Computer's Time

As computer costs increased, even batch processing was insufficient to offset the costs of the computer. Analysis of computer operations showed that much of the computer's time was spent waiting for information to be given to it. If the computer could be subdivided into individually operating parts (or *subprocessors*), it would be possible to request the information from a slow external device, such as a tape drive, and while the information was being retrieved, another job could be loaded into the computer and operated on. The processor could later return to the original job and finish it.

Eliminating the computer's inactive or waiting times greatly contributed to the efficient utilization of the computer. By sharing the computer's resources, several different jobs could run at the same time. Careful control of access to the various parts of the system could actually make the computer work like several different computers at the same time. Several users could therefore use the computer without interfering with or being aware of each other. This brought into being *time-share* computers. Since a user seldom uses the system continuously, someone else could use it when it would otherwise be idle.

Patterns Of Development

When the minicomputer (bigger than a "micro," smaller than a "mainframe") came into being, it went through the same sequence of development. It started out as a computer designed to solve a particular problem and developed into a general purpose machine. The difference: by the time the minicomputer was developed, it was cheaper to design it to be program-controlled rather than to have fixed control. This was true because many parts of the machine could be shared by many parts of the program. Because it was not necessary to have individual parts available for each action the computer performed, the computer could actually "rebuild" itself on the fly.

The result was a shift from the mainframe concept of computing. Since large mainframe computers operate best where there are large chunks of data to be processed, they tend to be run mostly as batch processing machines where an entire job, or a large portion of it, is operated on before moving on to something else. The minicomputer, however, is more suitable to applications where the job requirements are varied and rapidly shifting. They are thus most often found in time-share applications where the ability to handle a large number of jobs simultaneously is more important than the actual processing time. The minicomputer can't meet the raw crunch power of the mainframe, but it surpasses the mainframe in adaptability.

A Rapid Change In Microcomputers

When the microcomputer came along, again the same development pattern was followed. Like the mainframes and minicomputers, the microcomputer was initially developed for single-job applications. But it moved on to more generalized applications more rapidly than either of the other computers. Since the microcomputer was developed as a result of Large Scale Integration (LSI chips), the computers could be created at a very low development cost and an unbelievably low production cost.

The microcomputer too does not have the crunch power of the mainframe, nor does it have the adaptability of the mini. What it does have is low cost of implementation, which makes it the first computer ideally suited to fixed job applications. Some of these applications are found in the calculator, smart thermostats, microwave oven controllers, etc.

In between these fixed applications and the minicomputer are the high level microcomputers (which are coming to be called personal computers). These computers, though sometimes not suitable to the rapidly changing job environment of the mini, do have general processing capabilities. This makes them ideal for personal computing since only a single job generally needs to be run at one time, but the types of jobs that the computer is required to perform are varied.

Telecommunications Needs

You might be wondering, "That's all very fine, but what has this got to do with telecommunications?"

Actually, there is a very definite relationship between the type of computer and its needs in telecommunications. Large mainframes seldom need extensive telecommunications. When they do have such a need, it generally involves special communication circuits designed specifically for the computer system, such as airline or hotel reservation systems, or banking systems. Minicomputers, because they are highly adaptive, tend to use a wide variety of communications capabilities. Examples are the many time-share systems and service bureaus.

Microcomputers, as opposed to *personal* computers, generally don't have a need for telecommunications. When they do, the telecommunications tend to be specific to the device or application. In fact, in some applications, the microcomputer is the communications device, as it is with some of the high-powered modems available.

Finally, with the personal computer, communications vary depending on the use to which the computer is put. Generally, the application consists of machine to machine communications between users or connection to a large data base service like Micronet or The Source.

These are only generalizations, of course, and it is quite easy to find exceptions to the rule. You can find microcomputers handling multiple communications devices, and fully dedicated minicomputers that have no outside communications at all. As a general rule, however, these basic patterns prevail.

Jim Butterfield, Associate Editor

Speed Demon

Machine language programs are fast. So fast, in fact, that for many applications we can consider them to be instantaneous. That's good, of course, but sometimes we have to take steps to restrain the program's speed.

The first moon lander program that I wrote was carefully coded, and the calculations were carefully checked to see that they were correct. I was rather taken aback to discover that the instant I pressed GO I found myself crashed on the lunar surface. All the calculations had taken place correctly, but everything worked too fast.

Waiting For The World

In most cases, your machine language program is controlled by the speed of external events. If you're waiting for a user to type a line at the keyboard, chances are that machine language is running at about a ten thousandth of its potential speed. It can do nothing until the next key is pressed; and chances are that it will do little until a line has been completed by pressing the RETURN (or ENTER) key.

Even when we're not waiting for the operator, we are usually waiting for some external process. Using the printer? Your program will spend most of its time waiting for the printer to be ready for the next character. Disk? Same thing. Communications lines fall into the same category, but there's a difference: even though the transfer rate of characters to and from the communications interface is relatively slow, there will often be a need to check it very frequently.

The result is that your program speed is usually determined by the speed of external events. In this case, the "instantaneous" assumption is quite legitimate.

Let's take another example: you're printing material on the screen. Now you can deliver characters at blinding speed; but there's a limit to how fast a user can read. Better slow it down, or your program will be useless.

Compute Bound

Programs that spend most of their time waiting for external events are called *I/O-bound*. Sounds like a good name for a sailing ship, but it really means that if you could get a faster printer, disk, or whatever, your program would run significantly faster. Your speed is bound to the speed of these devices.

There are other programs that do a great deal of computation: they tend to be compute-bound. No, that doesn't mean that you plan to submit them to your favorite magazine; it means that if you could calculate faster, you'd get more work done. Compute-bound programs are often mathematical in nature: to calculate the millionth prime number you won't care much about your printer speed; you want the computation to be fast. Sorting programs are often compute-bound: there's a lot of calculation needed there.

It's often wise to think about your program in terms of its potential: will it be I/O-bound or compute-bound? It will give you an idea of where you might place extra effort in order to speed things up.

Slowing Down

There are many cases where we deliberately wish to slow down the speed of a machine language program. Animation is a prime example: you don't want your space ships, bombs, or cats to always travel at supersonic speed. Indeed, if you used maximum speed you'd never see them.

There are technical reasons to want to slow

COMPUTE! Back Issues

Here are some of the applications, tutorials, and games from available back issues of **COMPUTE!**. Each issue contains much, much more than there's space here to list, but here are some highlights:

February 1981: Simulating PRINT USING, Using the Atari as a Terminal for Telecommunications, Attach a Printer to the Atari, Double Density Graphing on C1P, Commodore Disk Systems, PET Crash Prevention, A 25¢ Apple II Clock.

May 1981: Named GOSUB/GOTO in Applesoft, Generating Lower Case Text on Apple II, Copy Atari Screens to the Printer, Disk Directory Printer for Atari, Realtime Clock on Atari, PET BASIC Delete Utility, PET Calculated Bar Graphs, Running 40 Column Programs on a CBM 8032.

June 1981: Computer Using Educators (CUE) on Software Pricing, Apple II Hires Character Generator, Ever- expanding Apple Power, Color Burst for Atari, Mixing Atari Graphics Modes 0 and 8, Relocating PET BASIC Programs, An Assembler In BASIC for PET, QuadraPET: Multitasking?

July 1981: Home Heating and Cooling, Animating Integer BASIC Lores Graphics, The Apple Hires Shape Writer, Adding a Voice Track to Atari Programs, Machine Language Atari Joystick Driver, Four Screen Utilities for the PET, Saving Machine Language Programs on PET Tape Headers, Commodore ROM Systems, The Voracious Butterfly on OSI.

August 1981: Minimize Code and Maximize Speed, Apple Disk Motor Control, A Cassette Tape Monitor for the Apple, Easy Reading of the Atari Joystick, Blockade Game for the Atari, Atari Sound Utility, The CBM "Fat 40," Keyword for PET, CBM/ PET Loading, Chaining, and Overlaying.

October 1981: Automatic DATA Statements for CBM and Atari. VIC News, Undeletable Lines on Apple, PET, VIC, Budgeting on the Apple, Switching Cleanly from Text to Graphics on Apple, Atari Cassette Boot-tapes, Atari Variable Name Utility, Atari Program Library, Train your PET to Run VIC Programs, Interface a BSR Remote Control System to PET, A General Purpose BCD to Binary Routine, Converting to Fat-40 PET.

December 1981: Saving Fuel $$ (Multiple Computers: versions for Apple, PET, and Atari), Unscramble Game (multiple computers), Maze Generator (multiple computers), Animating Applesoft Graphics, A Simple Printer Interface for the Apple II, A Simple Atari Wordprocessor, Adding High Speed Vertical Positioning to Atari P/ M Graphics, OSI Supercursor, A Look At SuperPET, Supermon for PET/CBM, PET Mine Maze Game.

January 1982: Invest (multiple computers), Developing a Business Algorithm (multiple computers), Apple Addresses, Lowercase with Unmodified Apple, Cryptogram Game for Atari, Superfont: Design Special Character Sets on Atari, PET Repairs for the Amateur, Micromon for PET, Self-modifying Programs in PET BASIC, Tinymon: a VIC Monitor, Vic Color Tips, VIC Memory Map, ZAP: A VIC Game.

February 1982: Insurance Inventory (multiple computers), Musical Transposition (multiple computers), Multitasking Emulator (multiple computers), Disassemble Apple Programs from BASIC, Plotting Polar Graphs on Apple, Atari P/M Graphics Made Easy, Atari PILOT, Put A Rainbow in your Atari, Marquee for PET, PET Disk Disassembler, VIC Paddles and Keyboard, VIC Timekeeping.

March 1982: Word Hunt Game (multiple computers), Infinite Precision Multiply (multiple computers), Atari Concentration Game, VIC Starfight Game, CBM BASIC 4.0 To Upgrade Conversion Kit, Apple Addresses, VIC Maps, EPROM Reliability, Atari Ghost Programming, Atari Machine Language Sort, Random Music Composition on PET, Comment Your Apple II Catalog.

April 1982: Track Down Those Memory Bugs (multiple computers), Shooting Stars Game (multiple computers), Intelligent Input Subroutines (multiple computers), Ultracube for Atari, Customizing Apple's Copy Program, Using PET/CBM In The High School Physics Lab, Grading Exams on a Microcomputer (multiple computers), Atari Mailing List, Renumber VIC Programs The Easy Way, Browsing the VIC Chip, Disk Checkout for PET/CBM.

May 1982: VIC Meteor Maze Game, Atari Disk Drive Speed Check, Modifying Apple's Floating Point BASIC, Fast Sort For PET/ CBM, Extra Atari Colors Through Artifacting, Life Insurance Estimator (multiple computers), PET Screen Input, Getting The Most Out Of VIC's 5000 Bytes.

June 1982: Outpost Game (multiple computers), Apple Pascal Lister, Income Property (multiple computers), VIC Intelligent Videodisc System, Atari Disk Operating Systems, PET/Apple Search, A Self-modifying Atari P/M Utility, Use Atari Joysticks with VIC, VIC/PET Program Transfers.

July 1982: Gold Miner Game (Atari and VIC), IRA Planner (multiple computers), Atari Video Graphics, Apple DOS Changer, Super QuadraPET, VIC Overview, Maze Race (multiple computers), Direct Access File Editor (PET and Atari), VIC Super Expander Memory Map, Using The 6560 Video Interface Chip, PET Compactor, Headless FORTH Metacompilation, Test RAM Nondestructively (multiple computers).

August 1982: The New Wave Of Personal Computers, Household Budget Manager (multiple computers), Word Games (multiple computers), Color Computer Home Energy Monitor, Intelligent Apple Filing Cabinet, Guess That Animal (multiple computers), PET/CBM Inner BASIC, VIC Communications, Keyprint Compendium, Animation With Atari, VIC Curiosities, Atari Substring Search, PET and VIC Electric Eraser.

September 1982: Apple and Atari and the Sounds of *TRON*, Commodore Automatic Disk Boot, VIC Joysticks, Three Atari GTIA Articles, Color Computer Graphics, The Apple Pilot Language, Sprites and Sound on the Commodore 64, Peripheral Vision Exerciser (multiple computers), Banish INPUT Statements (multiple computers), Charades (multiple computers), PET Pointer Sort, VIC Pause, Mapping Machine Language, Editing Atari BASIC With the Assembler Cartridge, Process Any Apple Disk File.

Home and Educational COMPUTING! (Fall 1981 and Summer 1981 – count as one back issue): Exploring The Rainbow Machine, VIC As Super Calculator, Custom Characters, Alternate Screens, Automatic Line Numbers, Using The Joystick (Spacewar Game), Fast Tape Locater, Window, VIC Memory Map.

Back issues are $3 each or six for $15. Price includes freight in the US. Outside the US add $1 per magazine ordered for surface postage. $4 per magazine for air mail postage. All back issues subject to availability.

In the Continental US call
TOLL FREE 800-334-0868
(In NC Call 919-275-9809)

Or write to **COMPUTE!** Back Issues, P.O. Box 5406, Greensboro, NC 27403 USA. Prepayment required in US funds. MasterCard, Visa and American Express accepted. North Carolina Residents add 4% sales tax.

down certain activities. Some types of interfaces want you to hold a voltage at a certain level for a minimum amount of time before you take it away again. You may need to "stall" for a few instructions (or a few dozen) to make sure that you're doing the job right.

The Stall Loop

The simplest way to slow things down is to kill time in a stall loop. If you're not doing anything else anyway, this is quite sensible and easy to do. We might code:

```
        LDX #$00
LOOP    DEX
        BNE LOOP
```

At a typical clock rate of 1 Mhz (a million cycles per second) the above routine will waste a little over a millisecond of time. You could make the time shorter by changing the LDX value at the start. For longer delays, you use a loop within a loop:

```
        LDY #$00
        LDX #$00
LOOP    DEX
        BNE LOOP
        DEY
        BNE LOOP
```

This will waste almost a third of a second as written above; change the LDY to reduce the delay.

The Timer

If the time is moderately long and you have other things to do, you may set the desired time into a timer and check it occasionally to see how the time has been going. Timers are part of the interface chips – the 6522 VIA has two of them, for example. They work a little like kitchen timers: you put the desired time in and it runs downward toward zero, showing you the time remaining. Time runs very quickly in these, however: the maximum time is often something like a fifteenth of a second. Don't try to time a boiled egg unless you either call the timer many times or you like it really soft-boiled.

When you have more than one event to time, it's nevertheless often best to stay with just one timer. Juggling various timers can be more work than just setting the next expected event into a single one. When you have numerous different things going on, you can often still work by a single timer, as we'll explain.

Countdowns

It's often convenient to have a single timer, and clock all events on a "countdown" basis. The timer can run at fixed intervals – on the PET, you can often use the interrupt timing of 1/60 second to clock many events.

The trick is this: whenever your timer signals, count one for each event you have going. You can count up or down; but when you have counted a fixed value, it's time to handle that particular process.

An example: you have a game involving tanks, planes, bombs and bullets (the usual destructive thing). On a sixtieth-of-a-second timer, you might move a tank every 20 time units; a plane, every 10 time units; a bomb, every eight; and a bullet every five. You don't need a dozen different timers: every time the bullet counter reaches five, you move it to the next spot of the screen and see what you've shot down.

Machine language is fast, and often seems instantaneous. It's often so much faster than other processes in the computer that we don't need to worry about speed calculations at all.

Sometimes machine language is too fast. When that happens, there are ways of slowing it down.

It's hard to believe that you can be so speed rich that you have to rein back your program, but it can happen. ©

PROGRAMMING THE TI

C. Regena

We are happy to welcome C. Regena and her new, monthly TI-99/4A column to the pages of **COMPUTE!**. *She has extensive experience in personal and educational computing and has written numerous articles on TI computers. To start things off, here is an overview of hardware, software, and miscellaneous resources for the TI.*

Welcome to the world of the TI-99/4A computer! (This column is also addressed to TI-99/4 owners and users, but since the "A" is the newer and more plentiful model, I'll refer to both computers when I write "TI-99/4A.") For home, personal, and educational applications, the TI-99/4A computer is a very powerful machine. In this column I'd like to illustrate some of the features unique to this microcomputer.

Extraordinary Graphics And Sound

Graphics. You may easily define your own high-resolution (detailed) graphics characters. There are 16 colors, and you may use all 16 on the screen *at the same time* in high-resolution graphics (unlike other computers). You may also use text anywhere on the screen at the same time you use high resolution graphics. Most other microcomputers are limited when combining text with graphics.

Music. You may play up to three notes and one noise for a specified time using *one* statement. The music is specified by a number which represents a frequency of 110 Hz to 44733 Hz, tones from low A on the bass clef up to out of human hearing range. The tone may be between regular musical notes. An example which plays a three-note, C-major chord for three seconds is:

```
CALL SOUND(3000,262,6,330,4,440,2)
```

The first number is the duration in milliseconds, in this case 3000. The next numbers are frequency and loudness for each note. You may also add a "frequency" of -1 through -8 and a loudness for the noise generator. You may combine tones and noises for all kinds of sounds – everything from classical music to sound effects from outer space.

Combining music and graphics. "Computer choreography" is possible because other statements (including graphics) may be executed while music is played. You may illustrate a song, for example. Or if you have a game program, you may make calculations while you are making a noise. The computer will play music and execute statements until the duration runs out or until the program comes to another CALL SOUND statement with a positive duration. A negative number for the duration will start that CALL SOUND statement even if the first duration has not finished. Try using a FOR/NEXT loop to vary any of the parameters for special effects. Here is a sample using just one tone:

```
100 FOR N=500 TO 880 STEP 20
110 CALL SOUND(-99,N,2)
120 NEXT N
130 FOR N=880 TO 500 STEP -20
140 CALL SOUND(-99,N,2)
150 NEXT N
160 GOTO 100
```

Noises. Using negative durations and combinations of music and noise numbers for frequency, you can make all sorts of synthesized noises. Quite often with noises you will want to use a FOR/NEXT loop and vary the loudness parameter.

Built-in BASIC. The programming language of TI BASIC is built into the main console – nothing extra to buy. The TI BASIC language is an excellent language for learning how to program, yet it is powerful enough for an experienced mathematician because of the built-in functions.

String manipulations. String (non-number) manipulations are also very powerful. Here is a sample program to print a phrase A$ on the screen starting at row R and column C:

```
100 FOR I=1 TO LEN(A$)
110 CALL HCHAR(R,C+I-1,ASC(SEG$(A$,I,1)))
120 NEXT I
```

The loop will go from 1 to the LENgth of the phrase A$. String variable names must always end with a dollar sign. SEG$ takes a SEGment of the phrase. In this case we are starting at the left side and taking one letter at a time. ASC gets the ASCII character code value of the character in the phrase. CALL HCHAR uses a graphic method to place the character on the screen at a certain row and column.

No Variable Name Worries

Variable naming. In your own programming on the TI-99/4A you may use meaningful variable names, although in many microcomputers the BASIC language recognizes only two letters – or a letter and a number – for a variable name. For example, if you have a program with the variable name BLUE and another variable name BLACK, other computers may recognize only one variable, BL, but the TI-99/4A knows you are using two variables. You also do *not* have to worry about embedded reserved words in variable names.

Documentation. Two excellent manuals are included with the computer. One teaches you programming in TI BASIC. The manual is very easy to understand, and a person with no previous computer experience can learn to program with this book. Also included is the *User's Reference Manual*, which may cost over $15 for other computers. The reference manual, which is in loose-leaf form, includes all the commands along with explanations and sample programs.

Plug-in modules. The easiest way to use the TI-99/4A is to insert a command module which contains a program. Modules are available for a variety of applications. The variation in price is largely dependent on the amount of memory built into the module. The modules actually add memory to the computer while they are being used.

Speech. Even though this feature is not built in, I am going to include speech in this list of unique features of the TI-99/4A because it is very easy to use and because, if you purchase six command modules before January 31, you can get the TI Speech Synthesizer free. The speech synthesizer is a small box that attaches to the side of your console. Command modules are available for you to program your own speech.

16-bit microprocessor. The TI-99/4A uses a 9900, 16-bit microprocessor, which offers more computing power and greater expansion and configuration flexibility than an eight-bit microprocessor. You can get higher numeric precision and simplified memory addressing.

Programmer's aids. Programmers will enjoy the easy line editing features. Various function keys allow you to insert or delete characters or to erase or clear a line. There is also a TRACE command to help in debugging.

Another feature programmers like is the built-in automatic numbering. Just type in NUM, press ENTER, and you can start programming. The line numbers start with 100 and automatically increment by 10. You may specify any starting number and increment. NUM 5,2 will start with line 5 then increment by 2.

After you have programmed and added or deleted statements here and there, you'll enjoy the automatic resequencing command, RES, which will automatically renumber your statements, including all statement numbers referenced by other statements.

There is a lot built into the TI-99/4A, and I have only touched on some features this month. Future columns will go into more detail, and I hope to be able to answer your questions and present programs and ideas to help you really enjoy your computer.

Since many readers may be new TI-99/4A owners and users, let's also describe some *peripherals* – hardware you can add on to your basic console. You may have noticed that buying a computer is much like buying a house – you can buy the basic house (computer), but then you need to add furniture (programs or software) to make it livable (usable), and soon you want to make major improvements (add peripherals).

Using The Cassette Recorder

Cassette. Probably one of the first items you'll need is a cassette cable to connect a cassette recorder to the computer to save your own programs or to use cassette programs available on a variety of subjects and applications. Nearly any cassette recorder is acceptable; however, the TI-99/4A is more critical on how you set the volume control than is the TI-99/4. In general, a battery-operated recorder does not work well enough for accurate data retrieval. Also, your recorder should have a tone control and a volume control. I have had the greatest success using the Panasonic RQ2309A cassette recorder.

Page I-9 in the *User's Reference Guide* tells how to connect the cassette cable, and the pages following describe how to save and load data from modules. Page II-42 shows an example of how to load a program that you have saved or purchased. Some other hints for using the cassette recorder are:

Turn the tone control to the highest setting.

Start with the volume about mid-range.

Follow the instructions after you type in OLD CS1.

If you get the message NO DATA FOUND, increase the volume.

If you get the message ERROR IN DATA, decrease the volume.

Sometimes a fraction of a change in volume can make all the difference in your success in reading a program. Once in a while, if I alternate between the two error messages at a volume setting near 2 or 3, I turn the volume to about 8 or 9 and the program will load.

The smallest jack of the cassette cable goes into the remote switch of the cassette recorder so the computer can turn the recorder on and off automatically. If the recorder does not turn on and off properly, simply remove the remote jack from the plug. You can operate the cassette recorder manually to save and load programs. For programs using the cassette recorder for data entry, you will need the remote capability. An adapter is available for the remote switch.

Disk drives. You can save and retrieve data or programs on a diskette much more quickly than by using a cassette system. The TI-99/4A uses 5¼-inch, single-sided, soft-sectored diskettes. To connect a disk drive, you also need a disk controller. One disk controller can handle up to three disk drives. Many business applications require two disk drives.

Memory Expansion. The TI Memory Expansion is for 32K RAM, and you need a module that will access it. You cannot use it with console BASIC. Extended BASIC does not require the memory expansion but can use it. Pascal, TI Logo, and Editor/Assembler require the memory expansion.

Peripheral Box. The "old" method had each peripheral in a separate "box" connected to the computer or the previous peripheral; each had its own power cord. The "new" system is the peripheral box, which has its own power supply and slots for cards for the RS-232 interface, memory expansion, disk controller, P-code, one disk drive, and possible future cards.

Monitor. Although the TI-99/4A may be connected to your regular television set, Texas Instruments has a very attractive, ten-inch, color monitor. The monitor gives a very clear, sharp picture and may be connected to other microcomputers as well as the TI-99/4A.

Making The Computer Speak

Speech. The TI Speech Synthesizer allows you to hear the computer speak to you. You will need a command module with built-in speech to hear the computer speak.

To program your own speech or to use any cassette or disk programs that use speech, you will need a module. Speech Editor and Extended BASIC have speech capabilities with a given list of words. Terminal Emulator II allows unlimited speech; the accompanying documentation gives you ideas for programming speech using this module. You may vary the pitch and slope and inflections. You may use allophones to create words, or you may have the computer speak words which you spell phonetically.

Telecommunications And Languages

Terminal. The Terminal Emulator II command module (or Terminal Emulator I, which does not have speech) allows you to use your TI-99/4A to act as a terminal either to another computer or to a large telecommunications service. You will also need the TI RS-232 Interface and a telephone modem.

The SOURCE is an on-line information service from Source Telecomputing Corporation. TEXNET is a special edition of The SOURCE especially for the Texas Instruments home computer.

Printer. You may use a number of different brands of printers with your microcomputer. To connect your TI-99/4A to a printer, you'll need the TI RS-232 Interface and a cable to go from the interface to the printer (the cable is usually sold with the printer).

RS-232. The RS-232 Interface has two ports so you may be connected to a modem and a printer at the same time. An instruction book comes with the RS-232 so you'll know how to operate the computer under different conditions.

Extended BASIC. TI Extended BASIC (XBASIC) is a programming language contained on a module. A manual (over 200 pages) and a programmer's reference card come with the module. No other peripherals are necessary to use XBASIC. If a program has been written in XBASIC, the XBASIC module must be inserted for the program to run. Some of the advantages of XBASIC are multi-statement lines, complex IF-THEN-ELSE logic, subroutine and MERGE capabilities, DISPLAY AT and PRINT USING, program security (SAVE protection), speech (with speech synthesizer), and moving sprites with greater graphics capabilities.

Logo. TI Logo is a fascinating programming language designed especially for young children. TI Logo is contained in a module, and the 32K memory expansion is required. Logo I can print using the TI thermal printer only. Logo II has music and also RS-232 capability so you can print listings on a regular printer.

Editor/Assembler. For machine language programmers, it requires the memory expansion, disk controller, and one disk drive.

USCD PASCAL. This language requires the memory expansion, P-code peripheral card, disk controller, and at least one disk drive.

Software

Software. I've mentioned software (programs) last, although it's probably the first extra purchase you will make for your computer. Software is what you need to use your computer. Software is available on command modules, cassettes, and diskettes, and in a variety of subjects. Scott, Foresman educational courseware is available for grade levels kindergarten through eighth grade, Texas Instruments has several educational modules, and other educational and publishing companies are also developing modules for all grade levels.

In addition, there are modules for all types of home use (budget, finances, decision making, record keeping) and, of course, games from chess to soccer, from *Hunt the Wumpus* to *TI Invaders*.

Cassette and diskette programs are available for many applications, including programs for two-year-olds learning colors to sophisticated business programs. When you purchase, a software's documentation should tell you what hardware is required. For example, much of the business software requires a printer and two disk drives (and thus the peripheral box, RS-232 Interface, and disk controller) plus perhaps the Extended BASIC module and/or the 32K memory expansion.

Current literature. Texas Instruments sends an informative newsletter to all owners (be sure to send in your registration card). Many user groups have formed which have their own newsletters and catalogs. Other magazines are available that support the TI-99/4A. Now **COMPUTE!** will offer you a monthly column dedicated to the TI-99/4A, and other articles and programs to help you enjoy your TI-99/4A home computer to the fullest.

Copy VIC Disk Files

Roger L. Smith, Mesa, AZ

The "Copy 2031 Files" program by G. H. Watson (**COMPUTE!**, August 1982) was greatly appreciated. It was suggested that the program might run on the VIC-20 if appropriate changes were made. The following program makes those changes.

The program will work with either VIC-1540 or the new VIC-1541 disk drives. However, your VIC *must* have a 3K memory expansion plugged in since the program uses memory beginning at the expansion start-of-BASIC address ($0400) to handle the file transfer.

As noted in the original article, the program will handle BASIC program files. To copy sequential files (data files, ASM/TED files), replace the appended P with an S by using POKE 7672,83. Program files use POKE 7672,80. If you want to transfer more than one file, you will have to type SYS 7354 each time.

```
500 FOR ADRES=7354TO7674:READ DATTA
    :POKE ADRES,DATTA:NEXT ADR
    ES
600 SYS7354
634 DATA 169,1,133,184,32,195
640 DATA 255,169,8,133,186,169
646 DATA 15,133,185,169,0,133
652 DATA 183,32,192,255,160,29
658 DATA 169,179,32,30,203,32
664 DATA 96,197,169,0,133,187
670 DATA 169,2,133,188,160,255
676 DATA 200,177,187,208,251,162
682 DATA 4,189,245,29,145,187
688 DATA 200,202,208,247,132,183
694 DATA 169,2,133,184,32,195
700 DATA 255,169,8,133,186,169
706 DATA 2,133,185,32,192,255
712 DATA 32,141,29,162,2,32
718 DATA 199,242,169,4,133,1
724 DATA 169,3,133,0,160,0
730 DATA 32,14,242,145,0,166
736 DATA 144,208,7,200,208,244
742 DATA 230,1,208,240,132,5
748 DATA 165,1,133,6,169,2
754 DATA 32,195,255,32,204,255
760 DATA 160,29,169,191,32,30
766 DATA 203,32,228,255,240,251
772 DATA 164,183,136,169,87,145
778 DATA 187,32,192,255,32,141
784 DATA 29,162,2,32,9,243
790 DATA 169,4,133,1,169,3
796 DATA 133,0,160,0,177,0
802 DATA 32,122,242,165,1,197
808 DATA 6,208,4,196,5,240
814 DATA 14,200,208,238,230,1
820 DATA 208,234,160,29,169,228
826 DATA 32,30,203,169,2,32
832 DATA 195,255,169,1,32,195
838 DATA 255,32,204,255,76,116
844 DATA 196,162,1,32,199,242
850 DATA 32,14,242,141,240,29
856 DATA 32,14,242,141,241,29
862 DATA 32,204,255,173,240,29
868 DATA 201,48,208,206,173,241
874 DATA 29,201,48,208,199,32
880 DATA 204,255,96,13,70,73
886 DATA 76,69,78,65,77,69
892 DATA 63,32,0,13,83,87
898 DATA 73,84,67,72,32,68
904 DATA 73,83,75,69,84,84
910 DATA 69,83,44,13,84,72
916 DATA 69,78,32,72,73,84
922 DATA 32,82,69,84,85,82
928 DATA 78,46,13,0,13,68
934 DATA 73,83,75,32,69,82
940 DATA 82,79,82,32,0,0
946 DATA 33,13,0,0,82,44
952 DATA 80,44,246,230,1,76
```

Keith Falkner

Tap Applesoft's Heartbeat

You can use machine language routines to enable Applesoft to read and rapidly process incoming data.

Imagine that your Apple is connected to some gizmo which feeds the Apple some data rapidly. The device could be, for example, a modem or some newfangled digital geiger counter monitoring an atomic reactor. In an example below, we will simulate this device with the game paddle buttons, or, if you have no paddles, with a mere piece of wire. The essential idea is that the attached device offers data to the Apple sporadically, and the data will be lost if it is not noticed and processed within a few milliseconds.

If you try to support this device with a program written in Applesoft BASIC, you will likely miss some of the data offered by the device, because Applesoft is rather slow. Assuming that such a problem does arise and must be solved, here's how.

Machine Language Patch Into CHRGET

Here is an intriguing exercise: type in and run the listing in Program 1. If you type it correctly, it will say "OK"; make sure you fix it if it says "OOPS." This program installs, but does not run, three tiny machine language routines. Now type CALL 909 and then run the program again. Inexplicably, it will make an irritating buzz for the 0.37 seconds it takes to run. Indeed, you can load and run almost any Applesoft program and listen to it run.

You may notice that difficult computations and lengthy array references are accompanied by buzzes, whereas fast-running code such as FOR/NEXT loops that do little more than count will produce brief musical tones. I do not suggest that this is a useful effect, but I hope it sparks your interest, for what is coming is a bit dull and difficult but results in a very powerful technique which you can harness to produce utterly amazing results at zero cost.

By the way, you can deactivate the noise-making routine and restore your Apple to normal by typing CALL 896. The DOS command FP is even more powerful; issue that if your Apple seems confused.

A Look Into CHRGET

Here is how the noise is caused. The Applesoft interpreter uses a tiny routine to fetch each byte of your program in turn as the program runs. The (valid) BASIC statement IF BAD THEN STOP is stored as six bytes, specifically the token for IF, the letters B, A, and D, and the tokens for THEN and STOP. The character-getting routine, which is known by the name CHRGET, will be invoked a total of seven times to execute all of this statement (the token for THEN is fetched twice, once to detect the end of the variable name BAD, and once to be executed).

Program 1 and the routine installed at location 909 introduce a detour into CHRGET so that the Apple's speaker is tweaked each time a character of the program is fetched. This of course makes the noise and accounts for the various buzzes and squeaks made by slow- and fast-running code. To see the actual machine language routine, enter the monitor via CALL -151 and enter 380L (number 380 followed by letter L) to see the routines at 896 ($0380), 909 ($038D), and 922 ($039A).

The CHRGET routine starts in location 177 ($00B1), and can be listed by B1L (letter B, digit 1, letter L). You can verify if you wish that CALL 909 installs a JMP instruction at location 186 ($00BA), and CALL 896 restores the CMP and BCS instructions which belong there. You can return from the monitor to Applesoft by typing CTRL-C and pressing RETURN.

Now let's put this technique to use. If you have game paddles, identify PDL (1) and skip the rest of this paragraph. To simulate the button on PDL(1), you will need a piece of slender wire at least two feet long. Solid wire works better than multi-strand. Strip about one-eighth inch from each end. You shuld now *turn off* the Apple and open the cover carefully. Locate the GAME I/O connector at coordinates J8 on the motherboard, and stick an end of the wire into hole number three, which is third from the front on the right side.

Do be careful with this, because disaster awaits you if you pick the wrong hole, or are careless with the other end of the wire. Now close the cover of the Apple, letting the free end of the wire hang down away from the computer. Reach under the front edge of the keyboard and you will find the heads of some bolts. You will be touching the free end of that wire to one of these to simulate a press of the button. If you choose, you can loosen one of these, attach another piece of wire, tighten the bolt, and attach the two loose ends of wire to any type of switch, but this is not essential. When these preparations are complete, turn the Apple on again.

Catching Every Count

Now type in Program 2 and run it. Please note the lengthy loop in lines 130-140. This takes over half a minute to run and obviously contains none of the PEEK statements necessary to test for a press of button number one. Those tests are done by the machine language routine patched into CHRGET, at locations 922 through 965. When the program is running, press the button (or touch the wire to the bolt) as fast as you can count, and you will find that the Apple catches every single one. Actually, when you try to touch the wire to the bolt once, you almost certainly cause it to bounce and touch the bolt more than once, so the count will be higher than you expect, and never lower.

In this example the switch was tested by a few instructions in machine language. This powerful technique is possible only in machine language. Perhaps it is possible to devise a routine that would permit a few lines of BASIC to be invoked by the routine which interrupts CHRGET, but what would be the point? Our objective here is to support a rapid-fire device, and any attempt to do this in BASIC will, it is assumed, lead to missed data. At least that is where this article started.

Using The Keyboard Buffer

A totally practical application of intercepting CHRGET is a keyboard buffer, except for one troublesome detail. From time to time, in any program which handles strings, Applesoft must pause to accomplish "garbage collection" – in other words, to make available again some memory which has been used for storage of strings which were later discarded. This process usually takes from one to thirty seconds, but in an artificial and extreme case it could take over an hour!

During "garbage collect," Applesoft is totally out of touch with all external events, so the keyboard buffering routine has no way to service the keyboard. Nonetheless, the routine is of genuine help when a speedy typist is using a slow data entry program. In fact, even a moderately slow hunt-and-peck typist like me can occasionally leave Applesoft behind. With the buffer running, I never lose a key.

There are two other limitations. During processing of the LIST command, Applesoft is not using CHRGET, so the buffering routine has no chance at the keyboard. Also, when DOS is active, all BASIC functions are inactive, so again the keyboard cannot be serviced.

Program 3 shows the complete keyboard buffer program. The program occupies the first 512 points of the BASIC program area, so it destroys any Applesoft program already present.

Briefly, here is how the program works. A preliminary test verifies that Applesoft is active, for this program is inapplicable to Integer BASIC. Next, the program sees if the beginning-of-BASIC pointer has been altered to $0A01 (from the usual $0801). If so, a warm start is done, retaining the current Applesoft program; if not, the pointer is so altered, and the new routine of Applesoft is called. Then the "patch" to CHRGET is made, as in Programs 1 and 2.

The next step is a connection to the keyboard-servicing routine at the "hook" known as KSW. Whenever such a connection is what you need, you must let DOS know your intentions, or it will patiently remove your connection and restore its own hook. This is very easy – just CALL 1002 (or JSR $3EA in machine language). The program ends by entering Applesoft at the warm-start entry $E003.

By this point, the program really has not done anything except insinuate itself into the system and protect itself from harm. The actual buffer is the 256-byte area from $0900 to $09FF (2304 to 2559), and two one-byte counters look after data in the buffer. The counter BIX points to the next place where a key can be stored, and the counter BOX points to the next byte to be sent to whoever asks for a key.

For example, if BIX contains $2E and BOX contains $28, the operator has keyed six bytes ahead, and they are stored in locations $0928 through $092D. If the operator now keys exactly 250 more bytes before the running program asks for any more, the keyed bytes will be stored in $092E through $09FF, then the buffer will "wrap around" and more keys will be stored in $0900 through $0926. By this time the value in BIX will be $27, one less than that in BOX. That's 249 in addition to the six already there, and now the buffer is full, so the buffering routine will sound the "bell" when it cannot store the last byte keyed.

At this point the operator must pause and wait for the program to catch up. I think this event is very unlikely.

Keys are detected and stored by the routine patched into CHRGET. A word of caution to anyone patching CHRGET: since BASIC uses this routine dozens or thousands of times a second, the patch must execute as fast as possible, else the program may be slowed to an unacceptable degree.

Does It Function?

When a key is wanted, the code at INLINK sees if one is in the buffer. If not, the standard ROM routine is called. If a key is available in the buffer, it is delivered, and the counter, BOX, is updated to account for the departed key. It is all very simple, mainly because of the eight-bit indexing automatically provided by the 6502's X-register. Indeed, if the buffer were any size but 256 bytes, the program would have been noticeably harder to write and debug.

OK, how do you key this program into your Apple? You could CALL -151 to get to the monitor, then type in all the hex stuff, 803:4C 09 08 4C 99 08, and so on. If you did the "homework" I assigned in last month's column, there is an easier way. Key in the pure Applesoft program in Program 4, then SAVE it, RUN it, and finally EXEC GEN KEYBUF. This final step will invoke the mini-assembler to build KEYBUF, save the result, and return control to the keyboard eventually. This process must destroy any Applesoft program in memory, so be sure you have saved Program 4 before typing the EXEC command!

To verify all this work, peer closely at the screen – the command JMP $083C should be in location 08A8. The acid test, of course, is "does it work?" Follow the instructions below to test your work, and when you actually make it work, you'll have a potent and versatile tool which makes your Apple a little bit better than it was before!

Homework Assignment. Boot your System Master and LOAD BRIAN'S THEME. That is the program which displays pretty moiré patterns in high resolution. Here is some code to add a fascinating effect! Type in the few lines in Program 5 and RUN the changed program. When the display starts acting oddly, play with the keyboard. The most recently pressed key controls the timing in a tiny machine language routine at location 600 ($258).

In my particular Apple, the keys W, K, 8, question mark, and especially CTRL-D, produce interesting effects. The machine language routine is completely relocatable, so it can be used without change in any place in memory where 26 bytes are free. So if you wish to use the routine in another program, change the variable ML to whatever suits you. The timing is so delicate that the effects change greatly when ML is just under a multiple of 256, so that a branch instruction crosses a page boundary. To stop this demonstration, you must press RESET, because the machine language routine treats CTRL-C as any other key.

Table: How to use the Keyboard Buffer

1. To load and initialize the routine,
   BRUN KEYBUF
2. Now use your Apple as usual, but be sure that you do not switch to Integer BASIC!
3. To suspend use of the buffer,
   CALL 2054
4. To resume use of the buffer,
   CALL 2051
5. To recover memory used by the buffer, after suspending it via CALL 2054,
   FP (or INT, if you choose)
6. To copy the routine from disk to disk,
   BLOAD KEYBUF
   Insert the disk to receive a copy.
   BSAVE KEYBUF,A$803,L$F8

Program 1.

```
10 REM   'TAP' DEMO 1
20 FOR I = 896 TO 935
30 READ X
40 Z = Z + X
50 POKE I,X
60 NEXT
70 IF Z <  > 5155 GOTO 90
80 PRINT "OK": END
90 PRINT "OOPS. Z=";Z: END
896 DATA 169,201,133,186,169,58
902 DATA 133,187,169,176,133,188
908 DATA 96, 169,76,133,186,169
914 DATA 154,133,187,169,3,133
920 DATA 188,96,141,48,192,201
926 DATA 58,176,3,76,190,0
932 DATA 76,200,0,0
```

Program 2.

```
10 REM   'TAP' DEMO 2
20 REM
30 FOR I = 896 TO 955
40 READ X
50 Z = Z + X
60 POKE I,X
70 NEXT
80 IF Z < > 7425 THEN PRINT "OOPS. Z=";Z: STOP
90 HOME : GR
100 PRINT "WHILE I SCRIBBLE AIMLESSLY,"
110 PRINT "PRESS BUTTON 1 SEVERAL TIMES."
120 POKE 24,0: POKE 25,0: POKE 26,0: CALL 909
130 FOR I = 1 TO 1000: COLOR= 16 * RND (I)
140 PLOT 40 * RND (I),40 * RND (I): NEXT
150 CALL 896:T =  PEEK (25) + 256 * PEEK (26)
160 TEXT : PRINT CHR$ (7): REM BELL!
170 HOME : PRINT "YOU PRESSED IT "; INT (T / 2
    );" TIMES."
```

```
896 DATA 169,201,133,186,169,58
902 DATA 133,187,169,176,133,188
908 DATA 96, 169,76,133,186,169
914 DATA 154,133,187,169,3,133
920 DATA 188,96,72,152,72,173
926 DATA 98,192,41,128,197,24
932 DATA 240,8,133,24,230,25
938 DATA 208,2,230,26,104,168
944 DATA 104,201,58,176,3,76
950 DATA 190,0,76,200,0,0
```

Program 3.

```
0000          ; THIS PROGRAM USES 512 BYTES FROM
0000          ; 2048 TO 2559 TO CONTAIN AND LOOK
0000          ; AFTER A 256-BYTE KEYBOARD BUFFER.
0000          ;
0000          ; 'BRUN KEYBUF' TO CREATE THE BUFFER.
0000          ; 'CALL 2054' TO DISABLE THE BUFFER.
0000          ; 'CALL 2051' TO RE-ENABLE THE BUFFER.
0000          ;
0000          ; HOW TO SAVE THE PROGRAM:
0000          ; BSAVE KEYBUF,A$803,L$F8
0000          ;
0000                 *=$803          ;START AT 2051.
0803          ;
0803          ; JUMP-TABLE OF ENTRY-POINTS:
0803          ;
0803 4C0908          JMP STARTS      ;ENABLE BUFFER
0806 4C9908          JMP CANCEL      ;DISABLE BUFFER
0809          ;
0809 AD00E0 STARTS LDA $E000          ;WHICH LANGUAGE?
080C C94C          CMP #$4C        ;APPLESOFT?
080E D036          BNE STEXIT      ;NO, SO QUIT!
0810          ;
0810 A90A          LDA #>BASIC     ;-> NEW START
0812 A001          LDY #1          ;OF BASIC (+1)
0814          ;
0814 C467          CPY $67         ;WARM ENTRY TO ME?
0816 D004          BNE STCOLD      ;NO
0818 C568          CMP $68         ;WARM FOR SURE?
081A F00C          BEQ STLINK      ;YES!
081C          ;
081C 8467   STCOLD STY $67             ;SET UP THE NEW
081E 8568          STA $68         ;START-OF-BASIC
0820 A900          LDA #0
0822 8D000A        STA BASIC       ;TRADITION
0825 204BD6        JSR $D64B       ;EXECUTE 'NEW'.
0828          ;
0828 A94C   STLINK LDA #$4C
082A 85BA          STA $BA
082C A949          LDA #<CHLINK    ;TIE IN TO
082E 85BB          STA $BB         ;CHRGET.
0830 A908          LDA #>CHLINK
0832 85BC          STA $BC
0834          ;
0834 A97C          LDA #<INLINK
0836 8538          STA $38         ;TIE IN TO THE
0838 A908          LDA #>INLINK    ;INPUT HOOK 'KSW'
083A 8539          STA $39
083C          ;
083C ADEA03 STTIES LDA $3EA
083F C94C          CMP #$4C        ;IS DOS PRESENT?
0841 D003          BNE STEXIT      ;NO, NO DISK HERE!
0843 20EA03        JSR $3EA        ;TELL DOS ABOUT TIE-IN
0846 4C03E0 STEXIT JMP $E003           ;WARM START
0849          ; THIS ROUTINE IS ENTERED EVERY TIME
0849          ; APPLESOFT FETCHES A BYTE OF BASIC.
0849          ;
0849 2C00C0 CHLINK BIT $C000           ;KEY PRESSED?
084C 1026          BPL CHCOLO      ;NO, NOT YET
084E 48            PHA             ;SAVE BASIC BYTE
084F 8A            TXA             ;SAVE X-REGISTER
0850 48            PHA
0851 AEAC08        LDX BIX         ;GET INPUT POINTER
0854 E8            INX             ;PREPARE TO STEP UP
0855 ECAD08        CPX BOX         ;BUT IS BUFFER FULL?
0858 D00A          BNE CHSTOW      ;NO, GO & STASH
085A 98            TYA             ;BUFFER FULL:
085B 48            PHA             ;(BELL USES Y-REG)
085C 20E2FB        JSR $FBE2       ;RING THE BELL!
085F 68            PLA
0860 A8            TAY
0861 4C6E08        JMP CHRETR
0864          ;
0864 8EAC08 CHSTOW STX BIX             ;SAVE NEW POINTER
0867 CA            DEX             ;-> PLACE FOR THE KEY
0868 AD00C0        LDA $C000       ;GET THE KEY
086B 9D0009        STA BUF,X       ;SAVE IN BUFFER
086E          ;
086E 8D10C0 CHRETR STA $C010           ;RESET KEYBOARD
0871 68            PLA
0872 AA            TAX             ;RECOVER X-REG
0873 68            PLA             ;& BYTE OF BASIC
0874          ;
0874 C93A   CHCOLO CMP #$3A            ;(CHRGET REPLACEMENT)
0876 B003          BCS CHBACK      ;"
0878 4CBE00        JMP $BE         ;"
087B 60     CHBACK RTS                 ;"
087C          ;
087C          ; THIS ROUTINE IS USED WHENEVER A
087C          ; KEY IS NEEDED FROM THE KEYBOARD.
087C          ;
087C 8EAB08 INLINK STX SAVX            ;SAVE IT
087F AEAD08        LDX BOX         ;GET OUTPUT POINTER
0882 ECAC08        CPX BIX         ;ANYTHING IN BUFFER?
0885 D006          BNE INSEND      ;YES, GO SEND IT!
0887 AEAB08        LDX SAVX        ;NO, RESTORE X-REG
088A 4C1BFD        JMP $FD1B       ;NORMAL KEY HANDLER
088D          ;
088D 9128   INSEND STA ($28),Y         ;STOP FLASHING
088F BD0009        LDA BUF,X       ;GET KEY FROM BUFFER
0892 EEAD08        INC BOX         ;UPDATE POINTER
0895 AEAB08        LDX SAVX        ;RESTORE X-REG
0898 60            RTS
0899          ; DISABLE THE KEYBOARD BUFFER
0899          ;
0899 A9C9   CANCEL LDA #$C9
089B 85BA          STA $BA
089D A93A          LDA #$3A        ;RESTORE CHRGET
089F 85BB          STA $BB         ;ORIGINAL STUFF
08A1 A9B0          LDA #$B0
08A3 85BC          STA $BC
08A5          ;
08A5 2089FE        JSR $FE89       ;EXECUTE "IN#0".
08A8 4C3C08        JMP STTIES
08AB          ;
08AB 00     SAVX   .BYT 0          ;SAVE AREA FOR X-REG
08AC 00     BIX    .BYT 0          ;-> PLACE FOR NEXT BYTE
08AD 00     BOX    .BYT 0          ;-> NEXT ONE TO DELIVER
08AE          ;
08AE          ; (BIX=BOX) MEANS BUF IS EMPTY
08AE          ; (BIX+1=BOX) MEANS IT'S FULL!
08AE          ;
08AE          ; THE ABOVE MUST END BY $8FF
08AE          ; OR IT WILL BE OVERWRITTEN!
08AE          ;
08AE        BUF=$900               ;BUFFER IS $900-$9FF
08AE        BASIC=$A00             ;NEW START-OF-BASIC
08AE          ;
08AE               .END
```

SYMBOL TABLE

SYMBOL	VALUE				
BASIC	0A00	BIX	08AC	BOX	08AD
BUF	0900	CANCEL	0899	CHBACK	087B
CHCOLO	0874	CHLINK	0849	CHRETR	086E
CHSTOW	0864	INLINK	087C	INSEND	088D
SAVX	08AB	STARTS	0809	STCOLD	081C
STEXIT	0846	STLINK	0828	STTIES	083C

Program 4.

```
100 REM MAKE "GEN KEYBUF"
110 D$ = CHR$ (4)
120 F$ = "GEN KEYBUF"
130 PRINT D$"OPEN "F$
140 PRINT D$"WRITE"F$
150 PRINT "FP"
160 PRINT "MON I"
170 PRINT "BRUN MINI-ASSM"
180 PRINT "803:";: REM NOTICE SEMICOLON
190 READ Z$
200 IF Z$ = "END" GOTO 230
210 PRINT " "Z$
220 GOTO 190
230 PRINT "FP"
240 PRINT "BSAVE KEYBUF,A$803,L$F8"
250 PRINT D$"CLOSE"
260 END
270 DATA JMP809,JMP899,LDAE000
280 DATA CMP#4C,BNE846,LDA#A
290 DATA LDY#1,CPY67,BNE81C
300 DATA CMP68,BEQ828,STY67
310 DATA STA68,LDA#0,STAA00
320 DATA JSRD64B,LDA#4C,STABA
330 DATA LDA#49,STABB,LDA#8
340 DATA STABC,LDA#7C,STA38
350 DATA LDA#8,STA39,LDA3EA
360 DATA CMP#4C,BNE846,JSR3EA
370 DATA JMPE003,BITC000,BPL874
380 DATA PHA,TXA,PHA
390 DATA LDX8AC,INX,CPX8AD
```

```
400 DATA BNE864,TYA,PHA
410 DATA JSRFBE2,PLA,TAY
420 DATA JMP86E,STX8AC,DEX
430 DATA LDAC000,"STA900,X",STAC010
440 DATA PLA,TAX,PLA
450 DATA CMP#3A,BCS87B,JMPBE
460 DATA RTS,STX8AB,LDX8AD
470 DATA CPX8AC,BNE88D,LDX8AB
480 DATA JMPFD1B,"STA(28),Y","LDA900,X"
490 DATA INC8AD,LDX8AB,RTS
500 DATA LDA#C9,STABA,LDA#3A
510 DATA STABB,LDA#B0,STABC
520 DATA JSRFE89,JMP83C,BRK
530 DATA BRK,BRK,END
```

If you have Integer BASIC in ROM or in a Language Card, substitute:

```
150 PRINT "INT"
170 PRINT "CALL -2667" :REM MINI-
    ASSM
```

Program 5.

```
460 ML = 600 : FOR I = ML TO ML+25
470 READ X: POKE I,X: NEXT
480 LIST  (... OR PRINT SOME STUFF)
490 CALL ML
500 DATA 173,80,192,173,0,192
510 DATA 41,127,170,202,208,253
520 DATA 173,81,192,173,0,192
530 DATA 41,127,170,202,208,253
540 DATA 240,230
```

©

Atari Lister

LeRoy J Baxter, Milwaukie, OR

Debugging a long program listing can be tedious. Most of us have typed in a long program and then had to hunt for errors when it wouldn't RUN. This utility routine can make the job a little easier.

Make a copy of this program and LIST it to tape or disk. When you need it, load it with the ENTER command (the line numbers shouldn't conflict). Then type GOTO 32700. A prompt will appear. Press RETURN, and the first set of six program lines will be LISTed to the screen, regardless of their line numbers, with spaces between the lines. Then with a touch of RETURN, it LISTs the next set of six lines.

Enter "EDIT," and the program goes to the Editing Subroutine. It asks for the line number of the offending line, then LISTs the line and prints the command "CONT" below it. You can then edit the line using the screen editor keys.

When you press RETURN, the line will be entered into the program. You can enter or delete complete lines using standard techniques. Simply move the cursor up and enter your new line between the LISTed line and CONT. When you are done, enter "ERASE," and the utility program will erase itself.

```
32700 DIM A$(5):T=0
32705 Z=0:INPUT A$:ON (A$="EDIT")+(A$
      ="ERASE")*2 GOTO 32730,32745
32710 ? CHR$(125):ADDR=PEEK(136)+PEEK
      (137)*256:FOR X=0 TO T:ADDR=ADD
      R+PEEK(ADDR+2)*(T>0):NEXT X
32715 LINENO=PEEK(ADDR)+PEEK(ADDR+1)*
      256:Z=Z+1:IF LINENO>=32700 THEN
       ? "* END OF LISTING *":GOTO 32
      710
32720 LIST LINENO:T=T+1:ADDR=ADDR+PEE
      K(ADDR+2):IF Z<6 THEN 32715
32725 GOTO 32710
32730 ? "WHAT LINE #";:INPUT X
32735 ? CHR$(125):POSITION 2,4:LIST X
      :? :? :? :? "CONT":INPUT A$:POS
      ITION 2,0:POKE 842,13:STOP
32740 POKE 842,12:T=T-6:GOTO 32705
32745 ? CHR$(125):POSITION 2,4:FOR X=
      32700 TO 32750 STEP 5:? X:NEXT
      X:? "POKE 842,12"
32750 POSITION 2,0:POKE 842,13:STOP
```

©

Perfect Commodore INPUTs

A one-line cure for accidental program exits during keyboard input.

Blaine D. Standage, Orange, CA

Solve the problem of inputting from the VIC, PET/CBM, 64 keyboard once and for all with one simple line of BASIC code. Why clutter your program with complex subroutines when there is a better way?

I waited a long time for someone to write this article, but no one did. Meanwhile, I kept seeing involved subroutines offered as solutions to the "input problem."

The "problem" is that when you hit the RETURN key in response to an INPUT statement without first giving a Commodore computer some data, it promptly dumps you out of the program – often a very undesirable result. The same thing happens if you accidentally hit the STOP key while the computer is in a GET loop waiting for data. (10 GET A$: IF A$=" " THEN 10). Since most of the proposed solutions use GET loops, it seems that they only move the problem by a keywidth rather than solve it.

Following the KISS (Keep It Simple, Sam) method, let's define our ideal objectives and try to take a completely fresh approach to the problem.

Avoid Subroutines

The highest probability of an operator disrupting a program occurs when he is responding to an INPUT or a GET command. At that time we need to simplify his task by preventing him from accidentally halting the computer.

To keep it simple for the programmer, we need to avoid subroutines, particularly large ones or those written in machine language which may require some form of special handling.

As to the fresh approach, remember that when a thing doesn't work the way you want it to, one alternative is to simply not use it! Oddly enough, that is exactly the way out of this problem. Don't use INPUT or GET.

When we remove INPUT and GET from the instruction list, we are left with only two commands which might take their place, INPUT# and GET#. But aren't they for use with external devices like tape or disk drives? Not exactly. They are the complete form of the commands for inputting. INPUT and GET are just simplified forms which allow easy access to the keyboard.

Using GET#

The results of investigating GET# showed that, overall, it operates very much like GET. We can't completely solve our problems with it because the program can always be halted with the STOP key.

The most obvious difference occurs when we press a non-numeric key while trying to get a numeric value (i.e., GET#1 A). The computer responds with:

```
?SYNTAX ERROR
READY.
```

"SYNTAX ERROR" doesn't seem a reasonable response, and I haven't found out why it happens. Nor can I see any way to take advantage of it, but maybe someone else can.

Perhaps the biggest potential advantage to GET# becomes evident by recalling that we commonly open output files to both the printer and the screen. We then direct the PRINT# output by selecting the appropriate file with a variable value. Similarly, the keyboard could be included in a selectable group of input devices through the use of GET# or INPUT# with a variable defining the file number.

How About INPUT# ?

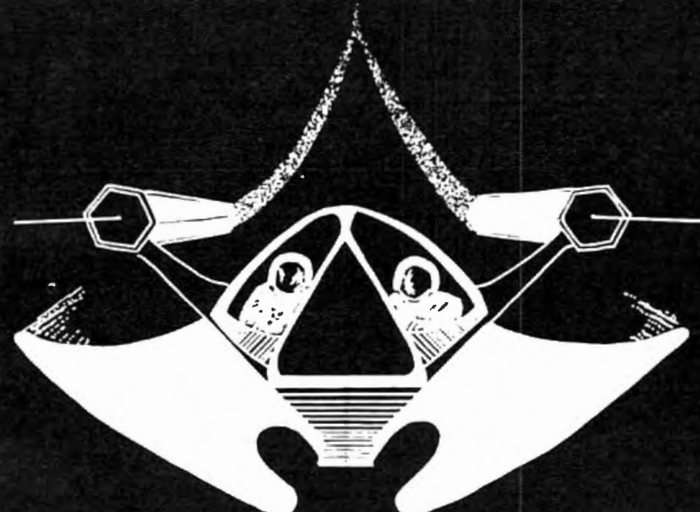
Using INPUT# proved to be the answer, but a lot was learned along the way.

When you enter and run the simple program:

```
10 OPEN 1,0
20 INPUT#1,A$: PRINT A$: GOTO 20
```

all the inputs are echoed back, and there seems to be no way out of the program. RETURNs are ignored, and the STOP key has no effect. Most of my test group (victims, to hear them tell it) decided there was no way to regain control short of cycling computer power. Only the most determined ones discovered that the SHIFTed RUN/STOP would cause a break. (On VIC, this would be Commodore Key/RUN\STOP.) This combination is so unlikely in an input situation that we can almost disregard it as an accidental response.

GAME PROGRAM DEVELOPMENT KIT



for the

COMMODORE VIC - 20

VIC - 20 is a registered trademark of Commodore Business Machines, Inc.

SIX TOOLS TO HELP YOU WRITE YOUR OWN FAST ACTION ARCADE-STYLE GAMES

DECODER — Decodes programs written in machine language (like game cartridges, utility cartridges, and even the computer's own internal operating programs). Produces a program in an English-like language (Assembler) which can be studied to figure out how they did it. The programs created with the decoder can be customized with the **EDITOR** AND INCORPORATED INTO YOUR OWN NEW GAME PROGRAM. The **ASSEMBLER** turns your programs created with the Decoder and the Editor back into machine language and puts them out to tape or disk so the **LOADER** can load them into the computer's memory to be tested and RUN. The **MONITOR** assists you in debugging your new game program by allowing you to run it a step at a time and making modifications if you need to. The **INSTRUCTION GUIDE** is written so that even a beginner can learn the skills needed to become a pro!!!

$49.95 plus **$2.00** p&h buys the kit that could make you rich. Why wait?

Send check, M.O., VISA/MC ($2.00 s.c., please include expiration date), or specify COD (add $3.00) to:

French Silk smooth ware

P.O. Box 207, Cannon Falls, MN 55009

507-263-4821

When I first performed this experiment, my impulse was to consider the problem solved and go on with my programming. What a variety of interesting possibilities I would have missed!

Go to your computer and follow this easy exploration of INPUT#. It will be worth your time.

First, enter the following test program. Use the line numbers as shown because we will be adding to the program.

```
10 OPEN 1,0
20 :
30 INPUT#1,A
70 PRINT A
80 FOR J=1 TO 1500: NEXT J
90 GOTO 20
```

Line 80 has no direct bearing on the problem. It gives you a way to halt the program by providing time to press the STOP key after an input.

When the program is run, notice that the customary question mark is not printed. If we want the question mark we must include it in the input prompting. But we are also free to leave it out for inputs where it is not appropriate.

Next, enter a few numbers and notice they are printed back on the same line as the input even though we made no special provision for that. It seems we will have to print a "return" after inputting to get back to the left margin.

Now press RETURN without any data, and the computer responds as though you had entered a zero. Here is a feature we have wished for. But don't get too excited yet, because when we enter non-numeric characters the computer responds:

?FILE DATA ERROR IN 30
READY.

and we see that we can't solve our problem with a numeric variable input. Why a "file data error"? Remember, the computer thinks it is getting data file inputs from an external device.

Now let's make some changes to our test program to correct the defects we have seen. Unlike INPUT, the INPUT# command has no provisions for built-in prompting, so we must provide it in a separate PRINT command (line 20).

```
20 PRINT "ENTER DATA ? "
30 INPUT#1,A$: PRINT
70 PRINT A$
```

Running this now reveals that we won't stop the program during the input process either with a bad input or with the STOP key. Also notice that a simple RETURN is interpreted as a null (nothing there) string. We can really make use of this feature. (You can prove that the string is really null by manually creating a field of characters, a line of reversed spaces, for example, where the printing will be done. Observe that they are not altered during the execution of line 70.)

Several Handy Features

Of course, we neglected the semicolon at the end of line 20 to force the input to follow its prompt, so let's put it in and re-test.

```
20 PRINT "ENTER DATA ? ";
```

Careful now – something important has changed. A simple RETURN is no longer treated as a null string. It is just ignored. Another feature we can use; no more tests for null inputs – they simply can't happen with this program structure. And so we have RETURN = "ignored" or RETURN = "null" by including or deleting a single semicolon after the prompting message. Very handy.

Let's add lines 40-60 and change 70 to complete our test program for INPUT#. Now it will accept only numeric inputs. The full test program is:

```
10 OPEN 1,0
20 PRINT "ENTER DATA ? ";
30 INPUT#1,A$: PRINT
40 IF A$="0" THEN 60
50 IF VAL(A$)=0 THEN 20
60 B=VAL(A$)
70 PRINT A$;B
80 FOR J=1 TO 1500: NEXT J
90 GOTO 20
```

This structure rejects non-numeric inputs and (because of the semicolon in line 20) will not accept null inputs.

If you want a simple RETURN to be accepted as a zero, delete the semicolon in line 20 and change line 40 to read:

```
40 IF A$="0" OR A$="" THEN 60
```

So we have solved our problem quite nicely for inputs and come up with several very useful options in the process.

Let's make a general observation on the lack of built-in prompting with INPUT#. Neither the double question mark requesting additional inputs nor the "extra ignored" warning is printed. This suggests that you should keep the structure of your input commands as simple as possible, because the operator will not get the usual warnings when he enters incorrect data patterns.

At last, as promised, here is the one-line, no-accidental-exit, solution to the long-standing input problem.

```
10 OPEN1,0:PRINT"PROMPT ? ";:INPUT
   #1,A$:PRINT:CLOSE 1
```

Obviously, the OPEN and CLOSE commands could span a group of inputs or even the entire program. You can safely leave the "file" open while doing other things.

As a bonus, you can accept null inputs by deleting the semicolon in the first print command. This allows you to convert the nulls to zeros for numeric applications.

Finally, I think you will find it interesting to watch your experienced friends try to terminate a program which uses the INPUT# technique. They tend to get very frustrated, which suggests that maybe you shouldn't build escape-proof programs unless they are really needed. ©

Atari Autonumber

Barry M Bernstein, Willowdale, Ontario

This provides quite a convenience when using Atari BASIC. AUTONUM adds automatic line numbering to Atari BASIC. It is used in much the same way as the Atari ASSEMBLER/EDITOR NUM function is used.

The program creates a cassette boot program to be loaded in when you turn the computer on with the BASIC cartridge in place. Once in, it can be called at any time and can quite easily be disengaged or reactivated.

Type in the BASIC program, being especially careful to get the DATA statements correct, and then execute the following statement in direct mode:

A = USR(12288)

You will hear two beeps, signalling you to press PLAY and RECORD on the 410 (with a tape in place) and then pressing the RETURN key. You have just made a boot tape.

To use BASIC AUTONUM, load the boot tape in the 410 tape player, make sure the BASIC cartridge is in place, press the START console switch and turn the computer on. When it beeps, press PLAY on the 410 and then RETURN. The AUTONUM program is now in memory. To activate it execute the following in direct mode:

A = USR(1550,a,b)

where a is the line number to begin at and b is the step size. If b is omitted then a is the step size and it will begin where it left off. If both a and b are omitted then it will begin numbering where it left off (ten to begin with) and increment by tens. You may have to press RETURN twice to activate AUTONUM. To disengage the automatic numbering simply press RETURN twice in a row. It is reactivated by repeating the above procedure.

Though BASIC AUTONUM may take up to half an hour to type in, you will find that it is well worth the effort for the great convenience that it offers, especially when typing in large programs.

```
10 DIM A$(100),B$(2),H$(23):H$="{,}
   {A}{B}{C}{D}{E}{F}{G}{H}{I}!!!!!!!
   {J}{K}{L}{M}{N}{O}":REM ALL CHARAC
   TERS IN BRACKETS ARE CONTROL CHARA
   CTERS
20 MEM=1536:M=-1
30 READ A$
40 FOR I=0 TO 49:B$=A$(I*2+1,I*2+2)
50 IF B$="YY" THEN RESTORE 700:MEM=12
   288:M=-1:GOTO 30
60 IF B$="ZZ" THEN ? "ALL DONE":END
70 N=0:FOR J=1 TO 2:N=N*16+ASC(H$(ASC
   (B$(J))-47)):NEXT J:M=M+1:POKE MEM
   +M,N
80 NEXT I
90 GOTO 30
100 DATA 000200060D06A93C8D02D3186060
    68C900D00BA90085CBA91085CC4C4F068
    5CFC901F00B20D306A5D585CDA5D485CE
    20D306
200 DATA A5D585CBA5D485CCA5CFC902D00F
    F838A5CEE5CC85CEA5CDE5CB85CDD8A98
    08510A9628D0802A9068D0902A9C08510
    60A5CF
300 DATA C9FFF011AD09D2C90CF0034CBEFF
    A9FF85CF4CBEFFAD09D2C90CD015A9808
    510A9BE8D0802A9FF8D0902A9C085104C
    BEFFA9
400 DATA 0085CFF818A5CE65CC85CEA5CD65
    CB85CDD820BD06A5CD20CA06A5CE20BD0
    6A5CE20CA064CBEFF29F04A4A4A4A1869
    3020A4
500 DATA F660290F18693020A4F6606885CC
    6885CB6885D56885D420AAD9A5D4C941F
    00BA5D585D4A90085D54CF706A5D685D4
    A5CB48
600 DATA A5CC4860YY
700 DATA A210A9039D4203A9089D4A03A908
    9D4B03A94A9D4403A9309D45032056E43
    028A90B9D4203A9009D4403A9069D4503
    A9009D
800 DATA 4803A9019D49032056E4300AA90C
    9D42032056E4300068604 33A9BZZ
```

©

VIC Super Expander Graphics

Tim Parker, Kanata, Ontario

Want to see some stunning graphics on your VIC? Type in these short programs and you might be surprised to see what's possible with the Super Expander cartridge.

The VIC-1211A Super Expander is a plug-in cartridge for the VIC-20 that provides several extra features to the graphics and sound abilities. It also adds an extra three kilobytes of memory, giving a power-up of 6519 bytes free. (The missing bytes are used by the expander.)

Program 1 is a short routine that draws a grid on the screen, then selectively erases parts. This is done by drawing vertical and horizontal lines in a character color, then redrawing at a random interval with the screen's color. When RUN for several cycles, the patterns produced can be quite complex. Changing the color of the character in line 20 and the STEP interval in lines 100, 200, 300 and 400 can alter the complexity and appearance.

A variation on this program is to draw the lines on the graphics display diagonally, as Program 2 does. Here, lines 100-220 draw a circular pattern, skipping dots at intervals set by the STEP command. Then, lines 300-420 redraw at intervals in the screen color. The effects are produced as a consequence of the 1024x1024 graphics screen being shortened to 160x160. As the coordinates are altered, some dots will lie on either side of the line. As a result, some dots that are on are turned off, and vice versa.

A long routine to accomplish the above could be arranged using the RDOT(x,y) command, to see if a dot is on, then reverse it. Needless to say, this is an extremely time-consuming task, even in machine language.

Program 3 provides a pattern familiar to most people, although here it is generated in four corners. To see the pattern by itself, leave out lines 110-130. The simple routine here can be enhanced by adding circles concentrically in the center, or by repeating sections in the screen color, as above.

Program 4 draws rectangles on the screen concentrically and is then repeated to color in some areas. Again, when this is elaborated, it can have the effect of a moiré pattern, almost achieving movement of its own.

An alternate method of obtaining the concentric rectangles of Program 4 requires drawing squares with multiple TO's in the DRAW statement (Program 5). Repeating the pattern without a screen-clear command (SCNCLR) produces overlapping bands in the pattern. The pattern can be inverted (i.e., have the rectangles drawn from the outside in) by rewriting lines 100-140 to step down, instead of up. Naturally, concentric circles can be done the same way, by changing line 120 to read:

```
120 CIRCLE1,511,511,X,X
```

This actually produces ellipses, as the axes are not of equal length. This can be changed to produce true circles by adding a constant parameter to the X-axis value.

These programs are by no means as sophisticated as can be achieved with the Super Expander, but they do fill the need for a basic subroutine library on which to base future graphics displays. Combining these with PAINT commands can produce some interesting effects. The Super Expander cartridge's graphics abilities are limited only by the resolution of the graphics screen.

Possible future work for examination of the commands available includes drawing Archimedes' spiral, a herringbone-grid of diagonals, and changing to multicolor graphics to build up a quilt-like display.

Program 1.

```
10 GRAPHIC 2
20 REGION 5
50 DEFFNA(X)=INT(RND(1)*X)+1
100 FOR X=1 TO 1023 STEP FNA(40)+10
110 DRAW1,X,ØTOX,1023
120 NEXT
200 FOR Y=1 TO 1023 STEP FNA(40)+10
210 DRAW1,Ø,YTO1023,Y
220 NEXT
300 FOR X=1 TO 1023 STEP FNA(40)+20
310 DRAWØ,X,ØTOX,1023
320 NEXT
400 FOR Y=1 TO Ø123 STEP FNA(40)+20
410 DRAWØ,Ø,YTO1023,Y
420 NEXT
500 GOTO 100
```

Program 2.

```
10 GRAPHIC 2
20 REGION 5
50 DEFFNA(X)=INT(RND(1)*X)+1
100 FOR X=1 TO 1023 STEP FNA(10)+10
110 DRAW1,X,0TO1023-X,1023
120 NEXT
200 FOR X=1 TO 1023 STEP FNA(10)+10
210 DRAW1,1023,XTO0,1023-X
220 NEXT
300 FOR X=1 TO 1023 STEP FNA(10)+20
310 DRAW0,X,0TO1023-X,1023
320 NEXT
400 FOR X=1 TO 1023 STEP FNA(10)+20
410 DRAW0,1023,XTO0,1023-X
420 NEXT
500 GOTO 100
```

Program 3.

```
10 GRAPHIC 2
20 REGION 5
50 DEFFNA(X)=INT(RND(1)*X)+1
100 FOR X=1 TO 1023 STEP FNA(70)+10
110 DRAW1,X,0TO1023,X
120 DRAW1,0,XTOX,1023
130 DRAW1,X,0TO0,1023-X
140 DRAW1,1023,XTO1023-X,1023
150 NEXT
```

Program 4.

```
10 GRAPHIC 2
20 REGION 5
50 DEFFNA(X)=INT(RND(1)*X)+1
100 FOR A=1 TO 2
110 FOR X=1 TO 1023 STEP FNA(10)+10
120 DRAW1,1023-X,XTOX,0+X
130 DRAW1,X,1023-XTO0+X,X
150 NEXTX,A
```

Program 5.

```
10 GRAPHIC 2
20 REGION 5
50 DEFFNA(X)=INT(RND(1)*X)+1
100 X=FNA(20):X1=X
    120 DRAW1,511-X,511-XTO511+X,511
    -XTO511+X,511+ XTO511-X,511+XTO5
    11-X,511-X
130 X=X+X1
140 IFX<511THEN120
```

©

VIC20™ PERSONAL COMPUTER

TOP 10 SALE!!

ARCADE GAMES

(VIC-20 TAPE PROGRAMS)

Rank	Name	List	Sale
1.	Super Paratrooper (Fantastic)	$24.95	$19.95
2.	Exterminator-Plus (Better than Centipede)	$24.95	$19.95
3.	Cricket (Better than Frogger)	$24.95	$19.95
4.	3-D Hackman (3-Dimensional)	$24.95	$19.95
5.	Snackman (Better than Packman)	$19.95	$15.95
6.	Bug Blast (Creepy)	$19.95	$16.95
7.	Anti Matter Splatter (Nuclear Disaster)	$24.95	$19.95
8.	Bombs Away (Great)	$18.95	$15.95
9.	3-D Maze-Escape	$16.95	$14.95
10.	Krazy Kong	$16.95	$14.95

BUY ANY FOUR — DEDUCT 10% MORE

VIC-20 ACCESSORY SALE!!

1. TRACTION-FRICTION LINE PRINTER This new COM-STAR deluxe printer, prints 8½" x 11" full size letter quality single, roll or fan fold paper, labels and etc. Impact dot matrix bi-directional 40, 66, 80, 132 columns. Includes interface cable that plugs direct into the VIC-20 computer, no other costly interface is needed. List $599.00 — Sale $399.00.

2. UP TO 60K EXPANSION MODULE Aero Space designed—6 slot—add up to 6 cartridges—switch select any program. Start and stop any program with reset button—not necessary to remove cartridges or turn off computer, saves time, television and computer (one year warranty) List $149.00 — Sale $89.00.

- **10 DAY FREE TRIAL**
- **WE HAVE THE LOWEST PRICES**
- **ONE DAY DELIVERY EXPRESS MAIL**
- **FREE CATALOGS**
- **WE LOVE OUR CUSTOMERS!**

PROTECTO ENTERPRIZES (FACTORY-DIRECT)

BOX 550, BARRINGTON, ILLINOIS 60010
Phone 312/382-5244 to order

Syntax SOFTWARE INC

VIC-20

CASSETTE SOFTWARE

CRABS

A new challenge every time

Agility is the key to successfully guiding HERBIE (the halibut) through the maze, avoiding the deadly gaze of SONIC CRABS while feeding on delectable night crawlers

The more you eat, the higher your score. Each time you clear the maze of tasty morcels, you will receive more time, additional lives, and a new group of night crawlers, as the game of SURVIVAL continues

But beware! With the passing of time your presence becomes increasingly aggravating to the KILLER crabs who lurk within, improving the accuracy of their menacing sonic waves

Set at beginner or advanced levels, each game is played in a totally new maze, and may consist of any number of rounds that start identically for each player

CRABS can be played using your VIC-20 keyboard or joystick, and will work on all standard VIC-20 memory configurations.

TANK WAR

Exciting action for two players

Your opponent watches closely as the BATTLEFIELD unfolds, and you both carefully plan strategies for the pending CONFLICT Suddenly, both LASER TANKS fire to initiate movement You begin to thread the way through your home territory, avoiding obstructions and buildings, as you proceed toward enemy ground

Outscore the rival tank by destroying enemy buildings as well as placing direct hits on your opponent during one to one combat. Higher skill levels will add additional targets, mountain ranges and landmines to the battle zone for increasing EXCITEMENT

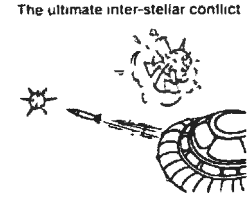
One of three skill levels, with a new battlefield created for each game, provides a new challenge for both players every time

TANK WAR may be played using your VIC-20 keyboard or paddles, and will work on all standard VIC-20 memory configurations

CYCLONS

Full Hi-Res Graphics, Arcade-Like Action

The ultimate inter-stellar conflict



Continuing with their plan to conquer the universe, the CYTRON EMPIRE has chosen your sector as the first target in our galaxy As COMMANDER of the protective forces, you must manoeuvre your craft, avoiding collision and enemy missiles, to attack and destroy enemy war ships

The CYCLON fighters relentlessly enter the battle zone, attempting to lure you into making errors that will lead to your destruction The menacing PULSAR DEATH SHIP also begins to attack, its only purpose to zero in on your location, chase you down, and put an end to your defense of civilization as we know it

Our future lies with your skill

CYCLON requires memory expansion to function When loaded on a system with a 3K expander (or Super Expander) you will play an advanced level game Loading the cassette onto a system with 8K or more expansion, you will be allowed to choose between a variety of difficulty/game-feature options. The game is controlled with the VIC-20 joystick.

CRITTERS

Full Hi-Res Graphics, Arcade-Like Action

Skill-testing defense against a new evil

While inspecting his prize pumpkin patch, Mr GREENSLEEVES becomes aware of a flock of strange CRITTERS hovering in the sky above Without warning small groups begin to leave the formation and dive in order to knock him down and STEAL the fruits of his labour Armed only with a revolver, he must now DEFEND his crop against this new blight

You will guide Greensleeves in his COURAGEOUS effort to save the patch Run or crouch in order to avoid the swooping MENACE, and attempt to exterminate the critters before they can loot the entire crop Most important, once a pumpkin is stolen, destroy the thief before he can reach the flock (taking care not to hit the pumpkin) or his prize will be your loss

As the struggle progresses, larger flocks will arrive and the speed of their attack will increase. But don't despair New pumpkins will grow with your point total providing additional opportunities to successfully fend off the raid When they succeed in clearing the field, the conflict is over.

CRITTERS requires a minimum of 8K memory expansion and is controlled with your VIC-20 joystick

Check for availability with your local dealer, or use the order form provided.
Dealer enquiries are invited.

FORWARD TO: SYNTAX SOFTWARE INC, 33 ELMHURST AVE., SUITE 502
WILLOWDALE, ONTARIO, CANADA M2N 6G8 **PHONE (416) 221-8008**

☐ CERTIFIED CHEQUE
☐ MONEY ORDER
Please Forward Charges To
☐ VISA
☐ MASTER-CARD
☐ AMERICAN EXPRESS
CARD NO
EXPIRY DATE

PLEASE SEND ME

___ **CRABS @ $15.95 (U.S.)/$18.95 (CDN.)** ______
___ **TANK WAR @ $15.95 (U.S.)/$18.95 (CDN.)** = ______
___ **CYCLONS @ $19.95 (U.S.)/$23.95 (CDN.)** ______
___ **CRITTERS @ $19.95 (U.S.)/$23.95 (CDN.)** ______
Shipping and Handling @ $1.00 per Cassette = ______
Ontario Residents Please Add 7% Sales Tax = ______
TOTAL ______

SIGNATURE ______
Please Print
NAME ______
ADDRESS ______
POSTAL CODE/ZIP ______

VIC-20 is a registered trademark of Commodore Business Machines Inc

Download/Upload For The Atari

Frank C Jones, Silver Spring, MD

Use this to transfer programs and text files to or from your Atari using a modem and the telephone lines.

The program described in this article was developed over several months, with feedback from many people. I started writing it primarily because I was too cheap to go out and buy one. I wanted to try out my new communications hardware and look into some of the electronic bulletin boards that I had heard about. Furthermore, I used a mainframe computer in my work, and I thought that it would be convenient to be able to access it from the privacy of my home.

My first approach was to copy a short BASIC program by Henrique Veludo from **COMPUTE!** (February 1981, #9). This worked fine until I realized that a lot of the text went by too fast for me to read. I tried POKEing the incoming characters into a string, hoping to print it out later, but this was too slow; characters were being dropped, making things a bit hard to read. This led to an assembly language routine to speed things up a bit, and before I knew it I was on my way.

It wasn't long before I had added the upload capability so that I could transfer programs and text files to friends who had computers and modems. About this time I joined the downtown Atari club of Washington, D.C., and discovered their ARMUDIC bulletin board, developed and operated by Frank Huband.

After several weeks of enjoying the capabilities of this system and downloading lots of useful programs, I discovered that some of the members did not have terminal programs that would do some of the things that mine would do. I offered to give my program to the club and subsequently uploaded it to the ARMUDIC BBS.

I got calls about problems. I got calls with complaints. I got calls with suggestions. When I next talked to Frank Huband, I discovered that he had picked up a few suggestions and complaints too. We started working together to incorporate as many of the more reasonable ideas as we could, and over the next few months the program grew.

Since this article was intended to be utilitarian rather than tutorial, I have included no discussion of how the program does what it does. Instead, I have included complete instructions on its use and the BASIC listing. There are, however, a few peculiarities about the program that should be pointed out before we get into its operation.

Two Cautions

You may have already noticed that the statements on line 90 are preceded by a REM so that they are not executed. This line is to be used only by those people who find that the BREAK function does not work as described on their machines. The fault lies not in the computer, but in the 850 Interface Module.

Whenever concurrent I/O is turned on, the RS-232 port handler substitutes its own interrupt handlers for the ones in the OS ROM. This is necessary because concurrent I/O handles the serial bus interrupts differently from the way the operating system handles them. The machine language portion of JTERM detected pressing of the BREAK key by sensing what the 850 interrupt handlers did with it. Of course, this was too good to last: newer versions of the 850 Module handle the BREAK key by ignoring it (undocumented).

This leads us to line 90; it's a patch into the interrupt handler (new version) that enables the BREAK key. *Warning:* if the BREAK function works on your machine without line 90, *don't use it;* it will cause a crash if used with the older version handlers. So try it without line 90 first, and remove the REM if you find the BREAK key does nothing when you press it.

A further warning: this program should not be renumbered unless the subroutine at lines 2080-2110 is changed. This routine removes all of the data statements and initialization code after they are used to gain as much memory space as possible for the text buffer. If the program is renumbered and this routine is not changed accordingly, it will perform fatal surgery, and whatever is left won't be of much use. (To find out more about how this routine works, see my article in *COMPUTE!'s Second Book Of Atari.*)

For those of you who have some download capability already, the latest version of this program is available (free) from the ARMUDIC BBS (202) 276-8342. The program is available under the name of JTERM32 (or JTERM33 or... who knows what version will be available by now; this thing seems to have a life of its own). For the rest

of you, type it in, being very careful with the data statements; an error here will probably make the system crash. Then read the instructions and start communicating.

Starting Up

JTERM is a download/upload terminal program for the Atari. It was written in BASIC and assembly language to combine ease in setting up and speed when in the actual terminal mode.

Note that an AUTORUN.SYS file with the RS-232 handler boot routine *must* be on disk and booted when you turn on your machine, if you are using a disk. The DOS command will cause the RS-232 handlers to be overwritten, so you must either have a MEM.SAV file on your disk or re-boot the handler after making a DOS call. Furthermore, it appears that the NEW command damages or wipes out the port handler as well. Therefore, you should always reboot the handler after using this command. An additional note: JTERM assumes that the modem is connected to RS-232 port #1. Also, the 850 Interface must be switched on *before* booting the disk.

JTERM is LOADed as a BASIC program, and when RUN it POKEs the machine language routine into a string called PROG$. While this is going on, the screen is black. After this set-up period is over, the first menu appears on the screen, along with information about the size and location in RAM of the available text buffer. All menu choices are made by simply typing the appropriate key that is highlighted in inverse video. (Type an ordinary character, not an inverse video one.)

The first choice to make is whether you wish to Download a file from the host computer or Upload a file to the host computer. If you wish to do simple communication without file transfer, Download is the proper mode to choose. After you choose between the Download and Upload modes, your next choice is between no translation (None), Light translation, and ATASCII. With Light translation, all high order bits are stripped from all outgoing and incoming characters, and the ATASCII EOL character (155) is changed to the ASCII CR character (13) on output and vice versa on input. No translation and ATASCII modes means that the 850 Interface Module does no changing of characters during either input or output. However, be warned that the program does some translation itself – more about that later.

The next choice is between the various modes of outgoing parity setting. (Note: incoming parity is not checked or changed by this program.) You should always select None if you have selected no translation because setting the parity on output will change the high order bit that was presumably to be preserved. This option was included for the users who wish to access mainframe computers that require certain parity configurations.

At this point, if you choose Upload you will be asked for the filespec of the file to be uploaded. When this has been entered, the file will be loaded into RAM and then listed to the screen as a check. You will then enter the terminal mode. But if you choose Download, you will go directly from the parity choice to the terminal mode without going through the file loading routine.

Terminal Operations

Whenever you enter the terminal mode, the flag (i.e., inverse video word) TERMINAL will appear at the top of the screen. This informs you that you are now in the machine language portion of JTERM. While you are in this mode, you may send data to and receive data from a host computer, provided all of the appropriate connections have been made. You may toggle the memory save function off and on by pressing the SELECT button; the flags MEMSTORE ON and MEMSTORE OFF will be printed on the screen as you toggle the memory.

While the memory save option is in effect, all incoming characters will be stored in sequence until the buffer is full. If the buffer should fill up, the flag MEMORY FULL will be printed on the screen. Note: If you have filled your buffer prior to an Upload, you should *not* turn on the memory save feature until you have completed the Upload. Otherwise, the incoming characters will overwrite your file.

When you enter the terminal mode, you will be in full duplex (i.e., only those characters that are received are printed on the screen and stored in memory). If the host computer echoes all characters that it receives, these characters will be incoming and will be printed and saved if desired. If the host computer operates in half duplex, it cannot send and receive at the same time, so it will not echo the characters that it receives from you. In this case you should turn on the half duplex mode. You can toggle between half and full duplex by pressing the OPTION button. Whenever you do, the flags HALF DUPLEX and FULL DUPLEX will be printed on the screen as appropriate.

Leaving Terminal Mode

When you are ready to leave the terminal mode, you may do so by pressing the START button. When you do, one of three things will happen, depending on the circumstances. If you have chosen the Upload option and have not yet uploaded the file, you will go into the upload mode. The flag UPLOADING will appear on your screen, and the buffer will be printed, 25 characters at a time, to the computer on the other end of the line. During this period you will still see all incoming characters displayed on your screen, so, if the host computer is echoing your transmission, you can watch the upload progress. When the upload is complete, you will reenter the terminal mode, this time in Download mode.

If you exit the terminal mode without saving anything to memory, you will automatically bail back out to the main menu, and you may start another session with different parameters if you wish.

If you were in Download mode and saved anything at all in memory, when you press START you will be asked for the filespec of the file to which you wish to save your memory, or you can type RETURN for further options (more about this in a moment). If you enter a filespec, this can be the cassette (C:), the printer (P:), the screen editor (E:), or a disk file (D:FILENAME). After you enter the filespec, the saved memory will be written to the file, and you will be told that you may reenter the terminal mode by pressing START.

If, however, you wish to save the memory to another file before returning to the terminal mode, press START and, *before releasing the start button*, press the OPTION button. This will bring you back to the request for a filespec. This may be repeated as many times as wished.

If you simply type RETURN instead of a filespec, you will be presented with a menu of three alternative choices. Pressing OPTION will erase your memory buffer and return you immediately to the terminal mode without changing any parameters. Pressing START will erase the buffer and return you to the main mode selection menu where you may reset any or all parameters. Finally, pressing SELECT will return you to the main mode menu while *preserving* everything saved in the memory buffer.

Internal Translations And Other Features

When you choose between Light and No translation in the second menu, you are choosing the configuration of your 850 Interface Module RS-232 ports. You should read your 850 instruction manual for information about these configurations. This program does some additional translation of its own, however.

First of all, nothing that comes in from the port is changed at all before it is stored in memory. Therefore, if you choose ATASCII or No translation for your port, you will save in memory everything *exactly* as it was sent. There will be some translation, however, before it is displayed on the screen. For example, no control characters (ASCII values < 32) are displayed. This means, for instance, that you will not see line feeds; they will, however, be stored and can mess up a program that you are downloading. You should *not* ask for line feeds; you do *not* need them even if the test messages are single spaced.

The carriage return character (ASCII 13) is translated to the ATASCII EOL character. The printer bell character (ASCII 7) is translated to the console bell (ATASCII 253). Finally, the ASCII backspace character (ASCII 8) is changed to the ATASCII DELETE/BACKSPACE (ATASCII 126). Again, none of this translation affects what is stored in memory; everything is stored exactly as it is received.

In ATASCII mode, no translation is done on any outgoing characters; everything is sent exactly as it comes from the keyboard. In the No translation mode, two characters are changed. The DELETE/BACKSPACE character is changed to the ASCII backspace character so that the key will have the same function with most host computers that it does in the Atari. Also, the RETURN key or EOL (ATASCII 155) is changed to the ASCII carriage return (ASCII 13) before it is sent. In light translation, the 850 module would do this translation automatically, but in the no translation mode it would not be done. There were enough situations in which inverse video characters (ASCII values >= 128) could be sent and received, but the host computer would still not recognize the EOL character to warrant this feature.

In half duplex operation, after a character has been sent to the port it is handed over to the input routine and handled just like any other incoming character.

An additional feature of JTERM is the ability to send a computer "BREAK" by simply pressing the BREAK key. This will cause the screen to flash, a beep to sound, the flag BREAK to be printed on the screen, and, last, a true break signal (approx. 0.5 sec. of SPACE tone) to be sent. (If all of this doesn't happen, see the discussion in the introduction.) Sending the BREAK signal will not be of much use when you are connected to a BBS since most of them do not recognize it, but it can be essential when you are connected to a mainframe computer whose attention cannot be gotten any other way.

You should note, however, that the BREAK routine passes briefly through BASIC. Should it be pressed more than once in rapid succession, you can cause a standard BASIC BREAK and terminate the program. If you should terminate the program, accidentally or on purpose, do not attempt to re-RUN it. Instead, *restart it with GOTO 100.* When the program is RUN for the first time, all of the DATA statements and most of the initialization statements are removed to make more room for the buffer; it cannot, therefore, be reinitialized without crashing.

One added note: when the terminal mode is entered for the first time, the DTR line on RS-232 port #1 is set for those modems that monitor this line.

Guidelines For Using JTERM

The JTERM menus were designed to provide maximum flexibility in communicating with many different types of computers, terminals, and bulletin board systems. This may cause some confusion, so here are some general guidelines:

- Most often, you will select Download, Light Translation, No Parity, and Full Duplex. This should work fine when communicating with information utilities such as CompuServe and The Source, as well as with most bulletin boards.
- For communicating between Ataris, choose the ATASCII mode instead of Light Translation. This allows full compatibility between characters sent and received. Also select half duplex instead of full duplex.
- For downloading TRS-80 graphics from TRS-80 bulletin board systems, choose No Translation.
- Usually you won't have to select a Parity option unless communicating with a mainframe computer.
- The half/full duplex option accomplishes with software what the half/full duplex switch on some modems does with hardware. This option was included for those users whose modems lack the duplex switch.

```
10 REM JTERM{3 SPACES}(VERSION 3.2){12 SPACES}by Frank C. Jones July10,1982
20 DIM PROG$(379),PROG2$(7),SPOOL$(15),IN$(26)
30 CON=53279:POKE 559,0:IF PEEK(ADR(PROG$))=104 AND PEEK(ADR(PROG$)+378)=216 THEN 50
40 FOR I=1 TO 379:READ A:PROG$(I,I)=CHR$(A):NEXT I
50 DIM MSG$(65):RESTORE 2000:FOR I=1 TO 65:READ A:MSG$(I,I)=CHR$(A):NEXT I
60 DIM S$(5),T$(8),U$(9):FOR I=1 TO 5:READ A:S$(I,I)=CHR$(A):NEXT I:FOR I=1 TO 8:READ A:T$(I,I)=CHR$(A):NEXT I
70 FOR I=1 TO 9:READ A:U$(I,I)=CHR$(A):NEXT I:DIM BR$(7):FOR I=1 TO 7:READ A:BR$(I,I)=CHR$(A):NEXT I
80 FOR I=1 TO 7:READ A:PROG2$(I,I)=CHR$(A):NEXT I:FLAG=0
90 REM FOR I=1 TO 3:READ A:POKE 8457+I,A:NEXT I:FOR I=1 TO 8:READ A:POKE 1663+I,A:NEXT I
95 GOSUB 2080:N=FRE(0)-256:DIM TXT$(N)
100 SETCOLOR 2,9,0:PROG$(200,200)=CHR$(13):PROG$(192,192)=CHR$(8)
110 POKE 82,0:PRINT "{CLEAR}";
120 PRINT N-1;". BYTES OF MEMORY AVAILABLE":PRINT "FROM-";ADR(TXT$);" TO-";ADR(TXT$)+N-2
130 CLOSE #1:OPEN #1,4,0,"K"
140 POKE 752,1:PRINT "{2 DOWN}{TAB}Operation Mode:":PRINT :PRINT "{TAB}";CHR$(196);"ownload":PRINT :PRINT "{TAB}";CHR$(213);"pload"
150 POKE 559,34:POKE 752,0:GET #1,ANS:IF ANS=68 THEN UPLD=0:GOTO 180
160 IF ANS=85 THEN UPLD=1:GOTO 180
170 GOTO 110
180 POKE 752,1:PRINT "{CLEAR}{2 DOWN}{TAB}Translation Mode:":PRINT :PRINT "{TAB}";CHR$(206);"one":PRINT :PRINT "{TAB}";CHR$(204);"ight"
190 PRINT :PRINT "{TAB}";CHR$(193);"TASCII"
200 POKE 752,0:GET #1,ANS:IF ANS=76 THEN MODE=0:GOTO 240
210 IF ANS=78 THEN MODE=32:GOTO 240
220 IF ANS=65 THEN MODE=32:PROG$(200,200)=CHR$(155):PROG$(192,192)=CHR$(126):GOTO 240
230 GOTO 180
240 POKE 752,1:PRINT "{CLEAR}{2 DOWN}{TAB}Parity:":PRINT :PRINT "{TAB}";CHR$(206);"one":PRINT :PRINT "{TAB}";CHR$(207);"dd"
250 PRINT :PRINT "{TAB}";CHR$(197);"ven":PRINT :PRINT "{TAB}";CHR$(211);"et"
260 POKE 752,0:GET #1,ANS:IF ANS=78 THEN PARITY=0:GOTO 310
270 IF ANS=79 THEN PARITY=1:GOTO 310
280 IF ANS=69 THEN PARITY=2:GOTO 310
290 IF ANS=83 THEN PARITY=3:GOTO 310
300 GOTO 240
310 IF UPLD THEN GOSUB 490
320 PRINT "{CLEAR}{2 TAB}";T$:POKE 65,0:IF  NOT FLAG THEN A=ADR(TXT$)
330 CLOSE #2:OPEN #2,13,0,"R":XIO 38,#2,MODE+PARITY,0,"R":XIO 34,#2,192,0,"R":XIO 40,#2,0,0,"R"
340 A=USR(ADR(PROG$),A,ADR(TXT$)+N-1,ADR(MSG$)):IF PEEK(207)=128 THEN 590
350 IF A=ADR(TXT$) AND  NOT UPLD THEN CLOSE #2:GOTO 100
360 ON UPLD+1 GOSUB 430,650
370 IF UPLD THEN UPLD=0:TXT$="":GOTO 320
380 PRINT "PRESS ";S$;" TO RE-ENTER TERMINAL MODE"
390 IF PEEK(CON)<>6 THEN 390
400 IF PEEK(CON)=6 THEN 400
410 IF PEEK(CON)=2 THEN 360
420 GOTO 320
430 CLOSE #2:? "{CLEAR}{4 DOWN}{TAB}ENTER OUTPUT FILENAME":? "{4 SPACES}OR HIT <RETURN> FOR OPTIONS":? :? "{TAB}";
435 POKE 702,64:POKE 65,3:TRAP 560:INPUT SPOOL$:FLAG=0:IF SPOOL$<>"" THEN 470
440 ? "{CLEAR}{2 DOWN}<START> erases buffer; to menus":? "<SELECT> retains buffer; to menus":? "<OPTION> erases buffer; to terminal"
445 I=PEEK(CON):IF I=5 THEN FLAG=1:GOTO 100
450 IF I=6 THEN 100
460 IF I=3 THEN 320
465 GOTO 445
```

```
470 CLOSE #3:OPEN #3,8,0,SPOOL$:IF SP
    OOL$(1,1)="E" THEN SETCOLOR 2,9,0
480 TXT$(A-ADR(TXT$)+1)=" ":PRINT #3;
    TXT$:CLOSE #3:RETURN
490 PRINT "{CLEAR}{4 DOWN}{TAB}ENTER
    UPLOAD FILENAME":PRINT :PRINT "
    {TAB}";:POKE 702,64:INPUT SPOOL$:
    TXT$=""
500 TRAP 560:CLOSE #3:OPEN #3,4,0,SPO
    OL$:TRAP 4:POKE 65,3
510 AD=ADR(TXT$):XX=INT(AD/256):WW=AD
    -XX*256:ZZ=INT((N-1)/256):YY=(N-1
    )-ZZ*256
520 IOCB=3:GOSUB 620:TXT$(QQ+1)=""
530 IF PEEK(883)=136 THEN 550
540 PRINT "ERROR ";PEEK(883);" DURING
     TEXT LOAD":STOP
550 CLOSE #3:PRINT TXT$:FOR I=1 TO 50
    0:NEXT I:RETURN
560 PRINT "{CLEAR}{4 DOWN}{TAB}UNABLE
     TO OPEN ";SPOOL$:PRINT "{TAB}PRE
    SS ";S$;" WHEN READY"
570 IF PEEK(CON)<>6 THEN 570
580 GOTO PEEK(186)+256*PEEK(187)-10
590 CLOSE #2:SETCOLOR 2,13,10:SOUND 0
    ,30,10,15:XIO 34,#2,2,15,"R":FOR
    I=1 TO 20:NEXT I:XIO 34,#2,3,0,"R
    "
600 SOUND 0,0,0,0:SETCOLOR 2,9,0
610 PRINT BR$:GOTO 330
620 POKE 834+IOCB*16,7:POKE 836+IOCB*
    16,WW:POKE 837+IOCB*16,XX:POKE 84
    0+IOCB*16,YY:POKE 841+IOCB*16,ZZ
630 K=USR(ADR(PROG2$),IOCB*16)
640 QQ=PEEK(840+IOCB*16)+256*PEEK(841
    +IOCB*16):RETURN
650 PRINT "{CLEAR}{4 DOWN}{2 TAB}";U$
660 LL=LEN(TXT$):LN=INT(LL/25)+1
670 FOR I=1 TO LN
680 IF I=LN THEN PRINT #2;TXT$((I-1)*
    25+1);:GOTO 700
690 PRINT #2;TXT$((I-1)*25+1,I*25);
700 STATUS #2,B:BY=PEEK(747):IF BY TH
    EN GET #2,A:PRINT CHR$(A);:GOTO 7
    00
710 NEXT I
720 FOR I=1 TO 20
730 STATUS #2,B:BY=PEEK(747):IF BY TH
    EN GET #2,A:PRINT CHR$(A);:GOTO 7
    30
740 NEXT I
750 PRINT "{TAB}UPLOAD COMPLETE":FOR
    I=1 TO 500:NEXT I:RETURN
1000 DATA 104,104,133,213,104,133,212
     ,104,133,215,104,133,214,104,133
     ,225,104,133,224,169,128,133,216
     ,169,0
1010 DATA 133,226,133,207,172,31,208,
     192,7,240,112,192,6,208,1,96,192
     ,5,208,32,172,31,208,192,5
1020 DATA 240,249,164,216,192,255,240
     ,90,152,73,128,133,216,208,6,169
     ,12,133,217,208,36,169,25,133,21
     7
1030 DATA 208,30,192,3,208,67,172,31,
     208,192,3,240,249,164,226,152,73
     ,128,133,226,208,6,169,51,133
1040 DATA 217,208,4,169,38,133,217,24
     ,165,224,101,217,141,68,3,165,22
     5,105,0,141,69,3,169,14,141
1050 DATA 72,3,169,0,141,73,3,169,11,
     141,66,3,162,0,32,86,228,169,0,2
     40,2,240,137,173,252
1060 DATA 2,201,255,240,54,162,32,169
     ,11,157,66,3,169,0,157,72,3,157,
     73,3,162,16,157,72,3
1070 DATA 157,73,3,169,7,157,66,3,32,
     86,228,201,126,208,4,169,8,208,6
     ,201,155,208,2,169,13
1080 DATA 162,32,32,86,228,164,226,20
     8,50,165,17,208,9,169,128,133,17
     ,133,207,96,240,243,162,32,169
1090 DATA 13,157,66,3,32,86,228,173,2
     35,2,201,0,240,163,169,7,157,66,
     3,169,0,157,72,3,157
1100 DATA 73,3,32,86,228,192,154,240,
     210,164,216,208,10,162,0,129,212
     ,230,212,208,2,230,213,201,13
1110 DATA 208,4,169,155,208,20,201,7,
     208,4,169,253,208,12,201,8,208,4
     ,169,126,208,4,201,32,144
1120 DATA 18,160,11,140,66,3,160,0,14
     0,72,3,140,73,3,162,0,32,86,228,
     165,215,197,213,144,16
1130 DATA 240,2,208,136,165,214,197,2
     12,144,6,240,4,169,0,240,135,169
     ,255,133,216,165,224,141,68,3
1140 DATA 165,225,141,69,3,169,13,141
     ,72,3,169,0,141,73,3,169,11,141,
     66,3,162,0,32,86,228
1150 DATA 169,0,240,216
2000 DATA 155,205,197,205,207,210,217
     ,160,198,213,204,204,155,205,197
     ,205,211,212,207,210
2010 DATA 197,160,207,206,160,155,205
     ,197,205,211
2020 DATA 212,207,210,197,160,207,198
     ,198,155,200,193,204,198,160,196
     ,213,208,204,197,216,160,155
2030 DATA 198,213,204,204,160,196,213
     ,208,204,197,216,160,155
2040 DATA 211,212,193,210,212
2050 DATA 212,197,210,205,201,206,193
     ,204
2060 DATA 213,208,204,207,193,196,201
     ,206,199,155,194,210,197,193,203
     ,155,104,104,104,170,76,86,228
2070 DATA 32,128,6,141,14,210,169,0,1
     33,17,96
2080 POKE 842,13:? "{CLEAR}":POSITION
      2,6:FOR I=1000 TO 1150 STEP 10:
     ? I:NEXT I:? "CONT":POSITION 0,0
     :STOP :LIST 100,200
2090 ? "{CLEAR}":POSITION 2,6:FOR I=1
     0 TO 90 STEP 10:? I:NEXT I:? "CO
     NT":POSITION 0,0:STOP :LIST 100,
     200
2100 ? "{CLEAR}":POSITION 2,6:FOR I=2
     000 TO 2100 STEP 10:? I:NEXT I:?
      "G.2110":POSITION 0,0:STOP
2110 POKE 842,12:RETURN
```

©

Commodore 64 Architecture

Jim Butterfield, Associate Editor

This guided tour of the new Commodore 64 allows you a peek inside the computer's structure and demonstrates some of its extraordinary features.

Let's build a Commodore 64 – at least in principle. We'll put the memory elements together and see how they all fit.

RAM – 64K

We start with a full 64K of RAM. That's the maximum amount of memory that the 6510 chip can address.

If we stopped at this point, we'd have problems. First of all, the screen is fed from memory, but it would contain nonsense. We'll need to put in two extra things: a video chip, and a character generator for the video chip to use. Then again, we have no programs of any sort, and no way to get them into RAM.

Building It Out

Here's what we will do: we'll add the extra features we need by piling them on top of RAM. That way, RAM will be "hidden" – if we look at that part of memory, we will see the new memory elements. But we'll include a set of switches which will allow us to "flip away" the overlaying material and expose the RAM beneath any time we choose. More about these later.

Keep in mind: the RAM is still there, but it's hidden behind the new memory chips.

Input/Output

We'll take the block of memory at hexadecimal D000 to DFFF and reserve it for our interface chips. This includes: two CIA's for timing and input/output, a SID chip for sound, and a video chip to deliver a screen to the television set.

About the 6566 video chip: its "registers" are located at hex D000 to D02E; these locations control how the chip works. But when the video chip needs information to put on the screen, it gets it directly from RAM memory. For example, the usual place for the screen characters is hex 0400 to 07E7. There's a distinction here: we control or check the chip by using its register addresses, but the chip gets information from almost anywhere it likes.

The video chip needs to look at RAM to get characters for the screen. It also needs to look somewhere else to get a "picture" of each character; this allows it to light up the individual dots, or "pixels," that make up a character. There needs to be a table which gives details of each character: what it looks like, and how to draw it. This table is called the "Character Base" table; hardware types may just call it the "character generator."

We could put this character base table in RAM and point the video chip to it. In fact, we are likely to do this if we want to define our own graphics. But on a standard 64, we'd just as soon have these characters built in – in other words, we'll put the character base table into ROM memory.

Now comes the tricky bit. We will put our ROM character base (it's 4K long when we allow for both graphics and text) into locations hex D000 to DFFF. Wait a minute! We just put our interface chips there!

No problem. We just pile the memory elements higher. The ROM character base sits above the RAM, and then we put the I/O on top. Any time we peek these locations, we'll see the I/O. The video chip, by the way, has a special circuit allowing it to go directly to the ROM character base, so there's no confusion there.

If you wanted to look at the character ROM, you'd have to flip it to the top somehow. It turns out you are allowed to do this: clearing bit two (mask four) of address one to zero will do the trick. But be sure you disable the interrupt first, or you're in serious trouble. After all, the interrupt routines expect the I/O to be in place. Bit 2 of address 0 is called the CHAREN control line.

Let's look at a small part of the character base – in BASIC! Be sure to do this on a single line, or as part of a program. First, to turn the interrupt off and back on again:

```
POKE 56333,127: ... ... : POKE 56333,129
```

Now, while the interrupt is in force, flip in the character base:

```
POKE 56333,127:POKE 1,51: ... POKE 1,55:POKE
  56333,129
```

Finally, let's PEEK at part of a character:

```
POKE 56333,127:POKE 1,51:X=PEEK(53248): POKE
  1,55:POKE 56333,129:PRINT X
```

You should see a value of 60; this is the top of the "@" character. To see its pixels, we would write it in binary as ..xxxx.. and to see the next

COMPU SENSE

VIC-20®

VIC-20®	Personal Computer	$179.95
VIC-1011A	RS232C Interface	39.95
VIC-1515	Printer	334.95
VIC-1530	Datasette	67.50
VIC-1540	Disk Drive	349.95
VIC-1111	16K RAM Exp.	99.95
VIC-1110	8K RAM Exp.	52.50
VIC-1210	3K RAM Exp.	34.95
VIC-1010	Expansion Module	139.95
VIC-1311	Joystick	9.95
VIC-1312	Game Paddles	19.95
VIC-1600	Telephone Modem	99.95

BUSINESS & HOME APPLICATIONS

WB101	Total Text 2.5	$24.95
WB102	Total Labels	19.95
WB103	Total Research	24.95
WB104	Predicator-Linear Regression	16.95
PB105	Billing Solver (20 & 64)	19.95
PB106	Utility Bill Saver (20 & 64)	12.95
WB107	The Gasoline UnGuzzler (20 & 64)	15.95
WB107	Accounting	29.95
WB108	Accounts Receivable	21.95
WB109	Calculator	12.95
WB110	Order Tracker	18.95
WB111	Business Inventory	19.95
WB112	Depreciation	10.95
WB113	Ratios	9.95
WB114	Cash Flow	14.95
WB115	Net Worth	14.95
WB116	Lease / Buy	14.95
WB117	Mortgage Calculator	10.95
WB118	Mortgage Comp.	10.95
WB119	Loan Amortiser	24.95
WB120	Loan Repayer	10.95
WB121	Phone Directory	9.95
WB122	Client Tickler	19.95
WB123	Estimates & Bids	14.95
WB124	Bar Charts	9.95
WB125	Stock Ticker Tape	16.95
WB126	Regress on VIC-20, 64	16.95
WB127	P.E.R.T. MY VIC	15.95
WB128	Business Appointments	13.95
WB129	The Predictor-Linear	16.95
PT130	Billing Solver VIC-20, 64	19.95
PT132	Utility Bill Solver VIC-20, 64	12.95
WB134	Nuismatic Panatic VIC-20, 64	15.95
WB135	The Pill Box VIC-20,64	14.95
PT136	Club Lister VIC-20, 64	14.95
WB137	Mother's Recipes VIC-20, 64	12.95
WB139	Terminal 40	29.95
WB140	Minimon	11.79
WB141	Typewriter	27.95
WB142	Data Files	14.95
SB143SS	Mailing List (Tape)	19.95
	(Disk)	24.95

CARDBOARD 6 $79.95

(**Special Christmas Price** — after Dec. 15, back to $99.95) An expansion interface for the VIC-20 — allows expansion to 40K or accepts up to six games — may be daisy chained for more versatility.

CARDBOARD 3 $29.95

Economy expansion interface for the VIC-20

CARD "?" CARD/PRINT $79.95

Universal Centronics Parallel Printer Interface for the VIC-20 or CBM-64. Use an Epson MX-80 or OKIDATA or TANDY or just about any other.

CARDETTE $39.95

Use any standard cassette player/recorder with your VIC-20 or CBM-64.

CARDRITER $29.95

A light pen with six good programs to use with your VIC-20 or CBM-64.

GAMES FOR ALL

WG101	Adventure Land (Cart.)	$37.95
WG102	Mission Impossible (Cart.)	37.95
WG103	Gorf (Cart.)	37.95
WG104	Omega (Cart.)	37.95
WG105	Money Wars	28.95
WG106	Breakout	9.95
WG107	Hangman	9.95
WG108	Monks	7.95
WG109	Tank	17.95
WG110	Simmon	15.95
WG111	Pac Bomber	15.95
WG112	Dam Bomber	15.95
WG113	Cube	14.95
WG114	Spider from Mars	37.95
WG115	Exterminator Plus	19.95
WG116	3-D Hackman	19.95
WG117	Snackman	19.95
WG118	Anti-Matter Splatter	19.95
WG119	Bombs Away	15.95
WG120	3-D Maze Escape	14.95
WG121	Krazy Kong	14.95
WG122	Alien Panic	14.95
WG123	Rescue from Nufon	14.95
WG124	Escapes	14.95
WG125	Journey	14.95
WG126	Invasion	14.95
WK101	Help Chicken Little	14.95
WK102	Mole Attack	19.98
WK103	Car Chase	19.98
WK104	Blue Meanies	11.95
WK105	Space Math	11.95
WK106	Super Slither	11.95
WK107	Spiders of Mars	11.95
WS101	Cribbage	14.95

EDUCATION

WE101	Find the Words (20 & 64)	$10.95
WE102	Temperature Converter (20 & 64)	9.95
PE103	The Mathamagican (20 & 64)	14.95
PE104	The Metric Brain (20 & 64)	10 95
WE105	Money Addition Grades 4th & 5th (20)	9.95
WE106	My Body - Elementary (20)	9.95
WE107	Graphics (20)	17.95
WE108	Diagramming Sentences (20)	12.95
WE109	Fraction Reduction (20)	9.95
WE110	Countries (20)	9.95
WE111	Spell (20)	9.95
WE112	State Capitols (20)	9.95
WE113	Tutor Math (20)	9.95
WE115	Math Whiz (20)	12.95

Prices subject to change.

TO ORDER.
P.O. Box 18765
Wichita, KS 67218
(316) 684-4660

Personal Checks Accepted (Allow 3 Weeks), or C.O.D. (Add $2.00)
Handling Charges $2.00

* VIC-20 is a registered trademark of Commodore, Inc

line of pixels we would repeat the above code with X=PEEK(53249).

Remember that this is ROM; we can PEEK but can't POKE. If we wanted a new character set, we would point the video chip to some new location.

Kernal ROM

To allow the computer to work at all, we must have an operating system in place. The 64's system is called the Kernal: it's in ROM, and placed above RAM at addresses E000 to FFFF.

We can flip the Kernal away and expose the RAM beneath by clearing bit one (mask two) of address one. Be very careful! The computer can't exist for long without an operating system. Either put one into the RAM, or be prepared for a crash.

Even if you flip out the Kernal for a moment, you must be sure to disable the interrupt. The interrupt vectors themselves are in the Kernal; if the interrupt strikes while the Kernal is flipped away, we'll have utter confusion.

Flipping out the Kernal automatically flips out BASIC as well. So bit 1 of address 1, called the HIMEM control bit, switches out both ROMs. We can switch BASIC alone, however, by using bit 0 – the LOMEM control bit.

BASIC ROM

To run BASIC, we have another ROM which is placed above RAM at addresses A000 to BFFF. We may flip it out by clearing bit zero (mask one) of address one.

This is a very useful thing to do. When a word processor, spread sheet calculator, or other program is in the computer, we may not need BASIC at all. Flip it away, and we have extra memory for our program.

Do Your Own BASIC

We can do even more. If we copy BASIC – carefully! – from its ROM into the RAM behind it, we can get BASIC-in-RAM ... a BASIC we can change to meet our own needs.

Let's do this, just to show how. Write the following program into your Commodore 64:

```
100 FOR J = 40960 TO 49151
110 POKE J, PEEK(J)
120 NEXT J
```

Run the program. It will take a minute or so. While it's running, let's talk about that curious line 110. What's the point in POKEing a value into memory identical to what's already there? Here's the secret: when we PEEK, we see the BASIC ROM; but when we POKE, we store information into the RAM beneath.

The program should say READY; now we have made a copy of BASIC in the corresponding RAM. Flip the ROM away with POKE 1,54. If the cursor is still flashing, we're there. BASIC is now in RAM. How can we prove this?

Let's try to fix one of my pet peeves (PET peeves?). Whenever I try to take the ASC value of a null string, BASIC refuses. Try it:

```
PRINT ASC("")
.. will yield an ?ILLEGAL QUANTITY ERROR.
```

Now, it's my fixation that you should be able to take the ASCII value of a null string, and have BASIC give you a value of zero. (Don't ask why; that would take a couple more pages). By peering inside BASIC, I have established that the situation can be changed by modifying the contents of address 46991. There is usually a value of eight there. Normally, we couldn't change it: it's in ROM. But now BASIC is in RAM, and we'll change the ASC function slightly by:

```
POKE 46991,5
```

Now try PRINT ASC(""); it will print a value of zero. In every other way, BASIC is exactly the same.

Just for fun: you can change some of BASIC's keywords or error message to create your own style of machine. For example, POKE 41122,69 changes the FOR keyword ... you must type the new keyword to get the FOR action. Say LIST and see how line 100 has changed. Alternatively, POKE 41230,85; now you must say LUST instead of LIST.

You may go back to ROM BASIC at any time with a POKE 1,55.

Combination Switch

When we use the HIMEM control to flip out the Kernal, BASIC ROM is also removed. Is there any point in flipping both HIMEM and LOMEM? If you do, the I/O and character generator also disappear, giving you a solid 64K of RAM. You can't talk to anybody, since you have no I/O ... but you can do it.

We have named three control lines: CHAREN, which flips I/O with the character base; HIRAM, which flips out Kernal and BASIC ROMs; and LORAM, which controls BASIC. In my memory maps (**COMPUTE!** #29, October 1982), I've called them D-ROM switch, EF-RAM switch, and AB-RAM switch in an attempt to make them more descriptive.

But there are two other control lines, and your program cannot get to them. They are called EXROM and GAME and may be changed only by plugging a cartridge into the expansion slot. When these lines are switched by appropriate wiring

COMPU SENSE!

CARDBOARD 6
$79.95
(SPECIAL CHRISTMAS PRICE — after December 15, back to $99.95)
An expansion interface for the VIC-20. Allows expansion to 40 K or accepts up to six games. May be daisy chained for more versatility.

CARDBOARD 3
$29.95
Economy expansion interface for the VIC-20

CARD "?" CARD/PRINT
$79.95
Universal Centronics Parallel Printer Interface for the VIC-20 or CBM-64. Use an Epson MX-80 or OKIDATA or TANDY or just about any other.

CARDETTE
$39.95
Use any standard cassette player/recorder with your VIC-20 or CBM-64

CARDRITER
$29.95
A light pen with six good programs to use with your VIC-20 or CBM-64

Prices subject to change
TO ORDER. P O BOX 18765
WICHITA, KS 67218
(316) 684-4660
Personal Checks Accepted (Allow 3 Weeks) or C.O D (Add $2) Handling Charges $2 00

YOU DON'T HAVE TO BE A PIRATE TO AFFORD TOTL SOFTWARE
for the VIC 20* and COMMODORE 64*

WORD PROCESSING **$35.00**
Full capability word processing. Word-oriented—not a line editor. Menu-driven. Outstanding features including: footnotes, headings, footing, keyboard input, special printer control, and more. VIC: TOTL.TEXT 2.5 64: TOTL.TEXT 2.6

MAILING LIST and LABELS **$20.00**
Easy editing, automatically sorted, optional non-printing data line(s), browse and select functions. Menu-driven. VIC or 64: TOTL.LABEL 2.0

KEYWORD CROSS REFERENCE **$25.00**
Students and authors: keep track of reference notes and bibliographies. Quick reference by keyword. Requires printer. VIC or 64: RESEARCH ASSISTANT 2.0

TIME MANAGEMENT **$25.00**
Keep track of activities by date (and time). Screen inquiry by date, person, project. 56 different bar chart formats available. VIC or 64: TOTL TIME MANAGER 2.0

BUSINESS ACCOUNTING **(To be announced)**
Accounts receivable and payable. Inventory and expense tracking. Print invoices, statements, reports. Disk only.

SPECIAL VERSIONS AVAILABLE for QUANTUM DATA INC. 40/80 COLUMN VIDEO BOARD.
All programs work with disk and/or tape; VIC or RS-232 printers. VIC requires minimum 8K expansion.

LOOK for TOTL Software at your Dealer or order direct—shipping included.

Send check or money order and be sure to specify machine. California residents add 6% sales tax.

TOTL software
P.O. Box 4742 • Walnut Creek, CA 94596

Call (415) 943-7877

*VIC 20 and COMMODORE 64 are trademarks of Commodore Business Machines.

COMPUTE! Subscriber Services

Please help us serve you better. If you need to contact us for any of the reasons listed below, write to us at:

COMPUTE! Magazine
P.O. Box 5406
Greensboro, NC 27403

or call the Toll Free number listed below.

Change Of Address. Please allow us 6-8 weeks to effect the change; send your current mailing label along with your new address.

Renewal. Should you wish to renew your **COMPUTE!** subscription before we remind you to, send your current mailing label with payment or charge number or call the Toll Free number listed below.

New Subscription. A one year (12 month) US subscription to **COMPUTE!** is $20 (2 years, $36; 3 years, $54. For subscription rates outside the US, see staff page). Send us your name and address or call the Toll Free number listed below.

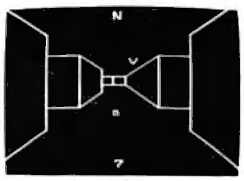
Delivery Problems. If you receive duplicate issues of **COMPUTE!**, if you experience late delivery or if you have problems with your subscription, please call the Toll Free number listed below.

COMPUTE!
800-334-0868
In NC 919-275-9809

VIC-20* COMMODORE

VICTORY SOFTWARE

COMMODORE 64® *



TREASURES OF THE BAT CAVE **$14.95**
Explore the ancient caves filled with treasures and guarded by deadly vampire bats. The realistic 3-D display brings out your claustrophobia. Machine code for fast action: keyboard or joystick. Over $6x10^{23}$ different caves to explore!

ENCODER **$14.95**
Use your VIC to keep prying eyes away from your personal matters. Encoder uses your password to scramble whatever you store in the computer: bank account numbers, household inventory, where you hid the jewelry. The scrambled data can be saved, or retrieved from tape. A 90 minute tape holds approximately 120 double spaced typed sheets. Keep a copy in your safety deposit box.

ADVENTURES*
The best adventures at the best prices! Controlled from the keyboard.

GRAVE ROBBERS* **$14.95**
Introducing the first GRAPHIC ADVENTURE ever available on the VIC-20! Explore an old deserted graveyard. Actually see the perils that lie beyond.

ADVENTURE PACK I* (3 Programs) **$14.95**
MOON BASE ALPHA—Destroy the meteor that is racing towards your base.
COMPUTER ADVENTURE—Re-live the excitement of getting your first computer.
BIG BAD WOLF—Don't let the wolf gobble you up.

ADVENTURE PACK II* (3 Programs) **$14.95**
AFRICAN ESCAPE—Find your way off the continent after surviving a plane crash.
HOSPITAL ADVENTURE—Written by a medical doctor. Don't check into this hospital!
BOMB THREAT—Get back to town in time to warn the bomb squad of the bomb.

ANNIHILATOR **$19.95**
Protect your planet against hostile aliens in this defender-like game. All machine code for fast arcade action. Joystick required.

KONGO KONG **$19.95**
Climb ladders; avoid barrels the crazy ape is rolling at you. Rescue the damsel. Partially machine code for smooth, fast action. Keyboard or joystick.

Send for free catalog
All programs fit in the standard VIC memory, and come on cassette tape.

Ordering—Please add $1.50 postage & handling per order. PA residents add 6% sales tax. Foreign orders must be drawn in U.S. funds or use credit card.
Credit card users—include number and expiration date.

VICTORY SOFTWARE CORP.
2027-A S.J. RUSSELL CIRCLE
ELKINS PARK, PA 19117
(215) 576-5625

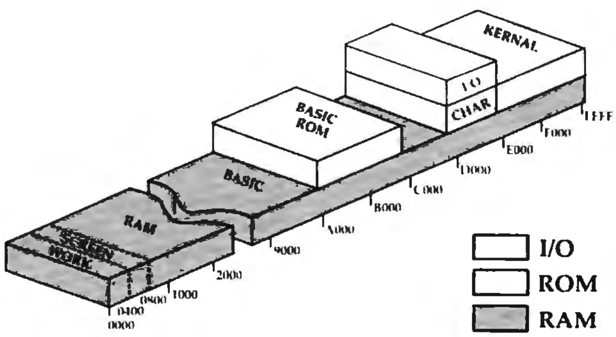
*ONLY ADVENTURES ARE AVAILABLE FOR THE COMMODORE 64 inside the cartridge, the memory map changes once again.

But that's another story.

For the first time, the machine's architecture is at your disposal. If you don't like BASIC, throw it out and replace it with your own. The same is true of the Kernal operating system; it's accessible or replaceable.

New horizons are opening. We'll need to do a lot of traveling to reach them.

Commodore 64 Memory
Addresses shown in hexadecimal.



VIC Pencil

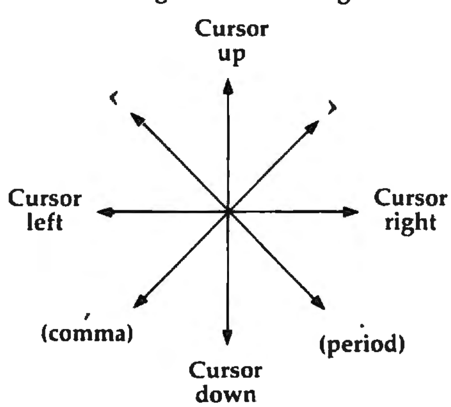
er Bowd, Huntsville, Ontario

This High Resolution Drawing Program will run on a 5K VIC with 1K in reserve. When the RUN command is given, the screen will turn black (POKE 36879,8). You will have to allow a few seconds for the VIC to execute the loops. Hit the "D" key, and one pixel will light on the screen; hitting "E" will extinguish it.

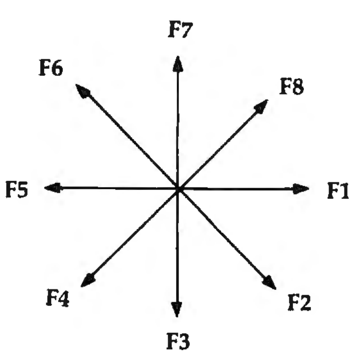
The pencil can be moved either continuously or one pixel at a time. The continuous movement is controlled by the function keys, while single-pixel writing is done with the "," "." "<" ">" and cursor control keys. The pencil will write left, right, up, or down, as well as in four directions diagonally. The "S" key stops all movement of the pencil. It is also possible to move the pencil without writing by hitting the "M" key.

Pencil Control

Single Pixel Drawing



Continuous Drawing



You can clear home or clear the screen by using standard VIC operating procedures. Line 50 reserves some memory from BASIC; line 60 tells the VIC to go to RAM location 7168 for description of characters. Line 70 clears our reserved memory, and line 80 turns the screen black and clears the screen. Lines 100 through 120 set up a mini screen.

Line 130 is the beginning of the main program. It asks the operator for instructions, and from this point down to line 380 the values of X and Y are assigned according to what character was "gotten" from the keyboard. Lines 390 to 420 check to make sure the pencil is on the screen. On down to line 490 the proper bit is selected, and the proper POKE is made to either turn the bit on or turn it off.

Although this is really a novelty program, the kids will probably enjoy drawing on the computer. You will also find after a little practice that it is possible to draw arcs and curves. It should be noted that, although you can't draw on the entire screen, you can address 4096 different spots as compared to the usual 506.

```
50 POKE56,24:POKE52,24
60 POKE36869,255
70 FORI=7168TO7679:POKEI,0:NEXT
80 POKE36879,8:PRINTCHR$(147)
90 FORI=7680TO8191:POKEI,160:NEXTI
100 FORL=0TO7:FORM=0TO7
110 POKE7841+M*22+L,L*8+M
120 NEXT:NEXT
130 GETB$
140 IFB$="D"THENC$="D"
150 IFB$="E"THENC$="E"
160 IFB$="M"THENC$="M"
170 IFB$="{HOME}"THENX=0:Y=0
180 IFB$="S"THENA$=""
190 IFB$="{F1}"ORB$="{F2}"ORB$="{F
    3}"ORB$="{F4}"ORB$="{F5}
    "ORB$="{F6}"ORB$="
    {F7}"ORB$="{F8}"THENA$=B$
200 IFB$="{CLEAR}"THEN220
210 GOTO230
220 FORI=7168TO7679:POKEI,0:NEXT
230 IFB$="{RIGHT}"THENX=X+1
240 IFB$="{LEFT}"THENX=X-1
250 IFB$="{UP}"THENY=Y-1
```

```
260 IFB$="{DOWN}"THENY=Y+1
270 IFB$=","THENX=X-1:Y=Y+1
280 IFB$="<"THENX=X-1:Y=Y-1
290 IFB$="."THENX=X+1:Y=Y+1
300 IFB$=">"THENX=X+1:Y=Y-1
310 IFA$="{F1}"THENX=X+1
320 IFA$="{F3}"THENY=Y+1
330 IFA$="{F5}"THENX=X-1
340 IFA$="{F7}"THENY=Y-1
350 IFA$="{F2}"THENX=X+1:Y=Y+1
360 IFA$="{F4}"THENX=X-1:Y=Y+1
370 IFA$="{F6}"THENX=X-1:Y=Y-1
380 IFA$="{F8}"THENX=X+1:Y=Y-1
390 IFX<0THENX=0
400 IFX>62THENX=62
410 IFY<0THENY=0
420 IFY>62THENY=62
430 CH=INT(X/8)*8+INT(Y/8)
440 RO=(Y/8-INT(Y/8))*8
450 BY=7169+8*CH+RO
460 BI=7-(X-(INT(X/8)*8))
470 IFC$="D"THENPOKEBY,PEEK(BY)OR
    (2↑BI)
480 IFC$="M"THENPOKEBY,0ORPEEK(BY)
490 IFC$="E"THENPOKEBY,0
500 GOTO130
```

©

Atari's Exponents

Matt Giwer, Annandale, VA

The exponential operator, ^, performs a very standard mathematical function, although if you are not familiar with mathematics you may not be aware of its potential. Also, there is another byte-saving use that I will save for the end.

The key to making full use of ^ is to realize that in mathematical notation the square root of four is the same as four to the one-half power. In BASIC you can write either SQR(4) or 4^(1/2). So what good is that? Well, you might want to do a cube root, which would be 8^(1/3). Get the idea? Not believing that this works, you might have tried this by now and have noticed that the machine insists that 4^(1/2) is not 2 but rather 1.998... something. It seems strange to accept a wrong answer from a very slow function.

To correct for this inaccuracy, we simply write the instruction INT (4^(1/2)+0.01), and this will return the number 2. In return for this inaccuracy we get the ability to calculate very unusual powers and roots. The above could have been written 4^0.5 and the same answer returned. Also we could just as easily have written 4^0.4321 or 2^2.223 and have gotten an answer correct enough for many calculations. Also, those complex problems such as two to the five-thirds power 2^(5/3) can be calculated with ease. So not only can we do the more common cube roots by using ^(1/3), but we can now also do an entire range of mathematical functions.

It is not only faster but more accurate to write 2*2 rather than 2^2. If we are not doing mathematics, how do we make use of this? How about instead of writing a byte-consuming timing loop for a beep, we simply write A=1^1? If the beep should last longer, then there is always A=1^1^1^1^1, etc. It takes quite a while before this simple statement equals the number of bytes consumed by a timing loop. Thus the major drawback to more frequent use of ^ can be turned to our advantage. ©

Atari's Exponents

Matt Giwer, Annandale, VA

The exponential operator, ^, performs a very standard mathematical function, although if you are not familiar with mathematics you may not be aware of its potential. Also, there is another byte-saving use that I will save for the end.

The key to making full use of ^ is to realize that in mathematical notation the square root of four is the same as four to the one-half power. In BASIC you can write either SQR(4) or 4^(1/2). So what good is that? Well, you might want to do a cube root, which would be 8^(1/3). Get the idea? Not believing that this works, you might have tried this by now and have noticed that the machine insists that 4^(1/2) is not 2 but rather 1.998... something. It seems strange to accept a wrong answer from a very slow function.

To correct for this inaccuracy, we simply write the instruction INT (4^(1/2)+0.01), and this will return the number 2. In return for this inaccuracy we get the ability to calculate very unusual powers and roots. The above could have been written 4^0.5 and the same answer returned. Also we could just as easily have written 4^0.4321 or 2^2.223 and have gotten an answer correct enough for many calculations. Also, those complex problems such as two to the five-thirds power 2^(5/3) can be calculated with ease. So not only can we do the more common cube roots by using ^(1/3), but we can now also do an entire range of mathematical functions.

It is not only faster but more accurate to write 2*2 rather than 2^2. If we are not doing mathematics, how do we make use of this? How about instead of writing a byte-consuming timing loop for a beep, we simply write A=1^1? If the beep should last longer, then there is always A=1^1^1^1^1, etc. It takes quite a while before this simple statement equals the number of bytes consumed by a timing loop. Thus the major drawback to more frequent use of ^ can be turned to our advantage. ©

VIC Personal Accountant

Peter Mendall, Monmouth, ME

Much ado is made about IRA accounts these days. Can you really be a millionaire, or is this a fantasy even less real than an eight-color printer for your home computer? For unexpanded or expanded VICs.

Here is a program with a four-option menu which demonstrates the growth power of an exponential function. This is especially true if it is non-taxed money!

Remember that IRA money will be taxed sometime and carries a penalty for early withdrawal. However, a 10% one-time "linear" deduction can be offset by an investment compounded exponentially.

This program will provide you with instantaneous delight at the thought of investing ten dollars out of your weekly pay check for 40 years (if you are 25 or younger).

Option one gives you the future value of your choice of payment, interest rate, and payment per year. Option two gives you the payment needed for your choice of future value. The third option prints a table showing how quickly you can multiply your money by factors of two through ten given your interest rate and the frequency it is compounded. The fourth option shows what happens to a one-time investment under compounded interest.

```
1 REM PETER MENDALL
2 REM NO. MAIN ST
3 REM MONMOUTH, ME
5 PRINT"{CLEAR}
10 REM:COMPOUND INTEREST
15 PRINT"{CLEAR}
20 P=PM=I=N=0
30 PRINT"COMP INTEREST MENU"
35 PRINT
40 PRINT"<1>FUT VAL:REG DEP"
45 PRINT
50 PRINT"<2>REG PMT FOR FUT VAL"
55 PRINT
60 PRINT"<3>COMPOUNDING TIME"
65 PRINT
70 PRINT"<4>FUT VAL:SINGLE DEP"
80 PRINT
85 PRINT"<5>END"
92 PRINT
95 PRINT"MENU CHOICE";
97 INPUT X
100 REM PROGRAM
110 ON X GOTO 1000,2000,3000,4000,10000
230 FOR W=2TO10
1000 REM FUTURE VAL
1005 PRINT"{CLEAR}
1010 P=PM=I=N=0
1100 PRINT"P=# OF PER/YR:P=";
1102 INPUT P
1105 PRINT
1110 PRINT"PM=PMT/PER:PM=";
1112 INPUT PM
1115 PRINT
1120 PRINT"I=RATE/YR:(.XX)";
1122 INPUT"I=";I
1125 PRINT
1130 PRINT"N=# OF YR:N=";
1132 INPUT N
1135 PRINT
1140 PRINT"IS INPUT OK? Y OR N"
1145 PRINT
1150 GET A$
1160 IF A$="" GOTO 1150
1170 IF A$="N" THEN 1000
1200 REM CALC I/PER=IR
1210 IR=((1+I)↑(1/P)-1)
1220 N=N*P
1300 REM CALC FUT VAL=FV
1310 FV=(PM*(((1+IR)↑N-1)/IR))
1320 FV=(INT((FV+.005)*100))/100
1400 REM PRINT FUT VAL
1410 PRINT"# PER",N
1415 PRINT"PMT",PM
1420 PRINT"I/YR",I
1425 PRINT"FV",FV
1430 PRINT
1440 PRINT"<1>FV <2>MAIN MENU"
1450 GET X
1460 IF X=0 GOTO 1450
1470 ON X GOTO 1000,5
```

```
2000 REM PMT FOR FUT VAL
2010 PM=FV=IR=N=P=I=0
2020 PRINT"{CLEAR}
2100 PRINT"FV=TOT AMT
     DESIRED"
2110 INPUT"FV=";FV
2115 PRINT
2120 PRINT"I=INT/YR:(.XX)";
2130 INPUT"I=";I
2135 PRINT
2140 PRINT"N=# OF YR:N=";
2150 INPUT N
2155 PRINT
2160 PRINT"P=#PMTS/YR:P=";
2170 INPUT P
2175 PRINT
2180 PRINT"INPUT OK? Y OR N"
2185 GET A$
2190 IF A$="" THEN 2185
2195 IF A$="N" THEN 2000
2200 REM CALC I/PER=IR
2210 IR=((1+I)↑(1/P)-1)
2220 N=P*N
2300 REM CALC PMT FOR  FV
2310 PM=((FV*IR)/((1+IR)
     ↑N-1))
2320 PM=((INT((PM+.005)
     *100))/100)
2400 REM PRINT PM
2410 PRINT"FV",FV
2415 PRINT"#/Y",P
2420 PRINT"I",I
2425 PRINT"PMT",PM
2430 PRINT"<1>PMT<2>MAIN
     MENU"
2440 GET X
2450 IF X=0 THEN 2440
2460 ON X GOTO 2000,5
3000 REM COMPOUNDING TIME
3003 DIM A(12)
3010 P=I=IR=N=0
3030 PRINT"{CLEAR}
3100 PRINT"COMPOUNDING TIME"
3105 PRINT
3110 PRINT"I=INT/Y:(.XX):I=";
3120 INPUT I
3125 PRINT
3130 PRINT"P=COMPOUND FREQ
     /Y:P=";
3140 INPUT P
3145 PRINT
3150 PRINT"INPUT OK? Y OR N"
3155 GET A$
3160 IF A$=""THEN 3155
3170 IF A$="N"THEN 3010
3200 REM CALC I/PER=IR
3204 PRINT"CALC TIME=20-50
     SEC"
3210 IR=((1+I)↑(1/P)-1)
3220 N=1
3230 FOR W=2TO10
3240 N=N+1
3250 X=((1+IR)↑(N))
3260 IF X>=W THEN 3280
3270 GOTO 3240
3280 A(W)=N
3290 NEXT W
3300 REM PRINTING
3400 PRINT"FACT#PER#OF YRS"
```

```
3405 PRINT
3410 FOR W=2TO 10
3412 NY=A(W)/P
3414 NY=(INT((NY+.05)*10))
     /10
3420 PRINTW;TAB(5)A(W);TAB
     (11)NY
3430 NEXT W
3435 PRINT
3440 PRINT"<1>COMP  TIME"
3450 PRINT"<2>MAIN MENU"
3460 GET X
3470 IF X=0 THEN 3560
3480 IF X=1 THEN 3010
3490 GO TO 5
3560 GET X
3570 IF X=0 THEN 3560
3580 IF X=1 THEN 3010
3590 GO TO 5
4000 REM SINGLEDEPOSIT
4010 FV=PV=IR=N=0
4020 PRINT"{CLEAR}
4100 PRINT"PV=SINGLEDEP:PV="
4110 INPUT PV
4115 PRINT
4120 PRINT"I=INT/YR:(.XX):I=";
4125 INPUT I
4127 PRINT
4130 PRINT"#PER/YR:P=";
4135 INPUT P
4137 PRINT
4140 PRINT"# OF YR:N=";
4145 INPUT N
4147 PRINT:PRINT
4150 PRINT"INPUT OK? Y/N"
4160 GET A$
4170 IF A$= ""THEN 4160
4180 IF A$="N" THEN 4000
4190 PRINT"3"
4210 IR=((1+I)↑(1/P)-1)
4220 N=P*N
4300 FV=PV*((1+IR)↑N)
4310 FV=(INT((FV+.005)*100)
     )/100
4400 REM PRINT FV
4405 PRINT"FUT VAL SINGLE
     DEP"
4407 PRINT
4410 PRINT"#PER/YR",P
4412 PRINT
4414 PRINT
4415 N=N/P
4416 PRINT"# OF YR",N
4418 PRINT
4420 PRINT"DEP",PV
4425 PRINT
4430 PRINT"I/YR",I
4435 PRINT
4440 PRINT"FV",FV
4450 PRINT
4455 PRINT
4460 PRINT"<1>SINGLE DEP"
4462 PRINT
4465 PRINT"<2>MAIN MENU"
4470 GET X
4480 IF X=0 THEN 4470
4490 ON X GOTO 4000,5
10000 PRINT"END PROGRAM"
10010 END
```

©

...PET/CBM/VIC? SEE SKYLES...

CAPUTE!

Modifications Or Corrections To Previous Articles

UXB On VIC

To produce the correct alternate character set for the VIC version of UXB (November 1982, p. 56) the following lines must be added to Program 1:

```
596 POKE51,0:POKE58,28:POKE55,0:POKE56,28:CLR:
    CB=7168
597 READ A:IF A=-1 THEN 600
598 FOR N=0TO7:READ B:POKE(CB+A*8+N),B:NEXT
599 GOTO 597
900 DATA 1,153,219,189,153,129,66,36,36
910 DATA 17,126,255,199,203,211,227,255,126
920 DATA 24,36,36,36,36,60,36,66,129
930 DATA 26,4,24,24,60,126,126,126,60
940 DATA 32,0,0,0,0,0,0,0,0
950 DATA -1
```

Pack Up Your DATA

There is an error in line 130 of the example program which condenses PET or VIC data for files (October 1982, p. 162). The line should read:

```
130 V%=V/256:L=V-(V%*256)
```

VIC Superchase

The VIC version of this game (October 1982, p. 66) requires several changes. Lines 10 and 22 should be removed from the program. In line 12100, a semicolon should be added after the CHR$(142). Also in that line, the characters shown as underlined ampersands (&) are typed in by holding down the Commodore logo key and hitting the "+" key. The characters shown as underlined dollar signs ($) in line 14010 are typed in by holding down the logo key and hitting the "@" key.

VIC Pixelator

The author sent in some additions and clarifications on his custom character editor (October 1982, p. 141):

Under certain circumstances, the "Pixdata" program will not adjust the end-of-program pointers causing the program to reappear and interfere with the newly written DATA statements. This can be taken into account by changing two lines in the program and adding two new lines. Notice that you must type the abbreviations pO and pE for POKE and PEEK to fit line 170 into the computer:

```
10 C=PEEK(56):POKE51,0:POKE52,20:POKE55,0:POK
   E56,20
170 A2=INT(AA/256):A1=AA-A2*256:B=PEEK(43)+256
    *PEEK(44):POKEB,A1:POKEB+1,A2:POKE56,C
180 ZZ=ZZ+257-AA+B:Z2=INT(ZZ/256):Z1=ZZ-Z2*256
190 POKE251,Z1:POKE174,0:POKE175,0:POKE46,Z2:P
    OKE45,PEEK(251)
```

Also, memory expansion of 8K or more on the VIC usually moves the screen memory so that there is not enough room between the end of the screen and the beginning of the last available character map area in RAM for the "Pixelator" to operate. Before loading the Pixelator, 8K users should enter the following as a single line and then hit RETURN:

```
POKE43,0:POKE44,2244::POKE45,00:POKE46,24:POKE
    47,0:POKE48,24
```

This moves the start-of-BASIC memory to 6144. Now type:

```
POKE6143,0:POKE6144,0:POKE6145,0
```

and hit RETURN. This simply cleans up the area. Now LOAD the Pixelator, delete line 30, and make this change:

```
20 XX=5120:SC=4096:CL=37888
4010 SYSXX-16:PRINT"{HOME}":PRINT"LOOKING AT  "
    ;S5$:POKESC+33,C0
```

SAVE the program before using it. Make the following change in both Pixaver and Pixeloader:

```
10 XX=5120
```

Make these changes in Pixdata:

```
10 C=PEEK(56):POKE51,0:POKE52,32:POKE55,0:POK
   E56,32
20 XX=5120
40 ZZ=8192:AA=ZZ
```

8K users can access the RAM character set by typing:

```
POKE36869,PEEK(36869)ANDNOT15OR13
```

Finally, the symbols appearing in lines 70, 80, and 90 of the printed listing of Pixelator might be confusing. A more attractive display can be achieved by changing the underlined slashes to Commodore-P (hold down the Commodore logo key while typing "P"); the underlined apostrophe (-) should be Commodore-M; the underlined percent sign (%) should be Commodore-G; and the underlined 7 should be Commodore-Y.

Atari Mathman Improvements

Our thanks to John Bergen for the following enhancements to this program (October 1982, p. 72). With these changes, Mathman can recover the incorrect answer after it is sent off the screen, print the correct response, turn the screen a new color after each trial, and also show a student the current question number:

```
11 Y=INT(RND(0)*15):SETCOLOR 4,Y,6:SE
   TCOLOR 2,Y,4
375 POSITION 0,22:?"SCORE-";SCO;"
    {3 SPACES}TURNS-";O
1070 POSITION 8,10:?"{4 SPACES}":POSI
     TION 10,10:? B*C;"-Correct Answe
     r"
1071 POSITION 8,12:?"{4 SPACES}":POSI
     TION 10,12:? AS;"-Your Answer Wa
     s"
1072 FOR Z=1 TO 200:NEXT Z
```

©

How To Type COMPUTE!'s Programs

Many of the programs which are listed in **COMPUTE!** contain special control characters (cursor control, color keys, inverse video, etc.). To make it easy to tell exactly what to type when entering one of these programs into your computer, we have established the following listing conventions. There is a separate key for each computer. Refer to the appropriate tables when you come across an unusual symbol in a program listing. If you are unsure how to actually enter a control character, consult your computer's manuals.

Atari 400/800

Characters in inverse video will appear like: INVERSE VIDEO
Enter these characters with the Atari logo key, {⅄}.

When you see	Type	See	
(CLEAR)	ESC SHIFT <	↖	Clear Screen
(UP)	ESC CTRL -	↑	Cursor Up
(DOWN)	ESC CTRL =	↓	Cursor Down
(LEFT)	ESC CTRL +	←	Cursor Left
(RIGHT)	ESC CTRL *	→	Cursor Right
(BACK S)	ESC DELETE	◄	Backspace
(DELETE)	ESC CTRL DELETE		Delete character
(INSERT)	ESC CTRL INSERT		Insert character
(DEL LINE)	ESC SHIFT DELETE		Delete line
(INS LINE)	ESC SHIFT INSERT		Insert line
(TAB)	ESC TAB	▶	TAB key
(CLR TAB)	ESC CTRL TAB		Clear tab
(SET TAB)	ESC SHIFT TAB		Set tab stop
(BELL)	ESC CTRL 2		Ring buzzer
(ESC)	ESC ESC		ESCape key

Graphics characters, such as CTRL-T, the ball character ● will appear as the "normal" letter enclosed in braces, e.g. {T .

A series of identical control characters, such as 10 spaces, three cursor-lefts, or 20 CTRL-R's, will appear as {10 SPACES}, {3 LEFT}, {20 R}, etc. If the character in braces is in inverse video, that character or characters should be entered with the Atari logo key. For example, {♥} means to enter a reverse-field heart with CTRL-comma, {5U} means to enter five inverse-video CTRL-U's.

Commodore PET/CBM/VIC

Generally, any PET/CBM/VIC program listings will contain bracketed words which spell out any special characters: {DOWN} would mean to press the cursor-down key; {3DOWN} would mean to press the cursor-down key three times.

To indicate that a key should be *shifted* (hold down the SHIFT key while pressing the other key), the key would be underlined in our listing. For example, S would mean to type the S key while holding the shift key. This would result in the "heart" graphics symbol appearing on your screen. Some graphics characters are inaccessible from the keyboard on CBM Business models (32N, 8032).

Sometimes in a program listing, especially within quoted text when a line runs over into the next line, it is difficult to tell where the first line ends. How many times should you type the SPACE bar? In our convention, when a line breaks in this way, the ~ symbol shows exactly where it broke. For example:

```
100 PRINT "TO START THE GAME  ~
    YOU MAY HIT ANY OF THE KEYS
    ON YOUR KEYBOARD."
```

shows that the program's author intended for you to type two spaces after the word *GAME*.

All Commodore Machines

Clear Screen	{CLEAR}	Cursor Left	{LEFT}
Home Cursor	{HOME}	Insert Character	{INST}
Cursor Up	{UP}	Delete Character	{DEL}
Cursor Down	{DOWN}	Reverse Field On	{RVS}
Cursor Right	{RIGHT}	Reverse Field Off	{OFF}

VIC/CBM 64 Conventions

Set Color To Black	{BLK}	Function Two	{F2}
Set Color To White	{WHT}	Function Three	{F3}
Set Color To Red	{RED}	Function Four	{F4}
Set Color To Cyan	{CYN}	Function Five	{F5}
Set Color To Purple	{PUR}	Function Six	{F6}
Set Color To Green	{GRN}	Function Seven	{F7}
Set Color To Blue	{BLU}	Function Eight	{F8}
Set Color To Yellow	{YEL}	Any Non-implemented Function	{NIM}
Function One	{F1}		

To enter any color code, hold down CTRL and press the appropriate color key. Use CTRL-9 for RVS on and CTRL-0 for RVS off.

8032/Fat 40 Conventions

Set Window Top	{SET TOP}	Erase To Beginning	{ERASE BEG}
Set Window Bottom	{SET BOT}	Erase To End	{ERASE END}
Scroll Up	{SCR UP}	Toggle Tab	{TGL TAB}
Scroll Down	{SCR DOWN}	Tab	{TAB}
Insert Line	{INST LINE}	Escape Key	{ESC}
Delete Line	{DEL LINE}		

When you see an underlined character in a PET/CBM/VIC program listing, you need to hold down SHIFT as you enter it. Since the VIC-20 and Commodore 64 have fewer keys than the PET/CBM, some graphics are grouped with other keys and have to be entered by holding down the Commodore key. If you see any of the symbols in the left column underlined in a listing, hold down the Commodore key and enter the symbol in the right column. Just use SHIFT to enter all other underlined characters.

!	K	←	*	1	E
"	I	↑	PI	2	R
#	T	.	S	3	W
$	@	–	Z	4	H
%	G	=	X	5	J
'	M	<	C	6	L
&	#	>	V	7	Y
\	–	,	D	8	U
;	F	/	P	9	I
?	B	*	N	@	SHIFT*
(	£	+	Q	[	SHIFT+
)	SHIFT-£	0	A	]	SHIFT–

Apple II / Apple II Plus

All programs are in Applesoft BASIC, unless otherwise stated. Control characters are printed as the "normal" character enclosed in brackets, such as {D} for CTRL-D. Hold down CTRL while pressing the control key. You will not see the special character on the screen.

TRS-80 Color Computer

No special characters are used, other than lowercase. When you see letters printed in inverse video (white on black), press SHIFT-0 to enter the characters, and then press SHIFT-0 again to return to normal uppercase typing.

Texas Instruments 99/4

No special control characters are used. Enter all programs with the ALPHA lock on (in the down position). Release the ALPHA lock to enter lowercase text.

Timex TS-1000, Sinclair ZX-81

Study your computer manual carefully to see how to enter programs. Do not type in the letters for each command, since your machine features single-keystroke entry of BASIC commands. You may want to switch to the FAST mode (where the screen blanks) while entering programs, since there will be less delay between lines. (If the blanking screen bothers you, switch to the SLOW mode.)

A Beginner's Guide To Typing In Programs

The first few times you type in a computer program, things can be quite confusing. We have prepared this guide to help new readers adjust to the novelties of program entry.

What Is A Program?

A computer cannot perform any task by itself. Like a car without gas, a computer has *potential*, but without a program, it isn't going anywhere. Most of the programs published in **COMPUTE!** are written in a computer language called BASIC. BASIC is easy to learn and is built into most computers (on some computers, you have to purchase an optional BASIC cartridge).

BASIC Programs

Each month, **COMPUTE!** publishes programs for many machines. To start out, type in only programs written for your machine, e.g., "TI Version" if you have a TI-99/4. Later, when you gain experience with your computer's BASIC, you can try typing in and converting certain programs from one computer to yours.

Computers can be picky. Unlike the English language, which is full of ambiguities, BASIC usually has only one "right way" of stating something. Every letter, character, or number is significant. A common mistake is substituting a letter such as "O" for the numeral "0", a lowercase "l" for the numeral "1", or an uppercase "B" for the numeral "8". Also, you must enter all punctuation such as colons and commas just as they appear in the magazine. Spacing can be important. To be safe, type in the listings *exactly* as they appear.

Brackets And Special Characters

The exception to this typing rule is when you see the curved bracket, such as "{DOWN}". Anything within a set of brackets is a special character or characters that cannot easily be listed on a printer. When you come across such a special statement, refer to the appropriate key for your computer. For example, if you have an Atari, refer to the "Atari" section in "How to Type COMPUTE!'s Programs."

About DATA Statements

Some programs contain a section or sections of DATA statements. These lines provide information needed by the program. Some DATA statements contain actual programs (called machine language); others contain graphics codes. These lines are especially sensitive to errors.

If a single number in any one DATA statement is mistyped, your machine could "lock up," or "crash." The keyboard, break key, and RESET (or STOP) keys may all seem "dead," and the screen may go blank. Don't panic – no damage is done. To regain control, you have to turn off your computer, then turn it back on. This will erase whatever program was in memory, so always SAVE a copy of your program before you RUN it. If your computer crashes, you can LOAD the program and look for your mistake.

Sometimes a mistyped DATA statement will cause an error message when the program is RUN. The error message may refer to the program line that READs the data. *The error is still in the DATA statements, though.*

Get To Know Your Machine

You should familiarize yourself with your computer before attempting to type in a program. Learn the statements you use to store and retrieve programs from tape or disk. You'll want to save a copy of your program, so that you won't have to type it in every time you want to use it. Learn to use your machine's editing functions. How do you change a line if you made a mistake? You can always retype the line, but you at least need to know how to backspace. Do you know how to enter inverse video, lowercase, and control characters? It's all explained in your computer's manuals.

A Quick Review

1) Type in the program a line at a time, in order. Press RETURN or ENTER at the end of each line. Use backspace or the back arrow to correct mistakes.

2) Check the line you've typed against the line in the magazine. You can check the entire program again if you get an error when you RUN the program.

3) Make sure you've entered statements in brackets as the appropriate control key (see "How To Type COMPUTE!'s Programs" elsewhere in the magazine.)

4) Save the program to tape or disk before you RUN it. Consult your computer manuals if you aren't sure how to do this.

5) If you get an error when you RUN the program, check your listing against the published program. ©

tele soft • inc.

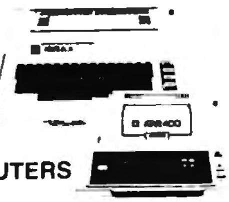
P.O. BOX 3456, TROY, MICH 48099
Complete line of Computers . . . Software
Video Games and Accessories
CALL TODAY!

FREE! STICK STAND with FASTBALL (With purchase of $60.00 or more. A $6.99 value!)

REDUCES HAND & WRIST FATIGUE

NEW!

ATARI HOME COMPUTERS



	400	800
16K	$279.95	$659.00
48K	499.00	799.00

ACCESSORIES

410	Cassette Recorder	$ 89.95
810	Disk Drive	499 99
850	Interface Module	169.95
825	80 Column Printer	699 95
MOSAIC	32K Ram	113.00

Bell & Howell (Black Apple)

Apple II plus 48K	$1287 00
Disk Drive w/Controller	550 00
Disk Drive without Controller	475 00
ACE 16K Expansion	165 00
Parallel Interface	125 00

commodore

Commodore "64"	$525.00
Commodore Vic 20	215.00
Disk Drive	499.00
Datassette	69 00
8K Memory Expander	49 95
RS232 Interface	45 00
Vic 20 Cartridge Games	27 95

BUSINESS MACHINES

CBM 8032	$1100.00
CBM 8050 Dual Disk	1299 00
TRS 80 Model III	1650.00

TIMEX Home Computer	89.95
SANYO Small Business Computer	1700.00
NEC Computer 32K Ram	834.00
XEROX 820 System I	2600.00
TeleVideo 64K Computer	1500 00
Texas Instruments TI99 (with $100.00 rebate)	200.00
HAYES MODEM 1200	575 00

ACCESSORIES

Stick Stand	$ 6.99
2 For	12.00
Atari Joy Stick	6.20
Wico Joy Stick	22.50
Wico Red Ball Joy Stick	24.80
Wico Track Ball Atari/Commandor	52.00
Many More Accessories	CALL

SOFTWARE FOR ATARI

Centipede	$34 75
Pac Man	34 75
Super Breakout	28.50
Missile Command	28.50
Star Raiders	34.75

CBS Software

Krazy Shootout	$36 00
Krazy Kritters	36 00
K-Star Patrol	36 00
Krazy Antiks	36.00

EDYX

Crush, Crumble, Chomp	$23 00
Ricochet	15.50
Star Warrior	30.50
Rescue at Rigel	23 00

Broderbund

Choplifter	$29.95
Apple Panic	23 00
Star Blazer	24 50
David's Midnightmagic	27 00
Stellar Shuttle	29 95

synapse

Nautilus	23.00
Slime	23.00
Dodge Racer	23.00
Protector	23.00
Chicken	23.00

IDSI.

Pool 1.5	27 00

Adventure

Preppie	23.00
Rear Guard	15.50
Treasure Quest	23.00
3D Tic Tac Toe	12.00
Adventure Series	15.50
War	19.50
Diskey	37.00

On-Line Systems

Ultima I	30.50
Ultima II	47.50
Frogger	27.00
Jaw Breaker	23.00
Crossfire	23.00

Datasoft

Shooting Arcade	23 00
Pacific Coast Highway	23 00
Clowns and Balloons	23 00
Atari Character Generator	15 50

Epyx

Andromeda	27.00
Doctor Goodeode's Cavern	24.00
Pathfinder	27.00
Match Racers	24.00

VISICALC

(For Apple, Atari, Commodore, and IBM)	200.00

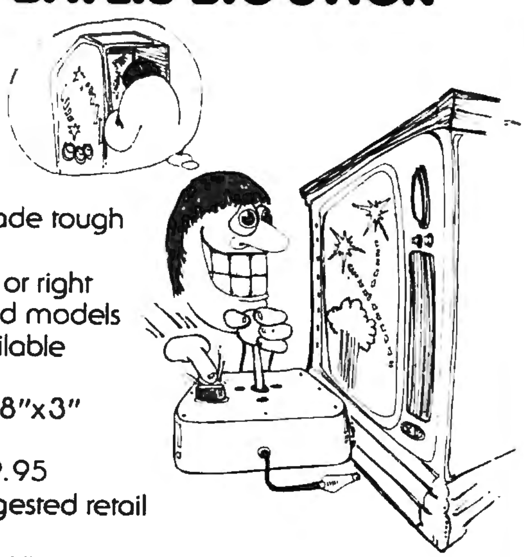
VISA & MASTERCARD ACCEPTED

ORDERING INFORMATION
Check, Money Order, MasterCard, Visa and C.O.D Orders accepted. Add $2.00 for C.O.D. All other orders shipped U.P.S. collect. Michigan residents add 4% sales tax
Hours 9 a.m. to 8 p.m. daily.

CALL FREE 1-800-255-2000
IN MICHIGAN 1-800-742-4242

Introducing the

BAYLIS BIG STICK

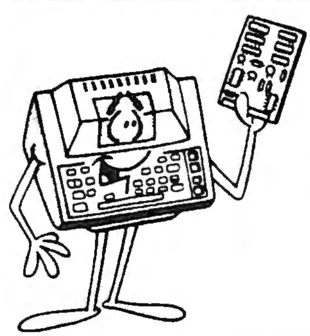


- Arcade tough
- Left or right hand models available
- 7"x8"x3"
- $39.95 suggested retail

The BAYLIS BIG STICK for Atari® and Vic-20® computers by R. Allen Baylis.

Available at Computer Store Int'l. Glendale, CA and other fine dealers.

Released through Torrey Engberg Smith Co.
P.O. Box 1075, Glendale, CA 91209 (213) 247-6484.
Dealer and distributor inquiries welcome.



ATARI * 400 MEMORY

48K + 4

- 52 K BYTES TOTAL MEMORY CAPACITY
- 4K BYTES MORE THAN OTHERS
- USES 64K RAMS FOR REDUCED POWER CONSUMP.
- GOLD PLATED CONTACTS FOR RELIABILITY
- SOLDER MASK AND PARTS LEGEND
- 90 DAY WARRANTY
- COMES WITH ILLUSTRATED "HOW TO" MANUAL AND SPECIAL JUMPERS

→ BARE BOARD	$ 50. ←
KIT	179.
ASSM/TESTED	199.

ECRL, INC.
PO BOX 387
CANBY, OR 97013
(503) 266-4982

ADD $3 SHIPPING PER BOARD. DELIVERY FROM STOCK WITH MONEY ORDER, 3 WEEKS FOR PERSONAL CHECKS.
* ATARI 400 IS A TRADEMARK OF ATARI, INC.

NEWS & PRODUCTS

Peripherals For The VIC

DATA 20 corporation has introduced its new line of four peripherals for the VIC-20: Video Pak, Printer Interface, Expansion Chassis, and Memory Cartridge. All are designed to give the VIC-20 added capability at an affordable price.

The Video Pak cartridge plugs right into the VIC-20 expansion port, instantly giving a standard 24-line display with a choice of 40 or 80 upper-and lowercase characters. It also increases memory from 5K to 20K or even 70K, enabling the VIC-20 to run Commodore business software. Video Pak also includes a terminal emulator and screen print feature.

Printer Interface matches the VIC-20 to most popular printers and gives a continuous visual monitoring of the data transfer functions. Status lights indicate if the printer is hooked up, if the buffer is full, and if data is being transmitted. The Printer Interface comes complete with cable and connector and needs no assembly.

Expansion Chassis lets the VIC-20 run a series of four compatible memory, software, or game cartridges with the standard 22 pin edge connector at the same time, while protecting the VIC-20 power supply with a built-in 500ma fuse.

Memory Cartridge boosts the brainpower of the VIC-20 to 20K and features 200ns RAMs for added reliability. Housed in a rugged plastic case, the Memory Cartridge is an ideal first add-on to any VIC-20.

Screen display options with Data 20's Video Pak Cartridge.

For further information on these four VIC-20 peripherals, contact:

DATA 20 Corporation
20311 Moulton Parkway
Suite B10
Laguna Hills, CA 92653
(714)770-2366

New Products From Krell Software

Krell Software has announced several new products. *Connections* is a game system designed to extend and develop the mental capacities of children of all ages. Users select from a variety of game formats as they search for logical connections and learn the principles of scientific reasoning. *Connections* draws subject matter from all fields of science and the humanities. *Connections* may be played competitively or cooperatively. Players may set difficulty levels and may add additional data as desired.

Available on disks for Apple, Atari, Commodore, Franklin, Radio Shack, and IBM personal computer systems, *Connections* costs $99.95 (includes one data base).

The Amazing Ben is designed to introduce programmers of all ages to the art of artificial intelligence. Extensive documentation is provided to help users create programs which learn from their environment. Ben's language enables him to sense his surroundings, to write and read messages, to ask questions and interpret answers, to move at will across the screen, and to create his own memory structures.

The Amazing Ben set of programs starts by teaching you Ben's language. It guides the user in writing a series of increasingly difficult programs to help Ben traverse mazes that he has never seen before. The user can control Ben and see what he sees as he follows the different mazes while both the user and Ben are learning.

The Amazing Ben sells for $79.95. It is available for Apple, Atari, Commodore, Franklin, Radio Shack, and IBM personal computer.

The Ciarcia High-Resolution Sprite Graphics board enables you to maintain as many as 32 sprites (single-color figures of 8x8 or 16x16 pixels) at one time or hi-res alphanumerics, all with a solid-color backdrop.

The Sprite Graphics Board is Apple (C) slot-compatible. Since it comes with Krell's Multi-Video Board, only one monitor is needed to display both normal Apple (C) video or sprites.

The Sprite Graphics Board comes with two disks (demonstration software and the sprite editor), with full documentation. The price is $325.

The *College Board 1983 SAT Exam Preparation Series* includes 42 programs covering vocabulary, reading comprehension, word relationships, mathematics, and the test of standard written English. It is available for Apple, Atari, Commodore, Franklin, IBM and Radio Shack microcomputers for $299.95.

Krell is also the marketing agent for the Socrates Chess Corporation's *Shelby Lyman Chess Tutorial Series* for Apple, Atari, Commodore, Franklin, and IBM personal computers.

The series uses the latest and most effective methods of chess instruction programming techniques to provide truly individualized instruction. All aspects of the game are covered, from basic chess weaponry to the

O
C
P

ATARI

400 16K	$265
410 RECORDER	76
800 48K	635
810 DISK DRIVE	429
PERCOM D/D DISK DRIVE	589
825 PRINTER	575
830 MODEM	149
850 INTERFACE	159
LE STICK	30
NEWPORT PROSTICK 150	30
VERBATIM SS/DD DISKS	29
MAXELL SS/DD DISKS	35
Joysticks or Paddles	18
The Entertainer	69
The Programmer	54
The Educator	119
The Communicator	299
Caverns of Mars	29
Centipede	33
Pac Man	33
Star Raiders	33
Apple Panic	21
Canyon Climber	21
Choplifter	26
Clowns and Balloons	21
Rosen's Brigade	26
Frogger	25
Gorf	30
Picnic Paranoia	26
K-Razy Shootout	36
Escape from Vulcan's Isle	21
Pacific Coast Highway	21
Nautilus	26
Prepple	23
O'Riley's Mine	26
Threshold	30
Shamus	25
Curse of Ra	16
Firebird	30
Zork I/II	30
Visicalc	179
APX SOFTWARE	CALL
APPLE SORTWARE	CALL
TRS-80 SOFTWARE	CALL

commodore

VIC 20	$179
VIC 1515 PRINTER	335
VIC 1530 DATASSETTE	67
VIC 1540 DISK DRIVE	489
VIC 1600 VICMODEM	97
VIC 1211 SUP. EXPANDER	53
VIC 16K MEM. EXPANDER	99
VIC 8K MEM. EXPANDER	52
Scott Adams Adventures	35
Vic Avenger	23
Super Alien	23
Choplifter	32
Apple Panic	32
Omega Race	35
Shamus	29
Gorf	35
PRINTERS	
NEC 8023A-C	$479
Okidata Microline 80	329
Okidata Microline 82A	469
Epson MX 100/Graftrax	679
MONITORS	
NEC JC-1203 Color	$779
Sanyo 13" Color	399
Amdek Video 300 Amber	169
Amdek Color I	339

Optomam Consumer Products

Dept. A
P.O. Box 1038, Placerville, CA 95667

916/621-1090

TO ORDER CALL OR WRITE

Terms: Certified check, M.O., personal check (2 wks. to clear), Bank Wire (U.S. Funds). No C.O.D. Visa, M/C add 3% surcharge, inc. card # & exp. date. CA res. add 6% tax. Ship. & Handling add 3% (Min. $3) in Cont'l. U.S. (call for ship. other areas). Send $2 for catalog, $2 off coupon on next order. Inc. phone # all orders. Prices subject to change w/o notice.

COMMODORE
★ PET OWNERS ★

NEW AUTHENTIC PROGRAMS

CASINO CRAPS

- Any bet made in Vegas, now can be made at home.
- The Field Hardways-Place Bets-Come-Pass Line
- Find a winning system, without losing a dime.

8K version (1 player) $10.95
16K version (5 players) $12.95

KONNECT FOUR

- Now play this popular game against your pet.
- Excellent sound & graphics
- Real time clock
- Three levels of play
- Can fit into 8K
- Fun & Educational for all ages

ONLY $10.95

G P Microsystems
72-31 67th Place
Glendale, N.Y. 11385

Please include $1.50 shipping & handling for each program. Indicate version.

Memory Expansion Boards for the ATARI* Computer

- Completely compatible with ATARI hardware and software
- No modifications necessary (16K & 32K)
- Fully assembled and tested
- User installable
- Gold connector tabs and sockets
- One year warranty

16K Memory Board (800)-$49.95
32K Memory Board (400/800)-$89.95
48K Memory Board (400)-$169.96

We Will Meet Advertised Prices

Build Your Own Memory

16 K board - no components - $10.00
32K board - no components - $30.00
48K board - no components - $50.00

— NEW ATARI SOFTWARE —

Add $2 Shipping and handling
Visa & MasterCard Accepted
*ATARI is a trademark of Atari Inc.

Dealer Inquiries Welcome

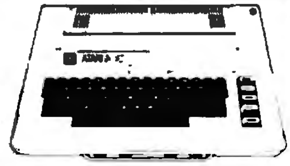
Tiny Tek, Inc.

P.O. Box 820249
Dallas, TX 75382-0249
214-373-8926

Eric Martin's

Where prices are born, not raised!

ATARI



Atari 800 48K $639.

810 Disc Drive	$439.
410 Recorder	$74.
825 Printer	$578.
850 Interface	$168.
481 Entertainer	$76.
482 Educator	$118.
483 Programmer	$54.
484 Communicator	$295.

ATARI Software

CX4104 Mailing List	$17
CXL4007 Music Composer	43
Programming 2 & 3	20
Conversational Languages	43
CX4018 Pilot	57
CX405 Pilot	97
CS8126 Microsoft Basic	65
CXL4022 Pac-Man	31
CXL4020 Centipede	31
CXL4009 Computer Chess	26
CSL4011 Star Raiders	33
CXL4012 Missile Command	26
CSL4013 Asteroids	26

Call for price of 3rd party software

ATARI SERVICE
FACTORY AUTHORIZED NETWORK

Orders shipped in 24 hours

For fast delivery, send certified or cashier checks, money orders, or direct bank wire transfers. Personal checks allow 2 to 3 weeks to clear. Prices reflect cash discount and are subject to change. Add 2% for credit card purchases. Shipping—Software $2 Minimum. Hardware—call. Foreign inquiries invited—add 15% for shipping. Ohio residents add 6.5% sales tax.

Eric Martin's

5485 Warrensville Center Road
Maple Heights, Ohio 44137
216/663-2032

Call Toll Free
1-800-482-7254
outside Ohio

Mon.-Sat. 10-6 EST

game's most subtle points. The series also offers a chance to match wits against international grandmasters.

Each instruction module interfaces with the Socrates Chess Player. The price of each module is $39.95.

Krell Software
1320 Stony Brook Road
Stony Brook, NY 11790
(516)751-5139

ZX-81 Software

RAK Electronics has announced a new line of software for the Sinclair ZX-81. Three recent releases are *Math Drill, Golf,* and *Towers of Hanoi.*

An educational program, *Math Drill* allows the user to select addition, subtraction, multiplication, or division. Written for elementary school aged children, *Math Drill* displays flashcard-style math and keeps score.

Golf allows up to four players to play 18 holes.

In *Towers of Hanoi*, the player tries to move all the discs to the last tower in the least number of moves possible.

Each program is available for $4.95. There is also a $2 (per order) shipping and handling charge. Each program requires a 16K memory expander. A free catalog of ZX-81 software is available on request.

RAK Electronics
P.O. Box 1585
Orange Park, FL 32073

Airstrike Game For Atari 400/800

English Software announces the release of *Airstrike*. Available on 16K cassette and disk, *Airstrike* provides arcade realism and demands a high level of skill from the game-player within a fast shoot-out scenario.

Game features include: superb fine-scrolling graphics; top-notch color; multiple skill levels (each denoted by a different color playfield); 100% machine program; one- or two-player mode. The suggested retail price is $39.95.

English Software Company
934 Hermosa Avenue
Suite 13, Sandcastle Plaza
Hermosa Beach, CA 90254
(213)372-3440

A 40/80 Character Expansion For The VIC

Computer World has designed an expansion to make the VIC suitable for professional applications, e.g., accounting and word processing, especially when using the Brother 8300 daisy wheel printer/typewriter, adapted to operate with the VIC-20.

When using the video cartridge, you may choose between a 25 x 40 and a 25 x 80 character mode (25 lines of 80 characters each). This enables you to use the programs written for the 2000, 3000, 4000 and 8000 CBM computers without major alterations.

The Computer World video cartridge for the VIC-20 has the following features:

– 40 or 80 column display (switchable with BASIC, without losing any program in memory).

– VIC and PET/CBM graphics.

– 32K RAM expansion (instead of 27.5K). Note: All you need is the expansion box with the 3, 8 and 16K RAM cartridges.

– Upper- and lowercase with true descenders.

– Full cursor control.

– Program editing in 40 or 80 column mode.

– No alterations needed in VIC.

– No external power supply needed.

– Fits in the Arfon or Commodore expansion box.

– Operates with all ROM cartridges, except those located in the area from A000 hex to C000 hex.

The price is $249 US.

The company was incorrectly identified in the November 1982 new product release in **COMPUTE!**. The correct information is:

Computer World
Hilvertsweg 99.
1214 JB Hilversum
Holland
Tel: 035-12633

Color Video For PET/CBM, AIM, SYM

Color Chart is a color video RAM board designed for Commodore CBM, Rockwell AIM and Synertek SYM type systems. The small 2 1/2 x 5 inch board plugs into a 2532 ROM socket. Two control wires clip to read/write signals in the system and convert the ROM socket into a 4K video RAM.

Color Chart operates in eight different modes, ranging from an alphanumeric 32 x 16 display with built-in character generator to a high resolution graphics mode with 128 x 192 pixels. Up to eight different colors are available, depending on mode. Based on the 6847 video controller, Color Chart reads information that the computer writes to video RAM and translates it into a standard RS170 type composite video output. This can directly drive most color video monitors or can be used with a RF modulator and color TV set.

Color Chart can be used to present independent color graphics displays on a PET/CBM while the main screen displays corresponding text. Business graphs and game graphics can be displayed. Color Chart provides an inexpensive full screen video display for AIM or SYM and has been used to convert these machines into remote computer terminals.

Color Chart sells for $139.95.

CGRS Microtech
P.O. Box 102
Langhorne, PA 19047
(215)757-0284

Disk Utility For The Apple

Central Point Software, Inc., has introduced The Filer, an Apple disk utility system for 35, 40, and 70 track drives. The Filer package contains a disk drive speed check, disk drive test, fast copy program (35 seconds), and a sophisticated file manager. Options include: Catalog with space on disk, Copy Files, Copy Disk, and Copy DOS, Delete, Lock and Unlock Files, and Change Booting Program (name and file type).

The suggested retail price is $19.95. Available at computer and software stores or direct from:

Central Point Software, Inc.
P.O. Box 19730-#203
Portland, OR 97219
(503)244-5782

Word Processing On Commodore VIC And 64

A full-featured, letter quality word processing package is available for the Commodore VIC and 64 computers from Quick Brown Fox.

With this package, a user can edit even with the VIC's standard 22-column width. Features include full line and global edit capabilities, text moving, boilerplating, tab and margin settings, right justification, and proportional spacing. Quick Brown Fox also automatically reformats edited text, without using the paragraph-by-paragraph method.

The cost of the package is $65.

With Quick Brown Fox software, a user can put together a workstation that includes letter-quality printing, 80-column display, and ample internal and disk storage.

Quick Brown Fox
548 Broadway, Suite 4F
New York, NY 10012
(212)925-8290

Space Game For Atari

K-Star Patrol is a game which has space ships confronting alien attack ships and an intergalactic leech. The package, with graphics, contains one ROM Cartridge and a 12-page, full-color instruction booklet with complete rules of the game. The cartridge fits Atari 400 and 800 personal computer systems.

K-Star Patrol, space game from K-Byte.

For more information on *K-Star Patrol* (Model No. ATR1002) and other computer games and products, contact:

K-Byte
Division of Kay Enterprises Co.
1705 Austin Street
Troy, MI 48084
(313)524-9878

Color Computer Expansion Unit

Maple Leaf Systems announces the Multiport, the first multiple-slot expansion unit for the Color Computer. This device allows simultaneous connection of up to four separate Color Computer compatible peripherals. For example, a disk, modem, program cartridge, printer, and clock cartridge can now be on-line at once. The computer can switch between peripherals under software control, allowing one program to access any or all of the peripherals at any time.

The Multiport is a powerful hardware circuit which allows selection of any of the four sockets with a simple POKE command. It connects directly to the expansion port of all models of the Color Computer.

The Multiport comes completely assembled and tested,

with full instructions, for $99.50.

Maple Leaf Systems
P.O. Box 2190, Station 'C'
Downsview, Ontario Canada M2N 2S9

Music And Sound For The Apple II/II+

The Soundchaser Computer Music System.

Passport Designs has added *Turbo-Traks* to the expanding library of software available for the Soundchaser Computer Music System.

The Soundchaser package, for the 48K Apple II+ with one disk drive, includes a polyphonic synthesizer, multi-track recorder, computer-aided instruction device, and realtime music transcriber. It may be purchased as a complete analog/digital system (suggested retail price, $1450) or as individual modules. Manuals are available separately for $8 each.

Turbo-Traks includes a 16-track digital recorder that simulates an analog tape deck. It also has a variable number (up to 16) of oscillators per voice, sync to tape or drum machine, and extended recording time. *Turbo-Traks* is a live performance synthesizer and 16-track recording studio all in one package.

Another recent release of Passport Designs is *Kaleido-Sound,* a realtime graphics program that synchronizes to any audio input. The four full-color kaleidoscopes change color, pattern, and location on a CRT monitor, television, or video screen as the music changes frequency and loudness. *Kaleido-Sound* (including connecting cable) has a suggested retail price of $39.95.

Passport Designs
116 North Cabrillo Hwy.
Half Moon Bay, CA 94019
(415)726-0280

EPYX Expands Line Of VIC-20 Software

EPYX has released two new action games, *Crush, Crumble and Chomp!* and *Rescue at Rigel*, for the VIC-20.

Crush, Crumble and Chomp!, a movie monster game, lets the player assume the role of any one of six hungry man-eating beasts in more than 100 possible scenarios. To satisfy his enormous appetite, the player snacks on his opponents, while doing battle with National Guard tanks, infantry, helicopters, and even a team of mad scientists. The player can wreak havoc on any one of four major cities – New York City, San Francisco, Tokyo and Washington, D.C.

Rescue at Rigel, a space adventure, takes the player into a maze-like, six-floor, 60-room complex inhabited by an alien in-

sectoid race, the Tollah. As Sudden Smith, the player has 60 minutes to find ten humans held captive inside the labyrinth and beam them up to a rescue ship.

During the mission, the player is up against hostile Tollahs, two types of armed robots, a huge amoebic slug called a plasmoid, and a dangerous six-legged cerbanth. The player must find his way through the alien complex and rescue the prisoners – all in one hour.

Crush, Crumble and Chomp! and *Rescue at Rigel* are available on cassette for the VIC-20 (16K expander required), Atari 400/800 (32K), and TRS-80 (Level II, 16K), and on disk for the Atari 400/800 (32K), Apple (48K with Applesoft in ROM), and TRS-80 (TRSDOS 32K). The suggested retail price is $29.95.

EPYX / Automated Simulations, Inc.
1043 Kiel Court
Sunnyvale, CA 94086

Futuristic Game For The Apple

Interactive Fantasies has released *Empire II: Interstellar Sharks*, the second system in its Empire Gaming Trilogy. *Interstellar Sharks* is set in a futuristic civilization at the height of its material prosperity and monopolistic bureaucracy.

The game system provides a bureaucratic jungle; the player must navigate through webs of red tape and survive the sometimes clandestine dealings of big monopolies to achieve, not wealth, but the rewards of wealth. The player's ultimate goal is buying and outfitting his own spacecraft.

A manual and a softcover novella accompany the system. *Interstellar Sharks* is available in Applesoft, 48K, DOS 3.3, and retails for $32.95.

Interactive Fantasies
P.O. Box 22222
Agoura, CA 91301
(213)706-0661

CALENDAR

March 10-12, 1983. National conference on the Use of Microcomputers in Special Education, Hartford, Connecticut. Program chairperson: Kathleen M. Hurley, Vice President of Research and Development, Developmental Learning Materials, Inc. (DLM), Allen, Texas. The conference will focus on the use of microcomputers in special education programs, and on adaptations and creation of software programs for specific exceptional needs. For further information, preregistration and housing forms, contact: John Grossi, Conference Manager, The Council for Exceptional Children, Department of Field Services, 1920 Association Drive, Reston, VA 22091.

March 14-15, 1983. Seventh annual conference of the Michigan Association for Computer Users in Learning, Dearborn, Michigan. Features sessions with Arthur Luerhman, Ramon Zamora, and David Moursund. It is expected to attract over 2,000 educators from the midwest. For information, contact: Betty VandenBosch Shaw, Coordinator of Mathematics, Flint Community Schools, 923 East Kearsley, Flint, MI 48502. (313) 762-1007.

March 17-19, 1983. Third annual Microcomputers in Education Conference – "Forward to the 3 C's: Communicating, Calculating and Computing" – directed by Paul Field, Chris Titus, Jon Titus, and David Larsen. Arizona State University, Tempe. A variety of workshops, demonstrations, and presentations is designed for anyone interested in innovative microcomputer applications in education. For information and registration materials, contact: Marilyn Sue Ford, B-47 Payne Hall, College of Education, Arizona State University, Tempe, AZ 85287. (602)965-3322 or (602)965-7363.

March 21-24, 1983. Workshop: Personal Microcomputer Interfacing and Scientific Instrumentation Automation. $595. The workshop is hands-on, with the participant designing and testing concepts with the actual hardware. For more information, call or write Dr. Linda Leffel, C.E.C., Virginia Tech, Blacksburg, VA 24061. (703)961-4848.

April 28-30, 1983. Ed • Com/Spring '83, national computer conference and exposition for educators of all levels. Washington, DC. Nationally recognized educators to address, evaluate, and analyze the developments of computers in education in more than 300 session hours featuring demonstrations, seminars, hands-on sessions, panels and MicroCourses. There will be exhibits of hardware, software and publications. For information contact: Carol Houts, Judeo Computer Expos, Inc., 2629 North Scottsdale Road, Suite 201, Scottsdale, AZ 85257. (602) 990-1715 or (800) 528-2355 outside Arizona.

COMPUTE! *welcomes notices of upcoming events and requests that the sponsors send a short description, their name and phone number, and an address to which interested readers may write for further information.*

Please send notices at least three months before the date of the event to: Calendar, P.O. Box 5406, Greensboro, NC 27403.

New Product releases are selected from submissions for reasons of timeliness, available space, and general interest to our readers. We regret that we are unable to select all new product submissions for publication. Readers should be aware that we present here some edited version of material submitted by vendors and are unable to vouch for its accuracy at time of publication.

PRODUCT MART

Advertisers Index

COMPUTE!

For Fastest Service, Call Our **Toll-Free** US Order Line **800-334-0868** **In NC call 919-275-9809**

My Computer Is:

☐ PET ☐ Apple ☐ Atari ☐ OSI ☐ VIC-20 ☐ TI 99/4A ☐ Sinclair ZX-81
☐ Radio Shack Color Computer ☐ Other __________ ☐ Don't yet have one...

☐ $20.00 One Year US Subscription
☐ $36.00 Two Year US Subscription
☐ $54.00 Three Year US Subscription

(Readers outside of the US, please see our foreign readers subscription card or inquire for rates).

Name ______________________________

Address ______________________________

City __________ State __________ Zip __________

☐ Payment Enclosed ☐ VISA ☐ Bill me
☐ MasterCard ☐ American Express

Account No. ______________ Expires ____/____

331101

COMPUTE!

Foreign Readers

Subscription rates outside the US:

☐ $25.00 Canada
☐ $38.00 Europe, Australia/Air Delivery
☐ $48.00 Middle East/Air Delivery
☐ $68.00 Elsewhere/Air Delivery
☐ $25.00 International Surface Mail (lengthy, unreliable delivery)

Name ______________________________

Address ______________________________

City __________ Postal Code __________

Country ______________________________

Payment must accompany this card.
Payment in US Funds drawn on a US Bank; International Money Order; or charge card: ☐ VISA ☐ MasterCard ☐ American Express

Account No. ______________ Expires ____/____

331101

COMPUTE! Books

For Fastest Service Call Our **TOLL FREE** US Order Line **800-334-0868** **In NC call 919-275-9809**

Quan.	Title	Price		S/H	Total
____	**The Beginner's Guide to Buying A Personal Computer**	$3.95	+	$1.00*	____
____	**COMPUTE!'s First Book of Atari**	12.95	+	2.00*	____
____	**Inside Atari DOS**	19.95	+	2.00*	____
____	**COMPUTE!'s First Book of PET/CBM**	12.95	+	2.00*	____
____	**Programming the PET/CBM**	24.95	+	3.00**	____
____	**Every Kid's First Book of Robots and Computers**	4.95	+	1.00*	____
____	**COMPUTE!'s First Book of VIC**	12.95	+	2.00*	____
____	**COMPUTE!'s Second Book of Atari**	12.95	+	2.00*	____

For air mail outside US: *$5.00 / **$10.00

All orders must be prepaid (money order, check, or charge). All payments must be in US funds. NC residents add 4% sales tax.

☐ Payment enclosed

Please charge my: ☐ VISA ☐ MC ☐ Am. Express

Acc't No. ______________

Expires ____/____

Name ______________________________

Address ______________________________

City __________ State __________ Zip __________

Country ______________________________

Allow 4-5 weeks for delivery.

The Editor's Feedback:

Computer: ☐ Pet ☐ Apple ☐ Atari ☐ OSI ☐ VIC-20 ☐ TI 99/4A ☐ Sinclair ZX-81
☐ Radio Shack Color Computer ☐ Other __________ ☐ Don't yet have one...

Are you a **COMPUTE!** Subscriber? ☐ Yes ☐ No I would like to see:

☐ More ☐ Fewer Specific applications programs.
☐ More ☐ Fewer BASIC programs.
☐ More ☐ Fewer Machine language programs.
☐ More ☐ Fewer Tutorials.
☐ More ☐ Fewer Educational articles.
☐ More ☐ Fewer Detailed explanations of programs.
☐ More ☐ Fewer Games.
☐ More ☐ Fewer Reviews of game software.
☐ More ☐ Fewer Reviews of business software.
☐ More ☐ Fewer Reviews of educational software.
☐ More ☐ Fewer Reviews of hardware.

What do you like best about **COMPUTE!**?

What do you like least?

1 2 3 4 5 6 7 8 9 10 11 12

Place
Stamp
Here

COMPUTE! Magazine
P.O. Box 914
Farmingdale, NY 11737

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 2312 GREENSBORO, NC

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE! Magazine
P.O. Box 914
Farmingdale, NY 11737

Place
Stamp
Here

COMPUTE! Magazine
Post Office Box 5406
Greensboro, NC 27403

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO 2312 GREENSBORO, NC

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE! Books
Post Office Box 5406
Greensboro, NC 27403

Introduce A Friend To COMPUTE!
Save $10.00 Off The Newsstand Price

One year, 12 issue subscriptions are $20.00 in the U.S., $25.00 (U.S. funds) in Canada.

PLEASE PRINT.

Please charge my

☐ VISA ☐ MasterCard ☐ American Express

Acct No. ____________________

Exp. ______ ______

YOUR NAME ____________________

ADDRESS ____________________

CITY ____________________

STATE ____________ ZIP ____________

$__________ payment enclosed ☐ Bill me later

Please enter my ☐ RENEWAL ☐ NEW SUBSCRIPTION at the same time

GIFT TO ____________________

ADDRESS ____________________

CITY ____________________

STATE ____________ ZIP ____________

☐ Renewal ☐ New subscription

SIGN CARD ____________________

GIFT TO ____________________

ADDRESS ____________________

CITY ____________________

STATE ____________ ZIP ____________

☐ Renewal ☐ New subscription

SIGN CARD ____________________

631101

Introduce A Friend To COMPUTE!
Save $10.00 Off The Newsstand Price

One year, 12 issue subscriptions are $20.00 in the U.S., $25.00 (U.S. funds) in Canada.

PLEASE PRINT.

Please charge my

☐ VISA ☐ MasterCard ☐ American Express

Acct No. ____________________

Exp. ______ ______

YOUR NAME ____________________

ADDRESS ____________________

CITY ____________________

STATE ____________ ZIP ____________

$__________ payment enclosed ☐ Bill me later

Please enter my ☐ RENEWAL ☐ NEW SUBSCRIPTION at the same time

GIFT TO ____________________

ADDRESS ____________________

CITY ____________________

STATE ____________ ZIP ____________

☐ Renewal ☐ New subscription

SIGN CARD ____________________

GIFT TO ____________________

ADDRESS ____________________

CITY ____________________

STATE ____________ ZIP ____________

☐ Renewal ☐ New subscription

SIGN CARD ____________________

631101

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 2312 GREENSBORO, NC

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE!
P.O. Box 914
Farmingdale, NY 11737

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 2312 GREENSBORO, NC

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE!
P.O. Box 914
Farmingdale, NY 11737

COMPUTE!'s FREE Reader Information Service

Use these cards to request FREE information about the products advertised in this issue. Clearly print or type your full name and address. Only one card should be used per person. Circle the numbers that correspond to the key number appearing in the advertisers index.

Send in the card and the advertisers will receive your inquiry. Although every effort is made to insure that only advertisers wishing to provide product information have reader service numbers, **COMPUTE!** cannot be responsible if advertisers do not provide literature to readers.

Please use these cards *only* for subscribing or for requesting product information. Editorial and customer service inquiries should be addressed to: **COMPUTE!**, P.O. Box 5406, Greensboro, NC 27403. Check the expiration date on the card to insure proper handling.

COMPUTE!

101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122
123	124	125	126	127	128	129	130	131	132	133
134	135	136	137	138	139	140	141	142	143	144
145	146	147	148	149	150	151	152	153	154	155
156	157	158	159	160	161	162	163	164	165	166
167	168	169	170	171	172	173	174	175	176	177
178	179	180	181	182	183	184	185	186	187	188
189	190	191	192	193	194	195	196	197	198	199
200	201	202	203	204	205	206	207	208	209	210
211	212	213	214	215	216	217	218	219	220	221
222	223	224	225	226	227	228	229	230	231	232
233	234	235	236	237	238	239	240	241	242	243
244	245	246	247	248	249	250	251	252	253	254
255	256	257	258	259	260	261	262	263	264	265
266	267	268	269	270	271	272	273	274	275	276
277	278	279	280	281	282	283	284	285	286	287
288	289	290	291	292	293	294	295	296	297	298
299	300	301	302	303	304	305	306	307	308	309
310	311	312	313	314	315	316	317	318	319	320
321	322	323	324	325	326	327	328	329	330	331
332	333	334	335	336	337	338	339	340	341	342
343	344	345	346	347	348	349	350			

Circle 101 for a one year new subscription to **COMPUTE!**: 12 monthly issues for $20

Please print or type your full name and address. Limit one card per person.

Name ____________________

Address ____________________

City ____________________

State/Province ____________ Zip ________

Country ____________________

Please include zip code. Expiration 3/31/83 C0183

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 27346 PHILADELPHIA, PA

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE!
P.O. Box 11747
Philadelphia, PA 19101

COMPUTE!

101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122
123	124	125	126	127	128	129	130	131	132	133
134	135	136	137	138	139	140	141	142	143	144
145	146	147	148	149	150	151	152	153	154	155
156	157	158	159	160	161	162	163	164	165	166
167	168	169	170	171	172	173	174	175	176	177
178	179	180	181	182	183	184	185	186	187	188
189	190	191	192	193	194	195	196	197	198	199
200	201	202	203	204	205	206	207	208	209	210
211	212	213	214	215	216	217	218	219	220	221
222	223	224	225	226	227	228	229	230	231	232
233	234	235	236	237	238	239	240	241	242	243
244	245	246	247	248	249	250	251	252	253	254
255	256	257	258	259	260	261	262	263	264	265
266	267	268	269	270	271	272	273	274	275	276
277	278	279	280	281	282	283	284	285	286	287
288	289	290	291	292	293	294	295	296	297	298
299	300	301	302	303	304	305	306	307	308	309
310	311	312	313	314	315	316	317	318	319	320
321	322	323	324	325	326	327	328	329	330	331
332	333	334	335	336	337	338	339	340	341	342
343	344	345	346	347	348	349	350			

Circle 101 for a one year new subscription to **COMPUTE!**: 12 monthly issues for $20.

Please print or type your full name and address.
Limit one card per person.

Name ______________________

Address ______________________

City ______________________

State/Province ______________ Zip ________

Country ______________________

Please include zip code. Expiration 3/31/83 C0183

COMPUTE!

101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122
123	124	125	126	127	128	129	130	131	132	133
134	135	136	137	138	139	140	141	142	143	144
145	146	147	148	149	150	151	152	153	154	155
156	157	158	159	160	161	162	163	164	165	166
167	168	169	170	171	172	173	174	175	176	177
178	179	180	181	182	183	184	185	186	187	188
189	190	191	192	193	194	195	196	197	198	199
200	201	202	203	204	205	206	207	208	209	210
211	212	213	214	215	216	217	218	219	220	221
222	223	224	225	226	227	228	229	230	231	232
233	234	235	236	237	238	239	240	241	242	243
244	245	246	247	248	249	250	251	252	253	254
255	256	257	258	259	260	261	262	263	264	265
266	267	268	269	270	271	272	273	274	275	276
277	278	279	280	281	282	283	284	285	286	287
288	289	290	291	292	293	294	295	296	297	298
299	300	301	302	303	304	305	306	307	308	309
310	311	312	313	314	315	316	317	318	319	320
321	322	323	324	325	326	327	328	329	330	331
332	333	334	335	336	337	338	339	340	341	342
343	344	345	346	347	348	349	350			

Circle 101 for a one year new subscription to **COMPUTE!**: 12 monthly issues for $20.

Please print or type your full name and address.
Limit one card per person.

Name ______________________

Address ______________________

City ______________________

State/Province ______________ Zip ________

Country ______________________

Please include zip code. Expiration 3/31/83 C0183

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 27346 PHILADELPHIA, PA

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE!
P.O. Box 11747
Philadelphia, PA 19101

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 27346 PHILADELPHIA, PA

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE!
P.O. Box 11747
Philadelphia, PA 19101

WHEN WE ANNOUNCED THE COMMODORE 64 FOR $595, OUR COMPETITORS SAID WE COULDN'T DO IT.

THAT'S BECAUSE THEY COULDN'T DO IT.

The reason is that, unlike our competitors, we make our own IC chips. *Plus* all the parts of the computer they go into.

So Commodore can get more advanced computers to market sooner than anybody else. And we can get them there for a lot less money.

WHAT PRICE POWER?

For your $595,* the Commodore 64™ gives you a built-in user memory of 64K. This is hundreds of dollars less than computers of comparable power.

Lest you think that the Commodore 64 is some stripped-down loss leader, a look at its available peripherals and interfaces will quickly convince you otherwise.

SOFTWARE THAT WORKS HARD.

The supply of software for the Commodore 64 will be extensive. And with the optional plug-in Z80 microprocessor, the Commodore 64 can accommodate the enormous amount of software available in CP/M.®

Add in the number of programs available in BASIC and you'll find that there are virtually no applications, from word processing to spreadsheets, that the Commodore 64 can't handle with the greatest of ease.

PERIPHERALS WITH VISION.

The Commodore 64 interfaces with all the peripherals you could want for total personal computing: disk drives, printers and a telephone modem that's about $100, including a free hour's access to some of the more popular computer information services. Including Commodore's own Information Network for users.

RUN YOUR BUSINESS BY DAY. SAVE THE EARTH BY NIGHT.

At the end of a business day, the Commodore 64 can go into your briefcase and ride home with you for an evening's fun and games.

Because of its superior video quality (320x200 pixel resolution, 16 available colors and 3D Sprite graphics), the Commodore 64 surpasses the best of the video game machines on the market. Yet, because it's such a powerful computer, it allows you to invent game programs that a game machine will never be able to play; as well as enjoy Commodore's own video game cartridges.

ATTACK, DECAY, SUSTAIN, RELEASE.

If you're a musicologist, you already know what an ADSR (attack, decay, sustain, release) envelope is. If you're not, you can learn this and much more about music with the Commodore 64's music synthesizing features.

It's a full-scale compositional tool. Besides a programmable ADSR envelope generator, it has 3 voices (each with a 9-octave range) and 4 waveforms for truly sophisticated composition and playback—through your home audio system, if you wish. It has sound quality you'll find only on separate, music-only synthesizers. And graphics and storage ability you won't find on any separate synthesizer.

DON'T WAIT.

The predictable effect of advanced technology is that it produces less expensive, more capable products the longer you wait.

If you've been waiting for this to happen to personal computers, your wait is over.

See the Commodore 64 soon at your local Commodore Computer dealer and compare it with the best the competition has to offer.

You can bet that's what the competition will be doing.

Commodore Business Machines
Personal Systems Division
P.O. Box 500, Conshohocken, Pennsylvania 19428

Please send me more information on the Commodore 64™

Name________ Title________

Company________

Address________

City________ State________

Zip________ Phone________

commodore COMPUTER

CPT-1

Manufacturer's Suggested Retail Price July 1, 1982. Disk drives and printers are not included in prices. The 64's price may change without notice.

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