A VIC-20 Intelligent Video Disk System

COMPUTE!

$2.50
June
1982
Issue 25
Vol. 4, No. 6
63379 £1.85

The Journal For Progressive Computing™

Income Property Report

A Self-Modifying Atari Player/Missile Graphics Utility

An Apple PASCAL Lister

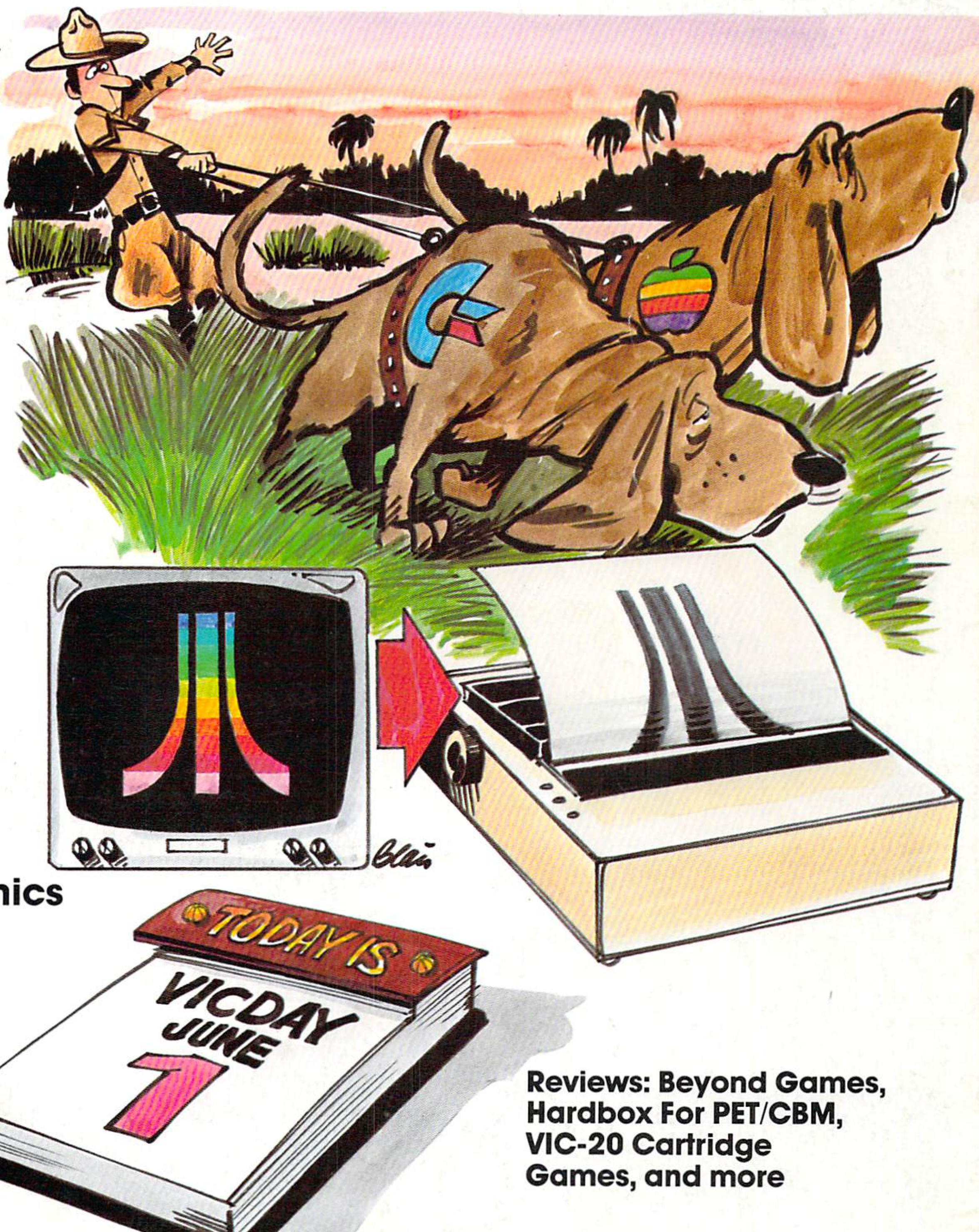
VIC/PET BASIC Program Transfers

Search For PET/CBM And Apple II Plus

How To Run 96K Programs On The SuperPet

Copy Atari Graphics To Your Printer

A Perpetual Calendar For The Vic-20



Reviews: Beyond Games, Hardbox For PET/CBM, VIC-20 Cartridge Games, and more

0 74470 63379 06

A GRAPHIC MASTERPIECE AT YOUR FINGER TIPS.

Datasoft™ introduces 4 new graphic programs to bring out the artist in you. Because we believe computers should exhibit, not inhibit your creativity, we've created Micro-Painter,™ Graphics Generator,™ Graphic-Master™ and Color Print.™

For openers, Micro-Painter will transform your computer screen into a canvas for your creativity. Whether the image is one you've imagined or one of 8 we supply, you have the artistic freedom to change line, color and composition. By using a joystick and special zoom feature, choose from a palette of color and patterns and apply any of 16 vibrant colors in 8 luminosities to even the smallest area. And once you think you've reached the limit of creative possibilities, think again. Micro-Painter can be used as a business tool as well. It can generate bar charts, graphs, illustrations and other visual aids. And best of all it works with Atari® and Apple II® (48K) for $34.95.

But Micro-Painter is just the beginning. There's more potential when you add the Graphics Generator to your graphic tool box. It lets you become your own typesetter by offering a variety of type from Serif to Sans Serif, Old English and Roman. You can change keyboard character sets into graphic symbols or create original character sets with fonts of your own design. It even allows you to develop spaceships or exotic aliens. And when you're ready to come back down to earth, this program can label charts, graphs and schematics. The Graphics Generator is used with Atari (32K) for $24.95.

With Graphic Master you'll master more of the visual arts. You can apply special effects such as rotating or skewing images. It employs windows of color that work like a paint brush. Choose between 8 differently sized brushes to underlay or overlay color or combine images on screen to be saved on the diskette and used with other programs later. Graphic Master is compatible with Atari (48K) and retails for $39.95.

And rounding out Datasoft's graphic design program is Color Print. Original compositions will emerge in B&W, 2, 3 or 4-color high resolution tones from an Epson® MX-80, as if rolling hot off the press. Color Print is applicable to the Atari (40K) and sells for $39.95.

So for the full picture, contact Datasoft or your nearest software dealer. And explore your full potential through the aesthetic software from Datasoft Inc.

Datasoft Inc.
COMPUTER SOFTWARE
19519 Business Center Drive
Northridge, CA 91324 • (213) 701-5161

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.

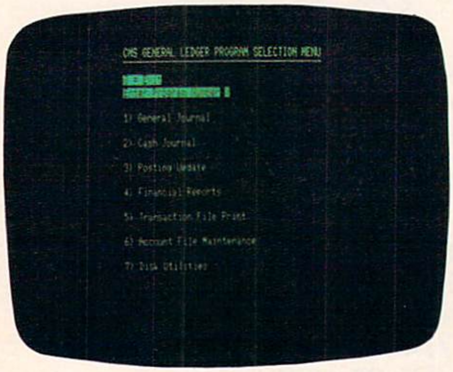
Professional Software Inc.
51 Fremont Street
Needham, MA 02194
(617) 444-5224
TELEX: 95 1579

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

CMS Softwa
General Accou

For the Commodore 8000

Now there is an interactive General Accounting System designed especially for the first time user. The user is led through each function by highlighted prompts explaining the required input at each point. And all input requests are prompted with complete verification. Plus, a step-by-step instruction manual guides the user through the program. The user prompts and the detailed manual make it virtually impossible for the first time user to get lost or to accidentally crash.

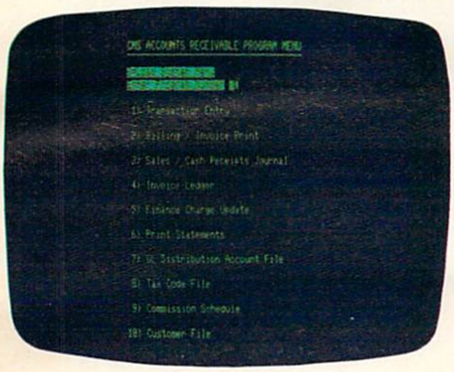


General Ledger

Cash Disbursements and Cash Receipts Journal as well as a General Journal. Maintains account balances for Present Month, Quarter to Date and Year to Date. Accepts postings from Accounts Receivable, Accounts Payable, Payroll and other programs. Also provides user customized financial statements. Up to 1000 accounts and up to 99 departments.

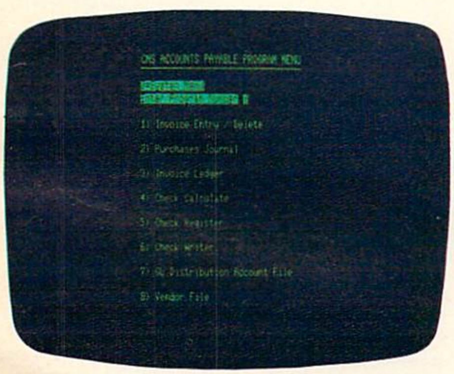
re Systems
nting System II

Series Computer System



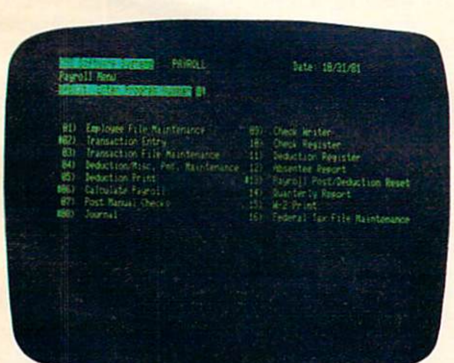
Accounts Receivable

Invoices and Monthly Statements as well as Credit and Debit Memos. Accomodates full or partial payments. User sets finance charge rate and period. User also sets invoice aging reports and aging breaks. Invoices may be distributed among nine different General Ledger accounts with automatic updating to the General Ledger. During invoice data entry the invoice is displayed on the screen and typed exactly as if in a typewriter. File capacity allows 1800 customers and invoices. Five hundred customers allows room for 1300 invoices.



Accounts Payable

Accounts Payable checks with full voucher detail. Detailed check register. Automatic application of Credit Memos. User sets invoice aging reports and aging breaks. Invoices may be distributed among nine different General Ledger accounts with automatic updating to the General Ledger. File capacity allows 2500 vendors and invoices.



Payroll

Payroll checks with full deductions and pay detail. Pays regular, overtime, holiday and piece work hours. Accommodates Weekly, Bi-Weekly, Semi-Monthly and Monthly employees. Up to eight miscellaneous deductions or payments per employee. Prints Payroll Journal, Payroll Check Register and an Absentee Report as well as 941 information and W2 forms. Maintains Monthly, Quarterly and Yearly totals for up to 350 employees. Automatic General Ledger updating.

See your nearest Commodore dealer for a demonstration.

CMS Software Systems

2204 Camp David Mesquite, TX. 75149 214 285-3581

Turn your Apple into the world's most versatile personal computer.

The SoftCard™ Solution. SoftCard turns your Apple into two computers. A Z-80 and a 6502. By adding a Z-80 microprocessor and CP/M to your Apple, SoftCard turns your Apple into a CP/M based machine. That means you can access the single largest body of microcomputer software in existence. Two computers in one. And, the advantages of both.

Plug and go. The SoftCard system starts with a Z-80 based circuit card. Just plug it into any slot (except 0) of your Apple. No modifications required. SoftCard supports most of your Apple peripherals, and, in 6502-mode, your Apple is still your Apple.

CP/M for your Apple. You get CP/M on disk with the SoftCard package. It's a powerful and simple-to-use operating system. It supports more software than any other microcomputer operating system. And that's the key to the versatility of the SoftCard/Apple.

BASIC included. A powerful tool, BASIC-80 is included in the SoftCard package. Running under CP/M, ANSI Standard BASIC-80 is the most powerful microcomputer BASIC available. It includes extensive disk I/O statements, error trapping, integer variables, 16-digit precision, extensive EDIT commands and string functions, high and low-res Apple graphics, PRINT USING, CHAIN and COMMON, plus many additional commands. And, it's a BASIC you can compile with Microsoft's BASIC Compiler.

More languages. With SoftCard and CP/M, you can add Microsoft's ANSI Standard COBOL, and FORTRAN, or Basic Compiler and Assembly Language Development System. All, more powerful tools for your Apple.

Seeing is believing. See the SoftCard in operation at your Microsoft or Apple dealer. We think you'll agree that the SoftCard turns your Apple into the world's most versatile personal computer.

Complete information? It's at your dealer's now. Or, we'll send it to you and include a dealer list. Write us. Call us.

SoftCard is a trademark of Microsoft. Apple II and Apple II Plus are registered trademarks of Apple Computer. Z-80 is a registered trademark of Zilog, Inc. CP/M is a registered trademark of Digital Research, Inc.

MICROSOFT
CONSUMER PRODUCTS

A Division of Microsoft Inc.
10700 Northup Way • Bellevue, WA 98004

Table of Contents

June, 1982 Vol. 4, No. 6

AP = Apple, **AT** = Atari, **P** = PET/CBM, **V** = VIC-20, **O** = OSI, **C** = Radio Shack Color Computer, * = 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, 515 Abbott Drive, Broomall, PA 19008. Controlled circulation postage paid at Greensboro, NC 27403 and additional mailing offices. ISSN 0194-357X.

The Editor's notes...

Robert C. Lock
Publisher/Editor

Commodore Announces A New "Generation" Of PET/CBM Computers With CP/M Compatibility

Between various press releases, the Hanover Fair in Germany, and a few of Dr. Chip's old friends, we've pieced together the major components of Commodore's new series of personal computers. There are three primary machines:

"PET II" has the capability of full color (you provide the monitor). The 40 column computer comes with 128K. Its microprocessor is the new 6509. Suggested retail, while unannounced, is thought to be less than $1,000.

"CBM II" is an 80 column unit with integral monitor, 128K RAM, and built-in dual floppy disk drives with 340K (total) of storage capacity. Again, this computer uses Commodore's new 6509. The suggested retail is unknown at press time, but the unit will certainly be targeted for a middle point between the PET II and the high end computer described below.

Both units are expected to have IEEE-488 and RS-232 ports. Both are said to be capable of accepting an add-on Z80 chip *or* a 16 bit 8088, thus opening up the world of CP/M software to the Commodore units, just as the Small Systems Engineering Hardbox has done for the current Commodore units.

The third unit appears to be directly targeted at the IBM-type personal computer. This *multiprocessor* computer uses both the 6509 and an 8088 16-bit microprocessor. The 80 column computer has attached monitor, 256K RAM, and dual disk drives with a capacity of 680K bytes. This unit, rumor has it, will be priced below $2500.

All three computers will be displayed at the National Computer conference in Houston in June. They were first shown at the Hanover Fair in Germany in mid-May.

COMPUTE! Grows On

We are quite pleased to announce the addition of Mr. Tom Halfhill to our Editorial Staff. Tom, formerly an Associate Editor with Cleveland Magazine, is a journalist by training and an Atari 800 owner by avocation. He'll add needed support to our staff as Features Editor, and will greatly help us in bringing you information about new products such as those from Commodore mentioned above.

A Note To Readers And Advertisers

We've had several complaints recently from readers having problems with a very small number of mail order advertisers. While we have yet to find any evidence of fraud, we have found some clear problems in terms of delivery, etc. We offer the following suggestions to our readers, suggestions you advertisers should be aware of:

1. When you're buying more expensive products, especially hardware, know who you're doing business with. If you can't get references, a contact name, etc., be careful. If your check gets cashed, and your order delayed for some reason, you'll have just as much trouble tracking it down. Get your names and contacts first, don't just send a check blind.

2. If you have trouble getting products you've ordered, and feel you're getting improper service or treatment from the advertiser, write them a letter, certified mail, and detail your problems and history with them. Send a copy to the Advertiser Complaint Department at **COMPUTE!**, and note on your letter to the advertiser that you've done so. Request that the advertiser respond to you by phone or mail in a timely fashion.

3. Keep copies of all of your correspondence with a company, including the name of whoever you've talked to there. You might need it later.

4. Our screening of advertisers has been significantly expanded because of these recent questions. Our screening however, is not absolutely fail-safe. We cannot, for example, screen in advance the customer service philosophies of new advertisers. Thus we make the above suggestions.

To our mail order advertisers: We are well aware that most of you have prompt and courteous customer service policies. We receive more letters of praise for our advertisers than we do of complaint. Our goal is to help **COMPUTE!** readers receive the kind of service they should be able to expect from **COMPUTE!** advertisers. We appreciate your cooperation in our recently expanded screening and complaint resolution efforts. ©

ATARI OWNERS, WE'VE OPENED UP A WHOLE NEW WORLD FOR YOU.

At SSI, we established our reputation in quality computer games by producing some of the finest strategy simulations for the Apple® and TRS-80.®

Well, Atari owners, this is your lucky day. Because we've just introduced our first game for the Atari® 800 computer: THE SHATTERED ALLIANCE.™

With it, we've opened up for you SSI's world of fun and excitement in a big way — by literally creating an entire planet called *Osgorth* to serve as your personal playground!

OSGORTH! It is a wonder world of fantasy and adventure populated with creatures as varied as they are enchanted: dwarves, unicorns, Valkyries, centaurs, lizardmen and zorgs (whatever they are!)... just to name a few. They are preassembled into different armies, categorized into five classes of infantry and three of cavalry. They are meticulously rated for armor, speed, strength, and morale. Their weapons may be the bow, javelin, spears, pikes, cavalry lance, swords, axes, clubs, and rocks. They can even call on magic spells to enhance their speed, attacking or defensive power, or courage.

THE SHATTERED ALLIANCE is much more than a sophisticated strategy game. It is a rich tale of magic, rivalry among realms, and war among exotic beings. Best of all, these fantasy soldiers are *yours* to command as you roam Osgorth in search of glory and conquest.

As an added bonus for the Ancient-battles enthusiasts, we have included scenarios such as *Carthage vs. Alexander the Great.*

THE COMPUTER PROGRAM is a veritable wizard. With color Hi-Res graphics, it paints a beautiful Osgorthian landscape that is your battleground. It keeps track of all details so you can concentrate solely on strategy planning while its *Rapidfire* movement system makes for smooth and fast action. It even provides a powerful and intelligent computer opponent for solitaire play.

All you need to play this game is a 48K Atari 800 with mini floppy disc drive. For only $39.95, you can own this fantastic piece of real estate. So hurry on down to your local game/computer store and get THE SHATTERED ALLIANCE today!

To order with your VISA or MC, call toll free 800-227-1617, ext. 335 (800-772-3545, ext. 335 in California). To order by mail, send your check to Strategic Simulations Inc, 465 Fairchild Drive, Suite 108, Mountain View, CA 94043. All our games carry a 14-day money back guarantee.

Also available for 48K Apple II with ROM for $59.95.

SSI
STRATEGIC SIMULATIONS INC

Atari is a registered trademark of Atari, Inc.

As part of our demanding standards of excellence, we use maxell floppy discs.

WE'RE LOOKING FOR A FEW GOOD DESIGNERS. We're always on the lookout for high-quality games from independent designers. If you have game programs that you'd like SSI to publish, give us a call at (415) 964-1353 or write to SSI, 465 Fairchild Drive, Suite 108, Mountain View, CA 94043.

Apple is a registered trademark of Apple Computer Inc.

TRS-80 is a registered trademark of the Tandy Corporation.

Staff of COMPUTE!

Robert C. Lock, Publisher/Editor
Kathleen Martinek, Managing Editor
Richard Mansfield, Assistant Editor
Tom R. Halfhill, Features Editor
Alice S. Wolfe, Publisher's Assistant
Georgia Papadopoulos, Art Director/ Production Manager
Harry Blair, Advertising Sales Manager
Terry Cash, Typesetting/Production Assistant
De Potter, Production Assistant
Dai Rees, Production Assistant
Kate Taylor, Production Assistant
Charles Brannon, Editorial Assistant
Vicki Jennings, Administrative Assistant
Sonja Whitesell, Dealer Coordinator
Bonnie Valentino, Accounting Coordinator
Fran Lyons, Circulation Coordinator
Christine Gordon, Customer Service

Associate Editors

Jim Butterfield, Toronto, Canada
Harvey Herman, Greensboro, NC

Contributing Editors

Marvin DeJong, Dept. of Mathematics-Physics
The School of the Ozarks
Pt. Lookout, MO 65726
Fred D'Ignazio, c/o COMPUTE!
P.O. Box 5406, Greensboro, NC 27403
David Thornburg, P.O. Box 1317
Los Altos, CA 94022
Bill Wilkinson, Optimized Systems Software
10379-C Lansdale, Ave.
Cupertino, CA 95014
Gene Zumchak, 1700 Niagara St.
Buffalo, NY 14207

Subscription Information

COMPUTE! Circulation Dept.
515 Abbott Drive
Broomall, PA 19008 USA

TOLL FREE
Subscription Order Line
800-345-8112
In PA 800-662-2444

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

Small System Services, Inc. publishes:

COMPUTE!
The Journal For Progressive Computing

COMPUTE! Books

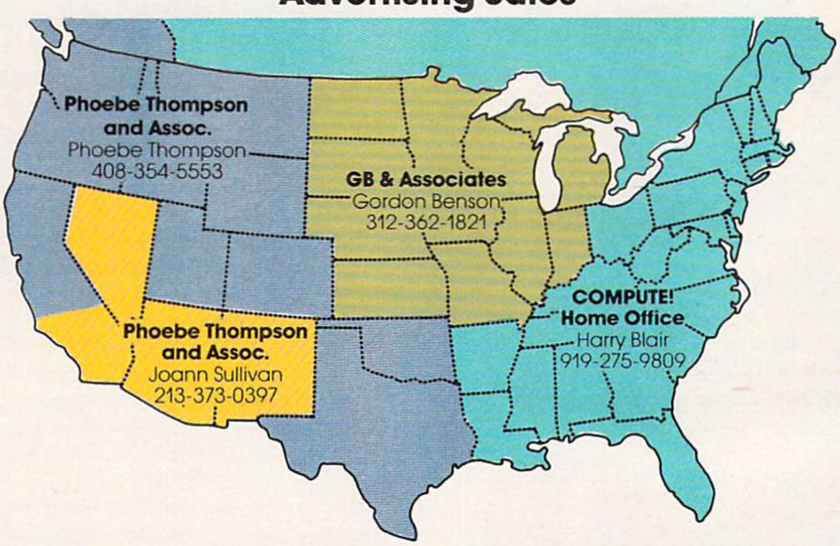
Corporate office:
625 Fulton Street,
Greensboro, NC 27403 USA

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

Telephone: 919-275-9809

Robert C. Lock, President
W. Jerry Day, Vice-President, Comptroller
Kathleen Martinek, Assistant To The President
Ellen Day, Accounting Manager
Carol Holmquist Lock, Research Assistant

Advertising Sales



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

Phoebe Thompson and Associates
P.O. Box 7000-462
Redondo Beach, CA 90277
JOANN SULLIVAN

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

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

In British Columbia, Canada call 408-354-5553.

Elsewhere in Canada or outside North America call 919-275-9809.

In Alaska or Hawaii call 408-354-5553.

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

Coming In The July Issue Of COMPUTE!

GOLD RUSH – an action game for Atari, VIC, PET/CBM, and Apple

CORRECTO – a disk file manager for PET/CBM, and Apple

Recursive BASIC subroutines

Super QuadraPET

Apple DOS Changer

ADD POWER™ TO YOUR COMMODORE COMPUTER

$89.95

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

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

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

Professional Software Inc.
51 Fremont Street
Needham, MA 02194
Tel: (617) 444-5224 Telex #951579

TM POWER is a Registered Trademark of Professional Software, Inc.
All specifications subject to change without notice.

Ask The Readers

Robert Lock, Richard Mansfield
And Readers

If you have any questions (or answers to questions printed below) please write to: Ask The Readers, **COMPUTE!** Magazine, P.O. Box 5406, Greensboro, NC 27403.

Questions

I have a question. Sometimes, after I type in a long program and run it a few times, my keyboard locks up (after you press RETURN, you can't do anything else). Is there any way I can unlock it – besides powering down? Oh, I have an Atari 800. Jon Chow

This "lock-up" is caused by a bug in the BASIC cartridge. It can occur when editing or deleting long program lines. There is no way to "un-crash" other than turning the power off and back on. It's best to save programs often, and avoid using very long program lines.

I picked up my first **COMPUTE!** *Magazine the other day* [February, 1982] *and found a fascinating article by Mike Smith, your whizz Canadian Apple contributor. My interest in named GOSUB with variable passing stems from a baptism in ALGOL when I was studying (law!) at the University of Virginia in 1969.*

I note that he is planning on being able to pass integer parameters soon. What I am looking forward to being able to do is to be able to use Boolean procedure calls, such as:

IF NILGRAVITY THEN JUMP

What about it, Mike? Sean Overend

Has anyone started a VIC-20 user club anywhere and, if so, how can I get in touch with them? If not, why don't we get one started?

Fred S. Dart
P.O. Box 525
Salem, UT 84653

Does anyone have a routine in BASIC [for PET] *so that, with the auto line feed on in the Epson, my printer will only print single line feeds, instead of double line feeds? I want to do it in the programs, instead of using the dip switch inside the Epson. Wordpro does it in machine language, so there must be a routine I could use in BASIC.* Hank Roth

---

The Atari has preset tabs on the video screen. How can I move or delete them? M. Sean Kilpatrick

There are three tab-related control keys: TAB, SET TAB (SHIFT-TAB), and CLEAR TAB (CTRL-TAB). To clear a tab, move the cursor to the tab stop and press CTRL-CLR (on TAB key). To set a tab at the cursor position, press SHIFT-TAB. Setting and clearing tabs from within a program involves PRINTing a series of TAB commands. PRINT"[TAB][TAB CLR][TAB][TAB CLR]" would clear the first two tab stops. To clear all tabs, use a series of [TAB][TAB CLR]'s. Then just PRINT [TAB SET]'s where you want to set tabs.

Answers

[In **COMPUTE!**, April, 1982, #23, pg. 181, reviewer Eric Giguere challenged anyone to beat his eight-second score on Vixel's "Fire" game.] *We have a VIC "20" and Feb., 6, 1982, I put out the fire in seven seconds, so I have the record. Will be waiting to hear from you.* Mary Payne

---

[In this column in the March, 1982, issue R. D. Young asked some questions about the **COMPUTE!** program "Keyword" (October, 1981) for PET/CBM. Liz Deal replies:] *Yes, indeed, there is a 15-character limit on file names. It can get as short as 12 bytes if you use replace (@) command and drive #, as in .S "@1:PROG",08,XXXX,XXXX. The conspiracy is coded at $FF41 in Upgrade PET.*

Keyword is IRQ driven, hence, any time you press shifted key, a keyword will appear, unless the program senses that you're running a BASIC program. Thing to do is turn it off (or get Power). Liz Deal

---

For all of you people who are completely stumped on how "TWENTY QUESTIONS" figures out whether to say "Yes" or "No," look at line 310. You will notice that it is looking at the last character in your inputted question and only *the last character! What it is looking for is a vowel, if it finds one then the answer is "YES"; however, if it does* not *find a vowel the answer is "NO."*

Another thing, on the Apple when you attempt to enter line number 350 you will get the following:

350 IF NOT HEN PRINT "NO"

to fix this problem you will have to enter:

350 IF NO = 1 THEN PRINT "NO"

you may wish to do this for YES (the variable) also.

One other thing, you might wish to change line number 340 to read as follows:

340 IF YES = 1 THEN PRINT "YES":GOTO 390

Clyde Bott

JINSAM™

OFFICIALLY APPROVED BY COMMODORE

Data Manager selected by NASA, Kennedy Space Center

SAVE TIME! SAVE MONEY!
JINSAM saves taxpayers 100's of thousands of $

Coming Soon!
JINSAM Executive for IBM P.C.
See your dealer

"Much more powerful than you can imagine!"
Robert Baker
Kilobaud Microcomputing

JINSAM is an integrated system. It makes it easy to use your information to its fullest. No more will hundreds of valuable hours be spent searching or analyzing needed information nor re-entering information for various reports.

JINSAM transforms your desk-top computer into the "state of the art" data processing machine with features and accessories found nowhere, even at 10 times the price. NASA, Kennedy Space Center selected JINSAM 8.0 and saved approximately $95,000 over other software/hardware costs. Riley County, Kansas also selected JINSAM 8.0 and saved approximately $90,000 over other software/hardware costs.

JINSAM is designed for you. It is forgiving. It has help commands for every option, available at the touch of a button. The amount of information you store, its structure and/or your hardware can change but your data won't have to be re-entered. Recovery utilites are included even for catastrophes, security passwords are built in for privacy, simple editing and entry includes auto recall, and deleting records is easy and the space is reclaimed. JINSAM includes TWO FREE accessories for reports and labels. You have unlimited report formats with summing and lined up decimals and the label printer prints up to 5 across – any size and even allows single envelopes or index cards.

JINSAM 1.0 allows fast and easy file handling, manipulation and report generation for any CBM computer with CBM 2040 disk drive. It features a menu for ease, has encrypted passwords, 3 deep sorts, .5 to 3 second recall.

JINSAM 4.0 for CBM 4000 series adds JINSORT, user accessible machine sort of 1000 records in 15 seconds; compaction/expansion of information, automatic list maintainance, unlimited number of fields, unlimited record length and much more.

JINSAM 8.0 for CBM 8000 series has all 4.0 features plus unlimited sort, horizontal format, and search by key or record number.

JINSAM 8.2, NEW FOR '82 expands 8.0 capabilities by adding information search by word, key or record number and machine language print, format and manipulation routines.

- ★ CUSTOM DATA FILES
- ★ CUSTOM REPORTS/LABELS
- ★ KEYED RANDOM ACCESS
- ★ FAST/EASY/MENU DRIVEN
- ★ MULTIPLE SEARCH KEYS
- ★ PRIVACY ACCESS CODES
- ★ WILD CARD SEARCH

JINSAM EXECUTIVE version (soon to be released) is our most powerful professional system for the CBM 8000 and 9000 series. Executive will have 8.2 extended features plus allow multiple users with in-use lockout protection, executive command files, automatic math relations, join, merge or link files, greatly increased record capacity and machine information search by word, as well as by key or record number and many, many more features.

There are currently 7 more interfacing modules – and more under development, including independent interfaces between JINSAM and business packages for your G/L, A/R, A/P needs. We announce the availability of modules and enhancements in JINSAM's quarterly newsletter.

WORDPROPACK - Intelligent interface for WordPro 3, 3+, 4, 4+, creates lists of information from JINSAM files. It allows up to 10 conditions based on each item of information. Produce individualized letters, report cards, special reports, checks, invoices, etc.

MULTI-LABEL - Prints multiple labels per record with up to 2 lines for messages and consecutive numbering. Produce inventory, caution labels, bulk mail labels, etc.

MATHPACK - global calculator/editor +, −, *, /, by another field or constant; null (remove contents) of a field or replace contents of a field with any word, number or phrase. Sum multiple fields in each record or running sum of single field in all records. Extract information or effect permanent change. Replace in the same field or place in a waiting field.

DESCRIPTIVE STATPACK - Determine MEAN, MEDIAN, MODE, STANDARD DEVIATION, VARIANCE, RANGE. Generate HISTOGRAMS from 1 to 25 steps, and produce Z-SCORE reports.

ADVANCED STATPACK - (You must also acquire DESCRIPTIVE STATPACK) Generate one, two or three way CROSSTABS (number of occurance) CHI SQUARE, LINEAR REGRESSION, with graphic representation and prediction, LINEAR CORRELATION and SIMPLE ANALYSIS OF VARIANCE.

CALCPACK - 2 way interface to VisiCalc or any user program. It lets you use VisiCalc for complex manipulation, editing, placing results in JINSAM for sorting, storing or moving data to WordPro as well as giving the ability for exchange with your own applications.

INTERAC - Interface which can read VisiCalc files, WordPro files and almost any sequential files to build JINSAM databases automatically. For example: You could "download" information on holerith cards to sequential files and INTERAC would place them into JINSAM files.

All accessories are accessed thru the JINSAM menu and require security password to gain entrance.

JINSAM gives you FREEDOM OF CHOICE. Start with JINSAM 1.0 and upgrade hardware and data at any time. Choose from accessories at any time. The JINSAM Newsletter brings the latest updates, user input and uses and keeps an eye on the future.

JINSAM stands alone by placing "a lot of potential computing power in one integrated program package" (Fred Klein, Ferson div. of Bausch & Lomb). "The JINSAM package is justification for buying a system no matter what the hardware, be it Vector or Commodore or whatever the system" (Larry Colvin, Micro Computer Systems). It is a "grandfather" in this young field. JINSAM EXECUTIVE will be the third generation in development. All JINSAM systems are sophisticated and flexible yet easy to use. JINSAM is saving its users valuable time and money in government, educational and research institutions, business and industry worldwide.

JINSAM is a Commodore approved product. See your local dealer for a demonstration.

JINSAM Data Manager

.... Additional Information and nearest dealer
.... Newsletter Subscription ($5 US/$8 Worldwide)
.... JINSAM Demo Disk ($15, plus shipping & tax)
.... User's Guide ($40.00, plus shipping & tax)

Please send to:

Name ______________________

Position ______________________

Company ______________________

Address ______________________

City, State, Zip ______________________

Phone (   ) ______________________

Computer, Disk ______________________

Application ______________________

JINI MICRO-SYSTEMS, Inc.
Box 274C • Kingsbridge Station
Riverdale, New York 10463
(212) 796-6200

Dealer Inquiry Welcome

[In reference to] *Mr. R. O. Danver's request [***COMPUTE!** April, 1982, #23, pg. 14] *for VIC-TTY interface, here are the hardware and software required to* [connect] *an ASR-33 TTY to a VIC: the attached interface will allow the VIC-20 to interface an ASR-33 Teletype with a two way communication capability.*

The ASR-33 is assumed to have a Call Unit #6 as Teletype calls it. This model has a nine pin terminal strip on the rear for connection to the interface. Also Jack #2 also can be used as per the attached diagram.

The ASR-33 must be wired for 20ma. and full duplex. These adjustments can be done inside the unit with the service manual. Also the assumed baud rate is 110. This may be different for a model 35.

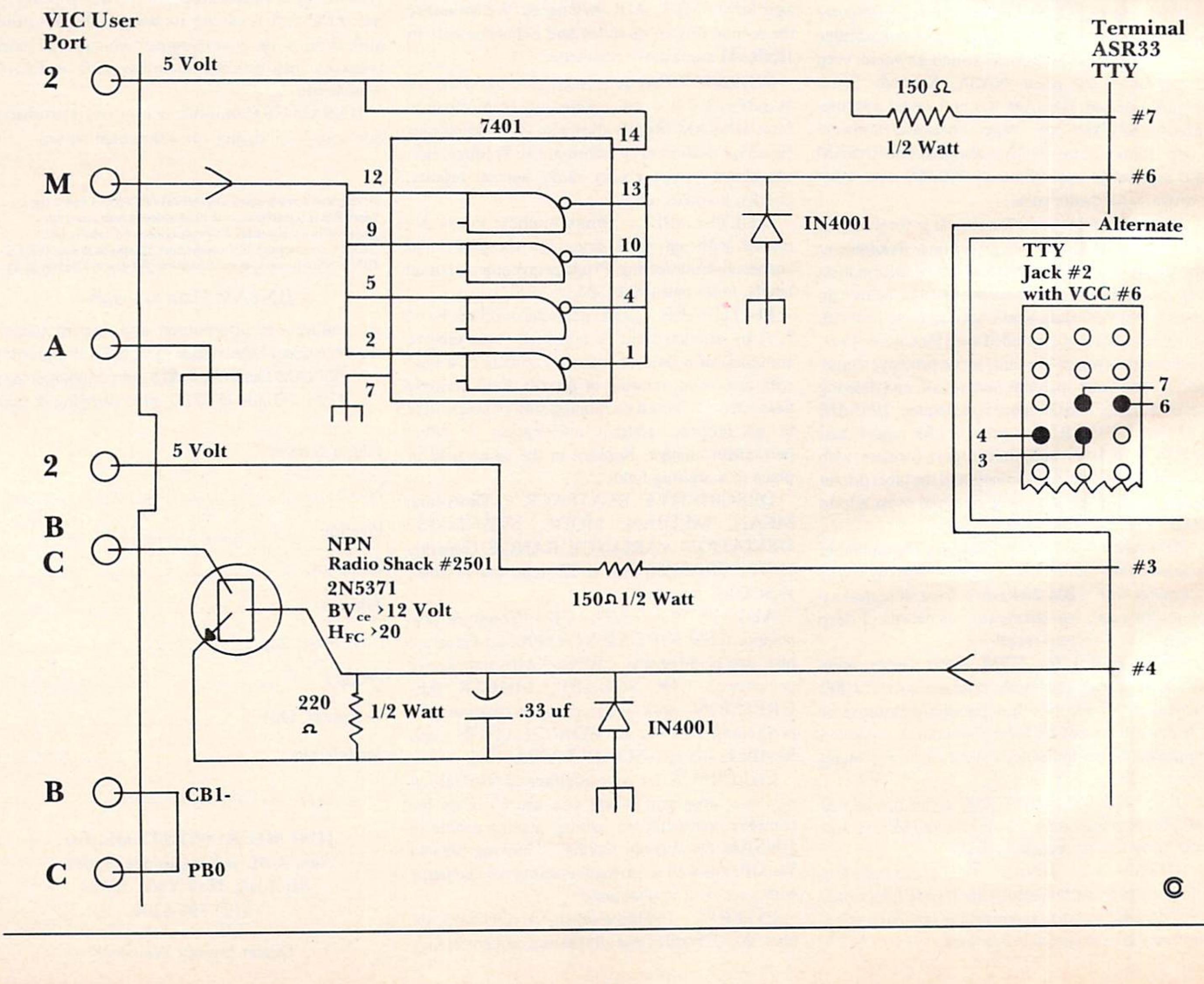
The VIC internal ACIA is set up for full duplex, 7 data bits, a space parity bit, 110 baud and two stop bits. This is more or less the standard configuration.

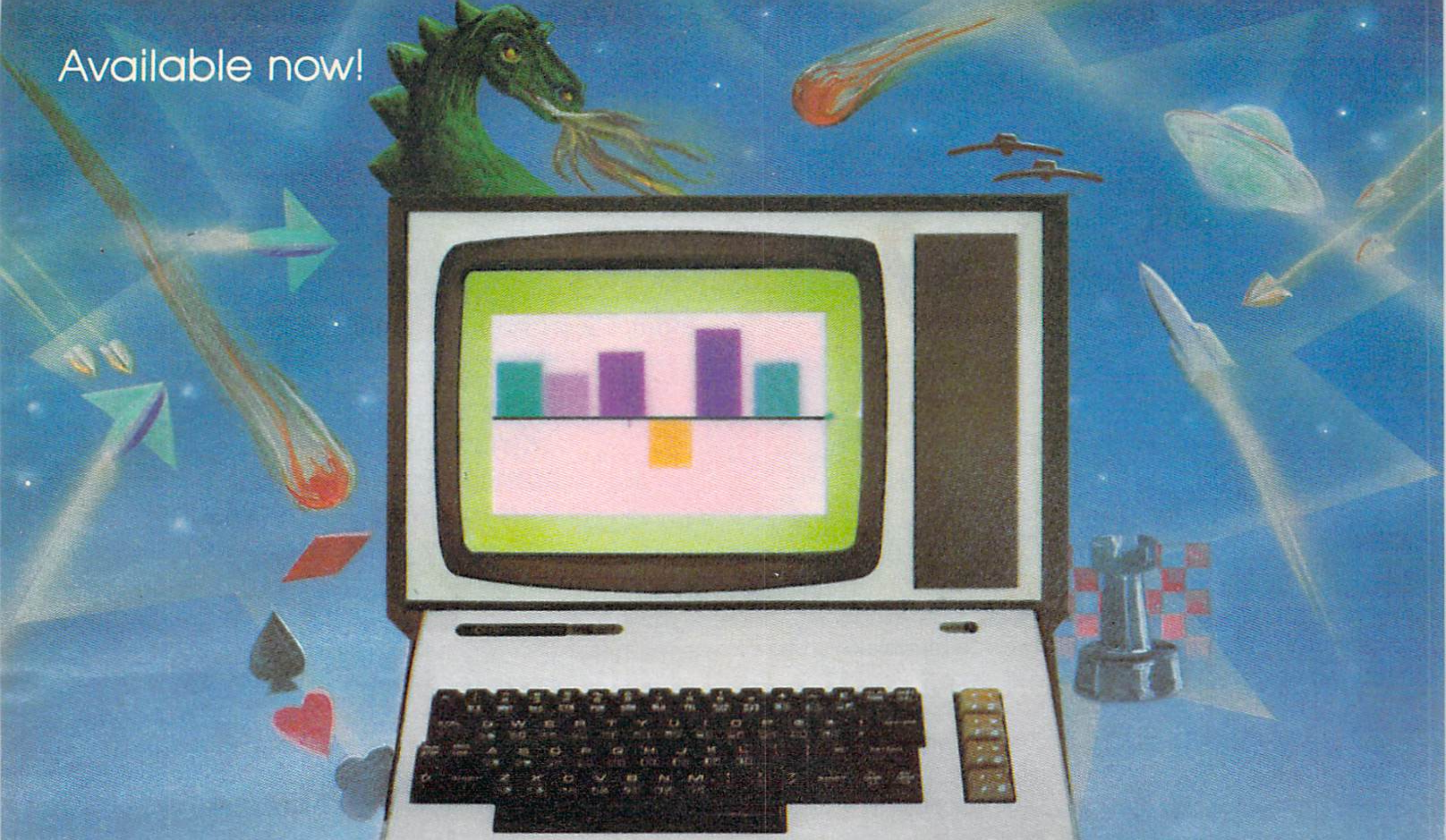
The interface itself is powered from the user port of the VIC and it can both send and receive. The short program is used as a demonstration to show how to use the VIC's serial capability. Kenneth Finn

Program 1.

***Note: Strap keyboard on ASR-33 for space parity.**

```
10 REM ASR 33 TTY
15 REM FILE # > 128 FOR CR WITH LF
20 REM 163=2 STOP,7ASCII, 110 BAUD
30 REM 224=SPACE PARITY, FULL DUPLEX
100 OPEN129,2,3,CHR$(163)+CHR$(224)
110 GET#129,A$
200 REM MAIN LOOP
210 :GET B$
220 IFB$<>""THENIFB$=CHR$(13)THENPRINT#
    129,B$;CHR$(0);CHR$(0);CHR$(0);:GOTO230
225 :IF B$<>""THENPRINT#129,B$;
230 :GET#129,C$:IFC$<>""THEN PRINT#129,C$
    REM ECHO
240 :PRINT B$;C$;
250 SR=ST: IFSR=0 THEN200
300 REM ERRORS
310 PRINT"ERROR";
320 IFSR AND 1 THEN PRINT"PARITY"
330 IFSR AND 2 THEN PRINT"FRAME"
340 IFSR AND 4 THEN PRINT"RCVR BUF FULL"
350 IFSR AND 8 THEN PRINT"BREAK"
360 IF (PEEK(37151)AND 64)=1 THEN 360
370 CLOSE129:END
READY.
```





Make your fun and games computer get serious
For only $79.95

VIC-20[1] 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 VIE which allows you to interface VIC to all existing Commodore IEEE peripherals (8050, 4040, 2031, and printer). In addition, you can build files, access data base and develop complex software while maintaining full expansion capabilities. All for $79.95. And available right now!

And if you want to keep the fun in your computer. Micro-Systems has developed five all new video games. Available in Cartridge $29.95; Cassette (requires 8K expansion) $19.95; and disk (requires IEEE interface above and 8K expansion) $19.95; all compatible with VIC-20. Also available in Atari[2] Game Console Cartridge $29.95.

VICTOROIDS — Eliminate the asteroid belt and enemy ships, only to fight more!

MOBILE ATTACK — Avoid crashing your attack ship with aliens in a labyrinth. Blow out all aliens and move on to next challenge.

CRACK — Escape a rotating prison and destroy its debris before it regenerates.

GUARDIAN — Defend the surface of the planet from multitudes of alien attackers.

QUARK — Consume the screen with boxes while avoiding the Electro-monster and fireballs.

Also available from Micro-Systems

VIC-20 Color Computer	$259.00
VIC-1540 Single Disk Drive (170K, 5¼" Floppy Disk Drive)	$499.00
VIC-1515 Graphic Printer	$325.00
VIC-1110 8K Expansion for VIC	$49.95
VIC-1111 16K Expansion for VIC	$95.00
V-232 RS232C Terminal Interface	$45.00
VTE Victerm I-Terminal Emulator — Tape	$ 8.00
VTE Victerm I-Terminal Emulator — Disk	$15.00

Yes, Please send me:

QTY.	MODEL #	NAME	PRICE
	VIE	IEEE Intf.	
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 • 11105 Shady Trail #103 • Dallas, Texas 75229 • (214) 484-7836

[1] Registered Trademark of Commodore Int. [2] Registered Trademark of Atari, Inc.

Computers And Society

David D. Thornburg
Los Altos, CA

Piracy Revisited...

My February "interview" with the software pirate Long John Silicon generated quite a bit of mail. Because of the sensitive and complex nature of the software copyright issue, I decided to share a reader's response with you and to elaborate a bit on my concerns.

Chris Crawford (Atari software expert *par excellance*) wrote as follows:

> Dave,
>
> I saw your column on piracy in **COMPUTE!** I believe that your logic is imprecise. You imply that effort is the proper index of rightful ownership of designs. While I agree that toil ennobles, I reject your implication that effort confers license. The thief who steals the jewels by dint of vast cleverness and painful effort is still a thief. And no matter how wealthy the owner, how wanton his wastefulness, it is still unethical to steal.
>
> Ethics is no place for sloppy logic. Think it through again – carefully.
>
> Chris Crawford

I agree that ethics is no place for sloppy logic. I am also concerned with Chris's analogy. There is a great deal of difference between stealing the Hope diamond and cutting a new diamond which looks somewhat similar. Nowhere did I condone the idea that it was acceptable for someone to steal a software product, marketing it at the expense of its rightful owner. I think that people who make carbon copies of other people's software and then sell these copies are doing a great disservice to the computer industry, and are breaking the law as well.

I am against true piracy – the copying of existing software for other than personal backup use by one who has purchased the product. This industry will collapse if talented authors aren't guaranteed protection for their effort. The basis of any protective law is that it protects everyone – designer and customer alike. The designer benefits by receiving appropriate compensation for his or her effort and the user benefits by the encouragement this reward provides to developers of new and better software. The best way to drive good designers out of software is to deprive them of their income for their effort. Software copiers might think they are getting something for nothing, but in reality they are damaging the industry.

But this type of copying is not what I had in mind when I wrote my editorial. The question I raised was concerned with the propriety of someone who makes a totally new and improved product which had its genesis in another product.

Software copiers might think they are getting something for nothing, but in reality they are damaging the industry.

In the hypothetical game "Tooth Fairy," Long John Silicon had taken the basic idea behind the arcade game and improved it during its conversion to run on a home computer system. It was not his goal to replicate the original game in every detail. Is he to be denied the right to do this?

What if no new word processor programs could be developed because the authors of the first antiquated teletype-based versions declared broad sweeping rights to the generic field? Who would benefit? The public wouldn't, because the existing software wouldn't be sophisticated enough for their needs, and the original developers wouldn't because their market would dry up.

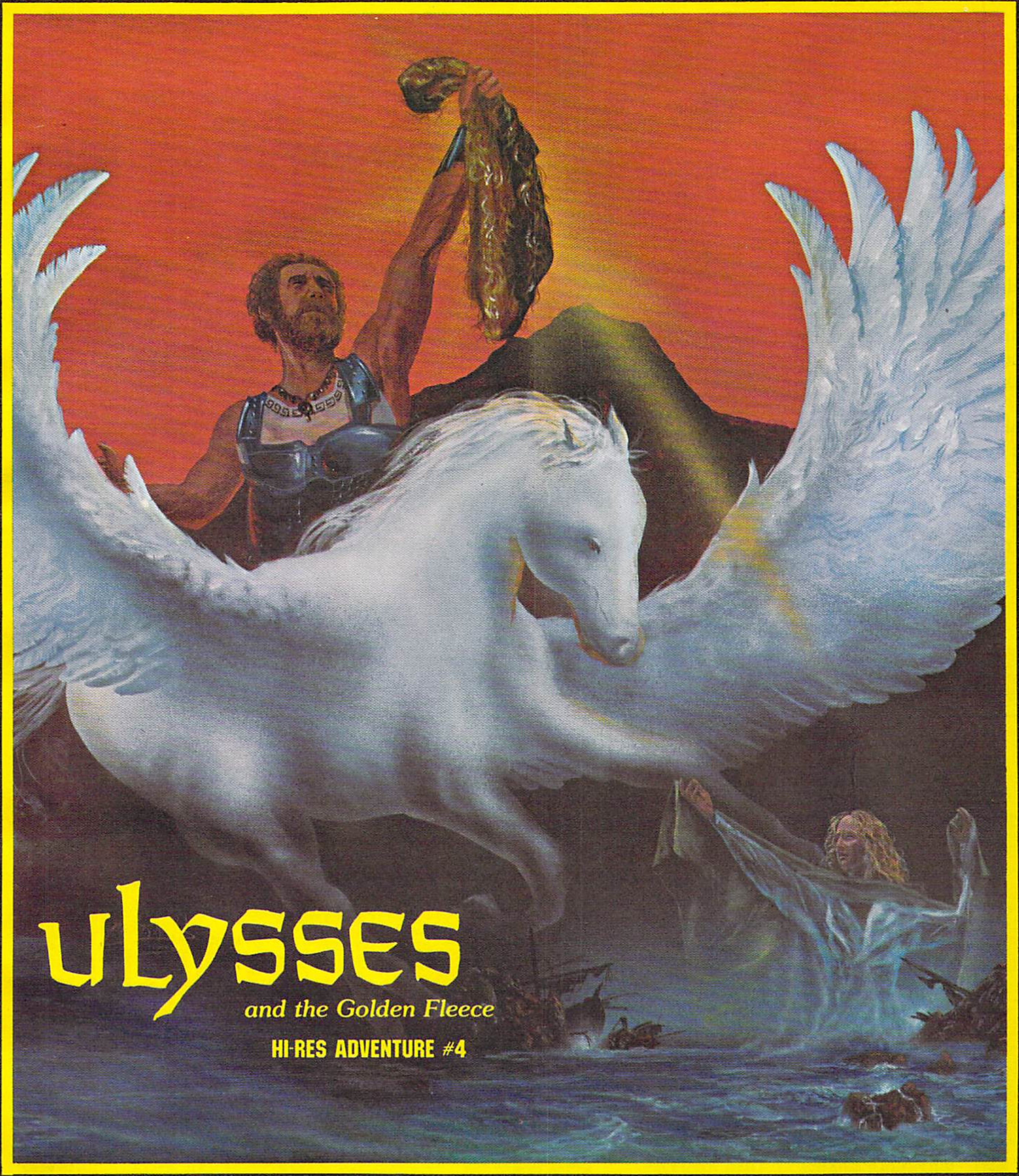
In the game area one might ask if software developers are to be forbidden from improving existing game concepts by adding a tutor mode, by providing dynamic handicapping, by converting a single player game into a multi player game, by modifying the playfield, etc. I may be too dense to follow Chris's logic, but I fail to see how developments along these lines are analogous to "stealing jewels by dint of vast cleverness."

I guess that the use of words like piracy, stealing, etc. to describe activities which, in the area of hardware, are called "patentable differences," really bothers me. Simplistic slogans regarding thievery and piracy bother me when they are applied to issues as complex as those I described.

Are you an accessory to a crime? Have you seen *My Fair Lady,* knowing full well that the authors blatantly made a musical from George Bernard Shaw's play *Pygmalion?* Have you engaged in the criminal act of watching *West Side Story,* knowing that the vicious criminal, Leonard Bernstein, stole the story line from Shakespeare?

Shame on us all.

Imagine how much better off the world would



Bob Davis, Co-ordinator for TIME ZONE, the "ultimate" HI-RES Adventure project, has teamed up with Ken Williams to bring the high adventure of ancient mythology to your home computer. "Cast-off" with ON-LINE SYSTEMS and ULYSSES and see if you can find the GOLDEN FLEECE.

ULYSSES is a HI-RES Adventure on both sides of one diskette. ULYSSES runs on any 48K APPLE II/II+ or 40K ATARI 400/800 with a disk drive and is available for $34.95 at your local computer store or order directly from.....

ON-LINE systems

36575 MUDGE RANCH ROAD, COARSEGOLD, CA 93614 · (209) 683-6858

ADD ONE DOLLAR FOR SHIPPING - VISA, MASTER CHARGE, CHECK COD be if we didn't have "piracy" rampant in the arts – if Archibald MacLeish were thrown behind bars for basing his play *J. B.* on the *Book of Job*. Copying from the Bible, no less – how criminal can one get?

Well, excuse me folks, but to say that one idea is so pristine, so pure, and so complete that it cannot and should not be improved upon is sheer stupidity. It is a concept which hurts designers and users alike. Just as designers must be protected from those who make outright copies of their work, so must they be granted the protection and right to benefit from their significant improvements on existing ideas.

How different is different? This question has plagued philosophers back to the time of Socrates. There are no easy answers.

What I do know is that there is much to be gained from a careful analysis of the problem and little to be gained from righteous finger pointing and sloganeering.

What do *you* think?

Let me know.

David D. Thornburg
Innovision
P.O. Box 1317
Los Altos, CA 94022

POOL 1.5™

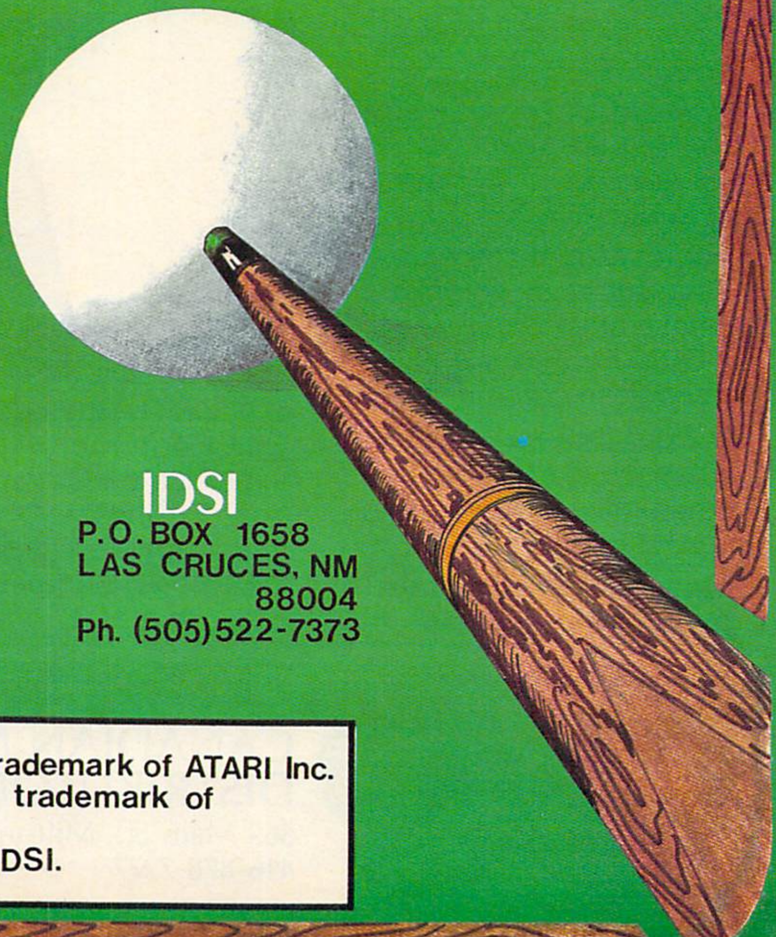
ATARI™ OWNERS,

NOW YOU TOO CAN PLAY POOL 1.5 ON YOUR ATARI 48K DISK SYSTEM.

ALSO AVAILABLE FOR THE APPLE II™ 48K DISK SYSTEM.

PRICE: $34.95

- **High resolution graphics- Life like motion.**
- **Instant replay on any shot.**
- **Superfine aiming.**
- **Pool room sound effects**
- **Choice of 4 popular games**



See your local computer dealer or order directly from IDSI. Specify ATARI or APPLE.

IDSI
P.O. BOX 1658
LAS CRUCES, NM
88004
Ph. (505) 522-7373

ATARI is the registered trademark of ATARI Inc.
APPLE II is the registered trademark of APPLE Computer Inc.
POOL 1.5 is trademark of IDSI.

..one good thing

MUPET

can save you $20,000.

Reduce your hardware costs - let Mupet work for you.

- Mupet allows as many as eight students to share a common disk drive and printer.
- With Mupet software transparency you can have any combination of CBM hardware and programs of your choice.
- Word processors share associated hardware, minimize hardware investment.

Mupet works hard, now priced from $750.00.

Call CMD - the company with world renowned expertise on Commodore and distributors of Mupet and Mupet II, The Manager, DTL Basic Compiler and Supergraphics.

CMD
CANADIAN MICRO DISTRIBUTORS LTD.
365 Main St., Milton, Ontario L9T 1P7
416-878-7277

leads to another..

MUPET II
saves money and time too!

Mupet II offers all the advantages of Mupet, plus:

- Will store commands if disk and printer are being used allowing continued use of computer and completion of stored command in turn.
- Protects against incorrect use of harmful disk commands.
- No more printer interface – background printing supported on IEEE, RS 232C and parallel ports included in controller.
- Programmable 16K RAM controller, expandable to 32K provides user defineable operating systems.
- Up to 16 computers may be connected to central printer and disk drives.

Integrate intra-office terminals for common data access allowing independent analysis for such functions as:

- Accounting
- Sales forecasts, budgets and orders
- Production scheduling
- Word processing

The only product available capable of serving Commodore Computers up to 100 feet apart.

MUPET II a small price for huge capabilities, from:

$995.00

See the notes for using this program on OSI, Apple, Commodore, and Atari computers.

Income Property Report

Roger T. Christensen
Racine, WI

I wrote this program because I needed a fast way to prepare information from my income property for yearly taxes.

The information input into this program is taken off the year-to-date total from my general ledger.

The information given from this report can be enclosed with your federal and state tax returns, also you can keep a copy for yourself. I found there is no reason to save this information so there is no save routine within this program.

I also set this program up to give a general monthly cash flow, profit/loss report on the CRT. This will very quickly give you an idea of what position you are in.

Line 680 is set at 21 cents per mile and can be changed.

Nine inputs can be made under other expenses and other repairs.

Three inputs can be made under other income.

The items listed from line 1320 to line 1350 can be changed.

TAXPAYERS NAME: JOE SMITH
SOCIAL SECURITY NUMBER: 222-22-2111

TYPE OF PROPERTY: 4-FAMILY FRAME

PROPERTY ADDRESS: 1000 MAIN ST
CITY: OURTOWN
STATE: WI.
ZIP: 53000

INCOME

RENTAL INCOME	$9800.00
ESCROW	$ 400.00

TOTAL INCOME = $ 10200

EXPENSES

GAS	$3200.00
ELECTRIC	$ 542.00
WATER	$ 125.00
CLEANING	$ 25.00
TAXES	$ 875.00
INSURANCE	$ 211.00
GARDENING	$ 20.00
RUBBISH	$ 50.00
PEST CONTROL	$ 24.00
PAINT	$ 75.00
PLASTER	$ 10.00
NAILS	$ 5.00
COMMISSION	$ 150.00

REPAIRS

CARPENTRY	$ 510.00
ELECTRICAL	$ 50.00
PLUMBING	$ 100.00
ROOFINNG	$ 500.00
HARDWARE	$ 231.00
MISC.	$ 12.00
DOOR	$ 100.00
WATER HEATER	$ 125.00
FURNACE	$ 525.00

TOTAL EXPENSES = $ 7465

MILEAGE & TRAVEL

1025 MILES @ .21	$ 215.25
OTHER TRAVEL EXP	$ 50.00

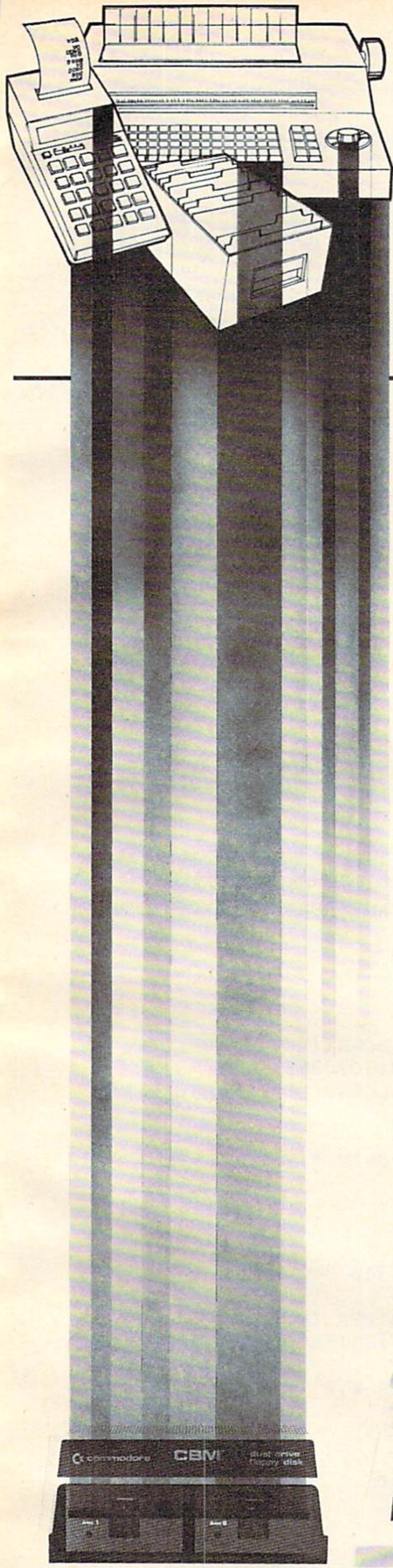
Notes On Using These Programs

Program 1 is written in Microsoft BASIC for the PET, so only minor changes are necessary to convert it to Apple, OSI, or any other BASIC. First of all, follow the suggestions given in COMPUTE's Listing Conventions. The yearly report given from lines 1810 and up are for hardcopy to the Commodore 2022 printer, which supports automatic paging (CHR$(147) sets Top Of Form, CHR$(19) performs forced paging) and formatting commands (similar to PRINT USING).

If your printer has any special features, such as horizontal tabs, you may want to use the approach in the Atari version of this program. It only outputs to the screen, and is not as rigidly formatted. It just TABs from the item field to the numeric field by using a POKE 85,33 statement. You'll want to use HTAB 31 or PRINT TAB(31). You can still have printer output by using PR# (on the Apple) or changing the PRINT statements to LPRINT

Program 1: Microsoft Version

```
10 REM  PROPERTY REPORT BY
20 REM (ROGER T CHRISTENSEN)
30 REM    1006 HAGERER ST
40 REM    RACINE WI 53403
50 REM    (414) 632-6922
60 REM ********9/28/81************
70 REM ***PROGRAM STARTS AT 90 ***
80 REM ***************************
90 DIMP$(10),I(10),Y(100)
100 POKE59468,14
```



LET THE MANAGER DO IT!

Inventory Control
Mailing Lists
Accounting systems
Personnel
Costing
Gathering test data
Budgeting
Scheduling

Organize and File Your Records Without a Programmer

The easy to use Database System suited for both businessman and programmer.

Today's Filing Cabinet for CBM

- Never again misplace an invoice, important document or key customer address. The Manager is your filing cabinet.
- Talk to the Manager and it responds.
- Take command of your filing, reporting and analysis requirements.
- Create, utilize and store information in familiar and compatible forms.
- The Manager automatically performs all predefined calculations as required in your business.
- Utilize it for one major function or include many important records for recall, update and print-out for analysis.
- Keep inventory levels and controls manageable in today's economic environment - the Manager shows you how.
- The Manager saves money, no in-house programmer required; you and/or your secretary can operate the system effectively.
- The Manager saves time; fast search capabilities allows speedy recall and easy merging.

Can You Afford Not To Have The (office) Manager Working For You?

For futher information and the name of your nearest dealer, call CMD - the company with world renowed expertise on Commodore and distributors of Mupet and Mupet II, The Manager, DTL Basic Compiler, and Supergraphics.

THE MANAGER

$250.00

CMD

CANADIAN MICRO DISTRIBUTORS LTD.
365 Main St., Milton, Ontario L9T 1P7
416-878-7277

```
110 PRINT"{CLEAR}{02 DOWN}{11 RIGHT}{REV}IN
    COME PROPERTY"
120 PRINT"{DOWN} THIS PROGRAM CAN BE USED F
    OR A MONTHLY"
130 PRINT"{DOWN}CASHFLOW REPORT GIVEN ON TH
    E {REV}CRT{OFF} OR"
140 PRINT"{DOWN}A YEAR END REPORT ON INCOME
     AND EXPENCES";
150 PRINT"{DOWN}FOR TAXES IN A HARD COPY."
160 PRINT"{DOWN}{REV}ENTER{OFF} '1' FOR A M
    ONTHLY REPORT"
170 PRINT"{DOWN}{REV}ENTER{OFF} '2' FOR A Y
    EARLY REPORT"
180 PRINT"{02 DOWN}A MONTHLY REPORT {REV}DO
    ES NOT REQUIRE{OFF} "
190 PRINT"{DOWN}INFORMATION THAT YEARLY REP
    ORTS REQUIRE{DOWN}"
200 FORAA=1TO1500:NEXT
210 INPUT"NUMBER  1{03 LEFT}";MY
220 IFMY=1THEN410
230 IFMY=2THEN250
240 PRINT"{UP}{REV}INCORRECT DATA":GOSUB135
    0:PRINT"{UP}                 {UP}":G
    OTO200
250 PRINT"{CLEAR}{02 DOWN}{11 RIGHT}{REV}IN
    COME PROPERTY"
260 REM   *****GENERAL INPUT*****
270 PRINT"{DOWN}{REV}ENTER{OFF} SCHEDULE NU
    MBER"
280 INPUTSN
290 PRINT"{DOWN}{REV}ENTER{OFF} YEAR"
300 INPUTDDD
310 PRINT"{DOWN}{REV}ENTER{OFF} TAXPAYER'S
    NAME"
320 INPUTTN$
330 PRINT"{DOWN}{REV}ENTER{OFF} SOC. SEC. N
    UMBER"
340 INPUTSSN$
350 PRINT"{DOWN}{REV}ENTER{OFF} TYPE OF PRO
    PERTY"
360 INPUT OP$
370 PRINT"{DOWN}{REV}ENTER{OFF} PROPERTY AD
    DRESS"
380 INPUTPA$
390 PRINT"{DOWN}{REV}ENTER{OFF} CITY, STATE
    , ZIP"
400 INPUTC$,S$,ZP
410 POKE59468,12:GOSUB1340:GOSUB1290
420 REM*****UTILITY INPUT*****
430 PRINT"{DOWN}{REV}ENTER{OFF} EXPENSES FO
    R:":PRINT
440 FORI=1TO10
450 PRINTP$(I)TAB(15);
460 INPUTN(I)
470 GOSUB1360
480 T=T+N(I)
490 NEXTI
500 GOSUB1340
510 REM****PROGRAM INPUT FOR REPAIRS****
520 PRINT"{DOWN}{REV}ENTER{OFF} EXPENCE'S F
    OR REPAIRS:":PRINT
530 FORK=1TO6
540 PRINTA$(K)TAB(15);
550 INPUT E(K)
560 GOSUB1360
570 U=U+E(K)
580 NEXTK
590 IFMY=1THENGOTO660
600 REM*****MILEAGE & TRAVEL INPUT*****
610 GOSUB1340:PRINT"{03 DOWN}ENTER MILEAGE"
    ;
620 INPUTM
630 GOSUB1360:PRINT"{02 DOWN}ENTER OTHER TR
    AVEL EXPENSES";
640 INPUTET
650 MM=M*.21
660 REM****USER INPUT FOR EXPENSES****
670 GOSUB1340:PRINT"{DOWN}{REV}ENTER{OFF} O
    THER EXPENSES (9)"
680 PRINT"{DOWN}ENTER ITEM ',' AMOUNT"
690 PRINT"{DOWN}{REV}ENTER{OFF} 0,0 {REV}TO
    {OFF} {REV}END{OFF}":PRINT
700 FORL=1TO9
710 INPUTII$(L),A(L)
720 IFA(L)=0THEN760
730 GOSUB1360
740 TA=TA+A(L)
750 NEXTL
760 GOSUB1360:GOSUB1350:GOSUB1340
770 REM****USER INPUT FOR REPAIRS****
780 PRINT"{DOWN}{REV}ENTER OTHER REPAIRS (9
    )"
790 PRINT"{DOWN}ENTER ITEM ',' AMOUNT"
800 PRINT"{DOWN}ENTER 0,0 TO END":PRINT
810 FORC=1TO9
820 INPUTRA$(C),R(C)
830 IFR(C)=0THEN870
840 GOSUB1360
850 TT=TT+R(C)
860 NEXTC
870 GOSUB1360:GOSUB1350:GOSUB1340
880 REM   *****USER INCOME INPUT*****
890 IFMY=2THENGOTO910
900 GOTO920
910 PRINT"{DOWN}{REV}ENTER{OFF} {REV}TOTAL{
    OFF} YEARLY INCOME FOR EACH UNIT"
920 PRINT"{DOWN}HOW MANY RENTAL {REV}UNITS{
    OFF} ":PRINT
930 IFMY=1THENGOTO950
940 PRINT"FOR ";PA$
950 INPUTQ
960 GOSUB1340
970 IFMY=2THENPRINT"{DOWN}{REV}ENTER{OFF} Y
    EARLY RENT":PRINT:GOTO990
980 PRINT"{DOWN}{REV}ENTER{OFF} MONTHLY REN
    T":PRINT
990 FORY=1TOQ
1000 PRINT"RENT FROM UNIT ";Y;
1010 INPUTZ(Y)
1020 GOSUB1360
1030 V=V+Z(Y)
1040 NEXTY
1050 GOSUB1350:GOSUB1340
1060 IFMY=1THEN1170
1070 PRINT"{02 DOWN}OTHER INCOME"
1080 PRINT"{DOWN}{REV}ENTER{OFF} ITEM,AMOUNT
     (3)"
1090 PRINT"{DOWN}ENTER 0,0 TO END":PRINT
1100 FORH=1TO3
1110 INPUTT$(H),EM(H)
1120 GOSUB1360
1130 IFEM(H)=0THEN1160
1140 BB=BB+EM(H)
1150 NEXTH
1160 GOSUB1350:GOSUB1340
1170 REM*****OUTPUT AREA**************
1180 PRINT"      {DOWN}{REV}ENTER{OFF}"
1190 IFMY=1THEN1210
1200 PRINT"{DOWN}'1' FOR PRINTER"
1210 PRINT"{DOWN}'2' FOR SCREEN":PRINT"{DOWN
     DOWN}'3' TO END":PRINT
1220 INPUT"NUMBER  2{03 LEFT}";H
1230 IFMY=1THEN1250
1240 IFH=1THEN1810
```

Introducing the PERCOM Alternative to ATARI Disk Storage

Your Atari 800 is the finest home computer on the market. Now you can own a floppy disk system that measures up — an RFD mini-disk storage system from Percom.

At Percom we've been making disk storage systems since 1977.

Our designs are proven, our quality is well known. And we back our dealers with service know-how. Expect more from Percom. You won't be disappointed.

- Operate in either single- or double-density storage mode using Atari DOS 2.OS. In double-density you can store almost 184 Kbytes (formatted) on one side of a 40-track diskette.
- Connect your Percom RFD first-drive system directly to your computer or connect into your system through your Atari 810 Disk Drive.
- Add an RFD first-drive system with its versatile four-drive controller, then connect as many as three more low-cost RFD add-on drives.
- Write application programs that can query and set up your system to operate a different type drive at each cable position — that can even change configuration as the program executes.
- Get quality and state-of-the-art capability at competitive prices. Percom first-drive RFD systems are priced from only $799, add-on drives from only $399.

Watch for announcement of a new, powerful, easy-to-use disk-operating system for your Percom-equipped Atari 800 computer.

Minimum system requirements — are an Atari 800 computer with 24-Kbytes of RAM and compatible video display system; Atari's disk-operating system (ver 2.OS) and owner's manual; and, for add-on drives (if used) an optional disk drives interconnecting cable available from Percom.

For the best thing next to your computer, see your Atari dealer about a Percom RFD floppy disk storage system. For the name of your nearest dealer, call Percom toll-free 1-800-527-1222.

PERCOM

PERCOM DATA COMPANY, INC.
11220 PAGEMILL RD • DALLAS, TX 75243 • (214) 340-7081

PRICES AND SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE.

Yes . . . I'd like to know more about Percom RFD disk drives for my Atari 800 Computer. Rush me free literature.
Send to: **PERCOM DATA COMPANY, Inc.,** Dept. 18-C01
11220 Pagemill Road, Dallas, Texas 75243

name ____________________
street ____________________
city ____________ state ________
zip ________ phone number ____________
I plan to add a hard disk system to my Atari: ☐ yes ☐ no.
MAIL TODAY!

```
1250 IFH=2THEN1400
1260 IFH=3THEN1730
1270 PRINT"{UP}{REV}INCORRECT DATA":GOSUB135
    0:PRINT"{UP}                    {UP}
    ":GOTO1220
1280 REM*****ITEMS CAN BE CHANGED*****
1290 P$(1)="GAS":P$(2)="ELECTRIC":P$(3)="WAT
    ER":P$(4)="OIL":P$(5)="CLEANING"
1300 P$(6)="TAXES":P$(7)="INSURANCE":P$(8)="
    GARDENING":P$(9)="RUBBISH"
1310 P$(10)="PEST CONTROL":A$(1)="CARPENTRY"
    :A$(2)="ELECTRICAL"
1320 A$(3)="PLUMBING":A$(4)="ROOFING":A$(5)=
    "HARDWARE":A$(6)="MISC.":RETURN
1330 REM*****GOSUB ROUTINES*****
1340 PRINT"{CLEAR}{02 DOWN}{11 RIGHT}{REV}IN
    COME PROPERTY"
1350 FORAA=1TO700:NEXTAA:RETURN
1360 PRINT"{UP}''''''''''''''''''''''''''''''
    ''":RETURN
1370 PRINT"{CLEAR}{HOME}":RETURN
1380 F=T+U+TA+TT:RETURN
1390 REM*****SCREEN PRINTOUT*****"
1400 GOSUB1350:GOSUB1340
1410 Q1$="UTILITIES":Q2$="REPAIRS":Q3$="OTHE
    R EXPENSES":Q4$="OTHER REPAIRS"
1420 Q5$="TOTAL INCOME = $":Q6$="TOTAL EXPEN
    CES = $"
1430 Y=10:VA=V+BB:YY=18
1440 GOSUB1380
1450 F$=STR$(F)
1460 T$=STR$(T):U$=STR$(U):TA$=STR$(TA):TT$=
    STR$(TT):VV$=STR$(VA)
1470 PRINT:PRINT
1480 IFMID$(T$,7,1)<=""THENT$=T$+".00"
1490 PRINTSPC(YY-LEN(Q1$));Q1$;SPC(Y-LEN(T$)
    );T$
1500 PRINT
1510 IFMID$(U$,7,1)<=""THENU$=U$+".00"
1520 PRINTSPC(YY-LEN(Q2$));Q2$;SPC(Y-LEN(U$)
    );U$
1530 PRINT
1540 IFMID$(TA$,7,1)<=""THENTA$=TA$+".00"
1550 PRINTSPC(YY-LEN(Q3$));Q3$;SPC(Y-LEN(TA$
    ));TA$
1560 PRINT
1570 IFMID$(TT$,7,1)<=""THENTT$=TT$+".00"
1580 PRINTSPC(YY-LEN(Q4$));Q4$;SPC(Y-LEN(TT$
    ));TT$
1590 PRINT"##############################":GOS
    UB1350
1600 IFMID$(F$,7,1)<=""THENF$=F$+".00"
1610 PRINTSPC(YY-LEN(Q6$));Q6$;SPC(Y-LEN(F$)
    );F$
1620 IFMID$(VV$,7,1)<=""THENVV$=VV$+".00"
1630 PRINT:PRINTSPC(YY-LEN(Q5$));Q5$;SPC(Y-L
    EN(VV$));VV$
1640 S=V-F
1650 SP$=STR$(S)
1660 IFMID$(SP$,7,1)<=""THENSP$=SP$+".00"
1670 PRINTTAB(20);"########"
1680 IFV>FTHEN1700
1690 IFV<FTHEN1720
1700 PRINT:PRINT"{REV}PROFIT{OFF} >>>>>>>>> ~
    $";SPC(Y-LEN(SP$));SP$
1710 GOTO1740
1720 PRINT:PRINT"{REV}\\\{OFF}  LOSS  {REV}\
    \\{OFF} > $";SPC(Y-LEN(SP$));SP$
1730 GOSUB1350
1740 PRINT"{DOWN}DO YOU WANT ANOTHER REPORT"
1750 PRINT"{DOWN}{REV}Y{OFF}ES OR {REV}N{OFF
    OFF}O";
1760 GETSS$:IFSS$=""THEN1760
1770 IFSS$="Y"THENPRINT"{CLEAR}{04 DOWN}":GO
    SUB1370:GOSUB1340:CLR:POKE59468,14
    :GOTO160
1780 IFSS$="N"THEN1800
1790 PRINT"{CLEAR}{06 DOWN}":GOTO1740
1800 PRINT"{CLEAR}{HOME}":END
1810 REM*****PRINTER PRINT-OUT*****"
1820 PRINT"{CLEAR}{HOME}{03 DOWN}POSITION PA
    PER AND HIT {REV}RETURN{OFF}"
1830 GETVV$:IFVV$=""THEN1830
1840 GOSUB1380
1850 OPEN1,4:OPEN2,4,1:OPEN3,4,2:OPEN4,4,4
1860 PRINT#4
1870 PRINT#1,CHR$(147)
1880 PRINT#1,TAB(30);"INCOME PROPERTY REPORT
             SCHEDULE #: ";SN
1890 PRINT#1,TAB(36);"FOR ";DDD
1900 PRINT#1
1910 PRINT#1,"TAXPAYER'S NAME: ";TN$
1920 PRINT#1,"SOCIAL SECURITY NUMBER: ";SSN$

1930 PRINT#1
1940 PRINT#1,"TYPE OF PROPERTY: ";OP$
1950 PRINT#1
1960 PRINT#1,"PROPERTY ADDRESS: ";PA$
1970 PRINT#1,TAB(10);"  CITY: ";C$
1980 PRINT#1,TAB(10);" STATE: ";S$
1990 PRINT#1,TAB(10);"   ZIP:";ZP
2000 PRINT#1
2010 G$="         AAAAAAAAAAAAAAAA   $9999.99"

2020 PRINT#3,G$
2030 PRINT#1,TAB(10)"INCOME"
2040 PRINT#1,TAB(10);"########"
2050 PRINT#1
2060 PRINT#2,"RENTAL INCOME",CHR$(29),V
2070 PRINT#1
2080 FORH=1TO3
2090 IFEM(H)=0THEN2110
2100 PRINT#2,T$(H),CHR$(29),EM(H)
2110 NEXTH
2120 PRINT#1
2130 PRINT#1,TAB(40);"TOTAL INCOME = $";V+BB

2140 PRINT#1
2150 PRINT#1,TAB(10)"EXPENSES"
2160 PRINT#1,TAB(10)"########"
2170 PRINT#1
2180 FORI=1TO10
2190 IFN(I)=0THEN2210
2200 PRINT#2,P$(I),CHR$(29),N(I)
2210 NEXTI
2220 FORL=1TO9
2230 IFA(L)=0THEN2250
2240 PRINT#2,II$(L),CHR$(29),A(L)
2250 NEXTL
2260 PRINT#1
2270 PRINT#1,TAB(10);"REPAIRS"
2280 PRINT#1,TAB(10);"########"
2290 PRINT#1
2300 FORK=1TO6
2310 IFE(K)=0THEN2330
2320 PRINT#2,A$(K),CHR$(29),E(K)
2330 NEXTK
2340 FORC=1TO9
2350 IFR(C)=0THEN2370
2360 PRINT#2,RA$(C),CHR$(29),R(C)
2370 NEXTC
2380 PRINT#1
2390 PRINT#1,TAB(40);"TOTAL EXPENSES = $";F
2400 PRINT#1
2410 PRINT#1,TAB(10);"MILEAGE & TRAVEL"
```


THEY CAME OUT OF THE BLUE OF THE WESTERN SKY
and it's up to you to stop them!
REAR GUARD
FEATURING ARCADE ACTION GRAPHICS AND SOUND
You know those alien residents who look like blue jello and smell like a cow pasture on a warm day? Well, they're up in arms about something and have declared war! Their weapons are Cyborg ships with their intelligent parts suspended inside. These Cyborgs pursue like crazed bats to try and kamikaze your crew pods. You're armed with an unlimited supply of Guided Energy Darts, but even when you blast 'em, their doomsday missiles come right at you! Good luck. (You'll need it!)
REAR GUARD features many extras which are usually found only on coin-operated video arcade machines. These include advanced play levels, running high score (with high score "clear" option), two player option, brilliant horizontal scrolling, and great sound effects.
APPLE VERSION
by John Anderson
48K DISK $29.95
ATARI VERSION
by Neil Larimer
Assisted by Sparky Starks
WE ARE OVERRUN
16K TAPE $19.95 24K DISK $24.95
TRS-80 VERSION
by Wayne Westmoreland and Terry Gilman
16K TAPE $19.95 32K DISK $24.95
TO ORDER!
Adventure INTERNATIONAL
A DIVISION OF SCOTT ADAMS, INC.
BOX 3435, LONGWOOD FL 32750
(305) 862-6917 (QUESTIONS)
ORDER FROM YOUR FAVORITE DEALER
or CALL TOLL FREE (800) 327-7172
(ORDERS ONLY PLEASE) 120

```
2420 PRINT#1,TAB(10);"################"
2430 PRINT#1
2440 REM******MILAGE COST AT LINE 650****"
2450 PRINT#2,M;" MILES @ .21",CHR$(29),MM
2460 PRINT#1
2470 PRINT#2,"OTHER TRAVEL EXPENSE";CHR$(29)
     ,ET
2480 PRINT#1,CHR$(19)
2490 PRINT#1
2500 CLOSE1,4
2510 PRINT"{02 DOWN}DO YOU WANT ANOTHER HARD
     COPY"
2520 PRINT"{DOWN}{REV}ENTER{OFF} 'Y' IF YOU ~
    DO OTHERWISE 'N'"
2530 GETSS$:IFSS$=""THEN2530
2540 IFSS$="Y"THEN1820
2550 GOTO1160
2560 END
```

Atari Notes

The Atari version has no provision for hardcopy (output to printer). One reason hardcopy is not included is because the type of printer used by Atari owners may vary from a 40-column printer such as the 820 or 822, to an 80-column one such as the 825 or the MX-80. To serve the greatest number of users (including the many who don't own a printer), the yearly report in lines 1810-2510 is written for screen output. Some changes would be necessary to route the output to the printer (like changing PRINT to LPRINT).

Program 2: Atari Version

```
90 DIM P$(10*15),I(10),Y(100),TN$(30),SS
N$(12),PA$(30),C$(15),S$(2),A$(6*15),II$
(30*9),IILN(9),T$(30)
95 DIM A(9),RA$(9*30),RALN(9),TT$(3*30),
TTLN(3),EM(3),N(10),E(6),R(9)
96 DIM Z(10),OP$(20)
100 P$=" ":P$(150)=" ":P$(2)=P$:A$=P$
105 OPEN #1,4,0,"K"
107 FOR I=0 TO 9:A(I)=0:N(I)=0:R(I)=0:Z(
I)=0:NEXT I:Z(10)=0:N(10)=0:FOR I=0 TO 3
:EM(I)=0:NEXT I
108 FOR I=0 TO 6:E(I)=0:NEXT I
110 GRAPHICS 0:POSITION 13,0:? "|INCOME
PROPERTY|"
120 ? "{DOWN}This program can be used fo
r a"
130 ? "{DOWN}monthly cashflow report or"

140 ? "{DOWN}a year end report for incom
e and"
150 ? "{DOWN}expenses for taxes."
160 ? "{DOWN}Enter |1| for a monthly rep
ort"
170 ? "{DOWN}Enter |2| for a yearly repo
rt"
180 ? "{2 DOWN}A monthly report does not
 require"
190 ? "{DOWN}information that yearly rep
orts{DOWN}        require.":?
200 ? "Number?1{2 LEFT}";:INPUT MY
220 IF MY=1 THEN 410
230 IF MY=2 THEN 250
240 ? "{UP}|INCORRECT DATA|{BELL}":? "{U
P}                      {UP}":GOTO 200
250 GRAPHICS 0:POSITION 13,0:? "|INCOME
PROPERTY|"
260 REM **** GENERAL INPUT ****
270 ? "{DOWN}|ENTER| Schedule number"
280 INPUT SN
290 ? "{DOWN}|ENTER| Year"
300 INPUT DDD
310 ? "{DOWN}|ENTER| Taxpayer's name"
320 INPUT TN$
330 ? "{DOWN}|ENTER| Social Security Num
ber"
340 INPUT SSN$
350 ? "{DOWN}|ENTER| Type of property"
360 INPUT OP$
370 ? "{DOWN}|ENTER| Property address"
380 INPUT PA$
390 ? "{DOWN}|ENTER| City":INPUT C$:? "|
ENTER| State (two letter)":INPUT S$
400 ? "|ENTER| ZIP CODE":INPUT ZIP
410 GOSUB 1340:GOSUB 1290
430 ? "{DOWN}|ENTER| Expenses for repair
s:":?
440 FOR I=1 TO 10
450 ? P$(I*15-14,I*15);
460 INPUT T:N(I)=T
470 GOSUB 1360
480 NEXT I
500 GOSUB 1340
510 REM **** PROGRAM INPUT FOR REPAIRS *
**
520 ? "{DOWN}|ENTER| Expenses for repair
s:":?
530 FOR K=1 TO 6
540 ? A$(K*15-14,K*15);
550 INPUT T:E(K)=T
560 U=U+E(K)
570 NEXT K
590 IF MY=1 THEN 660
600 REM *** MILEAGE & TRAVEL INPUT ***
610 GOSUB 1340:? "{3 DOWN}ENTER Mileage"
;
620 INPUT M
630 GOSUB 1360:? "{2 DOWN}ENTER other tr
avel expenses";
640 INPUT ET
650 MM=M*0.21
660 REM *** USER INPUT FOR EXPENSES ***
670 GOSUB 1340:? "{DOWN}|ENTER| other ex
```

```
penses":? "{TAB}(Up to nine)"
680 ? "{DOWN}ENTER ITEM <RETURN> AMOUNT
<RETURN>"
690 ? "{DOWN}|ENTER| <RETURN> when done"

700 FOR L=1 TO 9
705 ? "ITEM";
710 INPUT T$:IF LEN(T$) THEN II$(L*30-29
,(L-1)*30+LEN(T$))=T$:IILN(L)=LEN(T$)
711 IF LEN(T$)=0 THEN 760
715 ? "AMOUNT";
720 INPUT T:A(L)=T:IF T=0 THEN 760
730 GOSUB 1360
740 TA=TA+A(L)
750 NEXT L
760 GOSUB 1360:GOSUB 1350:GOSUB 1340
770 REM *** USER INPUT FOR REPAIRS ***
780 ? "{DOWN}|ENTER OTHER REPAIRS| (Up t
o 9)"
790 ? "{DOWN}ENTER ITEM <RETURN> AMOUNT
<RETURN>"
800 ? "{DOWN}|ENTER| <RETURN> when done"
810 FOR C=1 TO 9
815 ? "ITEM";
820 INPUT T$:IF LEN(T$) THEN RA$(C*30-29
,(C-1)*30+LEN(T$))=T$:RALN(C)=LEN(T$)
822 IF LEN(T$)=0 THEN 760
825 ? "AMOUNT";
830 INPUT T:R(C)=T:IF T=0 THEN 870
840 GOSUB 1360
850 TT=TT+R(C)
860 NEXT C
870 GOSUB 1360:GOSUB 1350:GOSUB 1340
880 REM *** USER INCOME INPUT ***
890 IF MY=2 THEN 910
900 GOTO 920
910 ? "{DOWN}|ENTER| |Total| Yearly inco
me for each    unit"
920 ? "{DOWN}How many rental |units|":?

930 IF MY=1 THEN 950
940 ? "For ";PA$
950 INPUT QQ
960 GOSUB 1340
970 IF MY=2 THEN ? "{DOWN}|ENTER| Yearly
 rent":? :GOTO 990
980 ? "{DOWN}|ENTER| monthly rent":?
990 FOR Y=1 TO QQ
1000 ? "Rent from unit ";Y;
1010 INPUT T:Z(Y)=T
1020 GOSUB 1360
1030 V=V+Z(Y)
1040 NEXT Y
1050 GOSUB 1350:GOSUB 1340
1060 IF MY=1 THEN 1170
1070 ? "{2 DOWN}Other income"
1080 ? "{DOWN}|ENTER| ITEM <RETURN> AMOU
NT <RETURN>":? "(Up to 3)"
1090 ? "{DOWN}|ENTER| <RETURN> when done
"
1100 FOR H=1 TO 3
1105 ? "ITEM";
1110 INPUT T$:IF LEN(T$) THEN TT$(H*30-2
9,(H-1)*30+LEN(T$))=T$:TTLN(H)=LEN(T$)
1111 IF LEN(T$)=0 THEN 1160
1115 ? "AMOUNT";
1120 INPUT T:EM(H)=T:GOSUB 1360
1130 IF EM(H)=0 THEN 1160
1140 BB=BB+EM(H)
1150 NEXT H
1160 GOSUB 1350:GOSUB 1340
1170 REM **** OUTPUT AREA ***
1180 ? "{DOWN}                 |ENTER|"
1200 ? "{DOWN}|1| for screen output"
1210 ? "{DOWN}|2| to end":?
1220 ? "Number?1{2 LEFT}";:INPUT H
1240 IF H=1 AND MY=1 THEN 1400
1245 IF H=1 AND MY=2 THEN 1810
1250 IF H=2 THEN 1800
1270 ? "{UP}|INCORRECT DATA|{BELL}":? "{
UP}                 {UP}":GOTO 1220
1280 REM ITEMS CAN BE CHANGED
1290 DATA GAS,ELECTRIC,WATER,OIL,CLEANIN
G,TAXES,INSURANCE,GARDENING

1300 DATA RUBBISH,PEST CONTROL,CARPENTRY
,ELECTRICAL,PLUMBING,ROOFING,HARDWARE,MI
SC.
1310 RESTORE 1290:FOR I=1 TO 10:READ T$:
P$(I*15-14,(I-1)*15+LEN(T$))=T$:NEXT I
1320 FOR I=1 TO 6:READ T$:A$(I*15-14,(I-
1)*15+LEN(T$))=T$:NEXT I
1330 REM *** SUBROUTINES ***
1340 GRAPHICS 0:POSITION 13,0:? "|INCOME
 PROPERTY|":RETURN
1350 FOR AA=1 TO 100:NEXT AA:RETURN
1360 RETURN
1370 ? "{CLEAR}";:RETURN
1380 F=T2+U+TA+TT:RETURN
1390 REM **** SCREEN PRINTOUT ***
1400 GOSUB 1350:GOSUB 1340
1430 Y=10:VA=V+BB:YY=18
1440 GOSUB 1380
1470 ? :?
1490 ? "UTILITIES: ",T2
1520 ? "REPAIRS:   ",U
1530 ?
1540 ? "OTHER EXPENSES:",TA
1550 ? "OTHER REPAIRS:",TT
1590 ? "|
    |":GOSUB 1350
1600 ? "TOTAL EXPENSES =$",F
1630 ? "TOTAL INCOME =$",VA
1640 S=V-F
1670 ?
1680 IF V>F THEN 1700
1690 IF V<F THEN 1720
```

```
1700 ? "PROFIT >>>>>>>>> $";S
1710 GOTO 1740
1720 ? "LOSS >>>>>>>>>> $";S
1730 GOSUB 1350
1740 ? "Do you want another report"
1750 ? "|Y|es or |N|o?";:POKE 764,255
1760 K=PEEK(764):IF K=255 THEN 1760
1765 POKE 764,255
1770 IF K=43 THEN ? "{CLEAR}{4 DOWN}":GO
SUB 1370:GOSUB 1340:GOTO 160
1780 IF K=35 THEN 1800
1790 ? "{CLEAR}{6 DOWN}":GOTO 1740
1800 GRAPHICS 0:END
1810 REM *** YEARLY REPORT ***
1820 ? "{CLEAR}          INCOME PROPERTY RE
PORT"
1830 ? "                   Schedule#";SN:?
1840 ? "Taxpayer's name:   ";TN$
1850 ? "Social Security # ";SSN$:?
1860 ? "Type of property: ";OP$:?
1870 ? "Property Address: ";PA$
1880 ? "              CITY  : ";C$
1890 ? "             STATE  : ";S$
1900 ? "               ZIP  : ";ZIP
2000 ?
2020 POKE 85,17:? "|INCOME|"
2030 ? :?
2040 ? "Rental Income";:POKE 85,32:? "$"
;V
2050 FOR H=1 TO 3
2060 IF EM(H)=0 THEN 2110
2100 ? TT$(H*30-29,(H-1)*30+TTLN(H));:PO
KE 85,33:? "$";EM(H)
2110 NEXT H
2120 ?
2130 ? "TOTAL INCOME = $";V+BB
2140 ? :? "| PRESS ANY KEY TO CONTINUE|"
;:GET #1,A:? "{CLEAR}"
2150 POKE 85,16:? "|EXPENSES|"
2170 ?
2180 FOR I=1 TO 10
2190 IF N(I)=0 THEN 2210
2200 ? P$(I*15-14,I*15);:POKE 85,33:? "$
";N(I)
2210 NEXT I
2220 FOR L=1 TO 9
2230 IF A(L)=0 THEN 2250
2240 ? II$(L*30-29,(L-1)*30+IILN(L));:PO
KE 85,33:? "$";A(L)
2250 NEXT L
2260 ? :? "| PRESS ANY KEY TO CONTINUE|"
;:GET #1,A
2270 ? "{CLEAR}";:POKE 85,16:? "|REPAIRS
|"
2290 ?
2300 FOR K=1 TO 6
2310 IF E(K)=0 THEN 2330
2320 ? A$(K*15-14,K*15);:POKE 85,33:? "$
";E(K)
2330 NEXT K
2340 FOR C=1 TO 9
2350 IF R(C)=0 THEN 2370
2360 ? RA$(C*30-29,(C-1)*30+RALN(C));:PO
KE 85,33:? "$";R(C)
2370 NEXT C
2380 ?
2390 ? "TOTAL EXPENSES = $";F
2400 ? :? "| PRESS ANY KEY TO CONTINUE|"
;:GET #1,A
2410 ? "{CLEAR}":POKE 85,11:? "Mileage a
nd Travel"
2420 ?
2440 REM *** MILEAGE COST AT LINE 650 **
*
2450 ? M;" Miles @ .21 : $";MM
2460 ?
2470 ? "Other travel expense: $";ET
2480 ? :?
2490 ? :? "| PRESS ANY KEY TO CONTINUE|"
;:GET #1,A
2510 GOTO 1160
```

©

Using *your* computer
in an interesting
application?
Write it up for
other **COMPUTE!**
readers to use.

This strategy game will run as presented on any Commodore computer. A version for the Atari is included separately. The Commodore version will also run on an Apple with the changes noted at the end of the article. Originally written for the VIC, it has no sound effects (to save memory). These were added to the Atari Version.

Outpost

Tim Parker
Kanata, Ontario

The object of Outpost is to survive. You are placed in an immovable outpost, armed with torpedoes, main and secondary energy armaments, and a targeting computer. Your opponents come in three sizes, labelled SML (small), MDM (medium) and HVY (heavy). Their objective is to overrun you, or destroy you by knocking out your armaments, computer, or energy supply.

When RUN, the screen gives you quite a lot of information. Your status is displayed to the right center, where values for ENGY (energy), COMP (computer), MAIN (main armament), SECN (secondary armament), TORP (torpedoes), and VP (victory points) are displayed. Energy is rated from zero to ninety-nine. If the energy drops to zero, you lose the game. Computer efficiency, and both main and secondary armaments, are rated as a percentage of capability. Ninety-nine is maximum.

If the computer falls to zero, you have lost all defensive capabilities, and lose the game. If either or both armaments fall to zero, they cannot be fired until recharged by a supply ship. Five torpedoes are supplied at the beginning of the game. A maximum of nine can be stored in the outpost at any time. Victory Points is your score. For each light enemy ship destroyed, one victory point is awarded; similarly, two for medium, and three for heavy ships.

The top of the screen shows the enemy. Up to four are active at a time. Each enemy ship is referenced by a number on the "radar screen" at center left. The index above gives the DIST (distance), PROB (hit probability), and ENGY (energy) of the enemy. The hit probability is a function of both enemy distance and your computer efficiency. If enemy energy falls to zero it is destroyed, and victory points are awarded.

When playing, the computer will give you a "Weapon" prompt. This requires an input of T(orpedo), M(ain), or S(econdary) for the different weapons. C can be entered to recharge your batteries, and the energy of the outpost will increase when employed, to a maximum of 99. If a weapon is being fired, the prompt "TARGET NO" appears, requiring a value of one to four, depending on the enemy number.

After your turn, the computer will move some of the enemy ships, and some will fire at you. They have a hit probability that is a function of their energy. Damage to energy, computer, or armaments may result.

Occasionally, a supply ship wanders onto the screen. This is shown by a white "S". If it reaches you successfully, it recharges energy, main and secondary armaments to full power, and adds up to five torpedoes. Since a maximum of nine can be held at one time, any extras are lost. Note that the supply ship does not recharge your computer. The supply ships can be destroyed if an enemy lands on top of them.

As might be expected, a hit on an enemy ship will decrease its energy. The amount of damage done is proportional to the type of enemy ship; the heavy ships are harder to destroy than mediums and lights. The type of weapon fired also affects damage. On an efficiency scale, torpedoes, main and secondary armament are approximately 9:6:4 in damage ratios. A few trial games quickly gives a feel for this.

High scores are not always easy to get. If a score of twenty is achieved, you are very good. Forty is excellent. Sixty is almost impossible, unless you're extremely lucky.

Strategy

The light ships are the most easily destroyed, but they do the least damage to you. If a heavy ship appears, try to get it fast. If an enemy gets within two moves of you, hit it hard. If it lands on you, you are destroyed. Also, protect your supply ships. They are needed and they are easily destroyed by the enemy.

As the computer efficiency rating drops, the hit probability also drops. With low computer values, you'll find that you have to wait for the enemy to get close before wasting shots. Torpedoes shouldn't be wasted, especially on low probability shots. If you get a few enemy ships on the screen at once, pick them off one at a time if possible to try and avoid concentrated fire. If you have four heavy ships bearing down on you, it's wisest to panic. If no enemy ships are on the screen, charge your batteries.

The Program

The program is divided into several blocks:

lines	**0-999**	**Control section**
	1000-1999	**Screen display**
	2000-2999	**Refuel routine**
	3000-3999	**Enemy movement**

Cavalier is looking for your original programs. You get the highest royalties and the best national exposure, plus a team of experts that'll smooth out those rough edges to make your program even better.

Your program is thoroughly tested before it's packaged, and pre-tested in selected U.S. markets. We're currently seeking programs in:

Apple, IBM, Intellevision, Atari, Atari VCS, TRS-80, and Odyssey 2.

All Cavalier programs have reached the "Top Thirty" charts — now what about yours? Call or write today for your FREE "Programmer's Kit" with complete information including stock options.

Cavalier, the company that cares.

Full-time/part-time author inquiries invited.

CAVALIER COMPUTER CORPORATION

P. O. BOX 2032 — DEL MAR, CA 92014 — (714) 755-8143

SUNRISE SOFTWARE

LEADER IN DISCOUNT SOFTWARE — 18 - 30% OFF

ATARI

Action Quest	D	13.95
Adventures 1-12 ea.	T	15.95
Adventure on a Boat	D	19.95
Ali Baba & Forty Thieves	D	26.35
Angle Worms	T	12.00
Andromeda	D	31.96
Assembler Editor	C	47.95
Asteroids	C	35.95
Bishops Square	D	23.95
Black Forest	D	19.95
B-1 Nuclear Bomber	T	13.00
Bug Attack	D,T	23.95
Centipede	C	35.95
Character Generator	D	15.95
Conflict 2500	T	13.00
Compu-Read	D,T	15.95
Compu-Math/Fractions	D,T	15.95
Compu-Math/Decimals	D,T	15.95
Crush, Crumble & Chomp	D,T	23.95
Crossfire	D,T	23.95
Datestones of Ryn	D,T	15.95
Deflection	T	11.95
Dung Beetle	D	23.95
Empire of Overmind	D	28.00
Entertainer Kit	C	95.00
Fastgammon	T	15.95
Fort Defiance	T	16.00
Frogger	D	27.95
Galactic Empire	T	15.95
Galactic Trader	T	15.95
Ghost Hunter	D,T	16.78
Ghostly Manor	D	19.95
HI-RES Mission Asteroid	D	19.95
HI-RES Wiz & Princess	D	26.35
Invasion Orion	D,T	19.95
Jawbreaker	D	23.95
Kayos	D,T	27.95
K-Razy Shootout	C	39.95
Lisp Interperter	D	120.00
Lords of Karma	T	16.00
Lunar Lander	T	12.00
Match Racers	D	23.96
Missle Command	C	35.95
Mountain Shoot	T	12.00
Mouseattach	D	27.95
Next Step	D	31.95
No. Atl. Convoy Raider	T	13.00
Nukewar	T	13.00
Pac Man	C	35.95
Pathfinder	D	27.95
Planet Miners	T	13.00
Pool 1.5	D	27.95
Programmer Kit	C	63.88
Q.S. Forth	D	63.96
Raster Blaster	D	23.95
Rear Guard	T	15.95
Rescue at Rigel	D,T	23.95
Ricochet	D,T	15.95
Shattered Alliance	D	31.95
Softporn Adventure	D	23.95
Space Invaders	C	35.95
Starbase Hyperion	D	18.35
Star Raiders	C	39.95
Star Warriors	D,T	31.95
Sunday Golf	T	13.00
Tanktics	D	23.20
Temple of Apshai	D,T	31.95
Threshold	D	31.95
3D Tic-Tac-Toe	T	15.95
3D Supergraphics	D,T	31.95
Treasure Quest	T	15.95
VersaWriter Tablet	A	240.00
Visicalc	D	200.00
Warlocks Revenge	D	28.00

PET / VIC-20

UNITED MICROWARE

Amok	T	15.15
Alien Blitz	T	19.95
Blastoids	T	15.15
Breakout	T	12.00
Carom	T	12.00
Dragon Maze	T	13.55
Gobbler	T	19.95
Invader Fall	T	17.55
Laser War	T	15.15
Long Division	T	13.55
Meteor Run	C	47.95
Raceway	T	12.00
Robot Blasters	C	31.95
Skymath	T	12.00
Space Division	T	12.00
Spiders of Mars	C	47.95
Super Hangman	T	15.15
Stars Wars	T	13.55
3D Maze	T	12.00
ViCalc	T	11.95
ViCat	T	19.95
ViCheck	T	19.95
ViTerm A.	T	15.95
RS232 Board	A	39.95
3K Cartridge	A	31.95
8K Cartridge	A	71.95

Others

B-1 Bomber	T	13.00
Computer Acquire	T	16.00
Computer Stocks & Bonds	T	13.00
Conflict 2500	T	13.00
Conquest	D	180.00
Dragons Eye	T	15.96
Guns of Fort Defiance	T	16.00
Midway Campaign	T	13.00
N.A. Convoy Raider	T	13.00
Planet Miners	T	13.00
Thinker	D	396.00

★ ★ ★ ★ ★ ★ ★ ★ ★ ★

BONUS

★ ★ ★ ★ ★ ★ ★ ★ ★ ★

PAC MAN

FOR ONLY $30.00

with

ANY ORDER OVER

$150!

★ ★ ★ ★ ★ ★ ★ ★ ★ ★

D-Disk, C-Cartridge, A-Accessory, T-Cassette

24 HOURS A DAY
7 DAYS A WEEK

ORDER TOLL FREE
800 - 854-0561 EXT. 812
800 - 432-7257 (CALIF.)

NO HASSLE!
GUARANTEED
REPLACEMENT POLICY

We take MASTERCARD or VISA. (Include card # and expiration date.) California residents add 6% tax. Include $2.00 for postage. Prices subject to change.

MasterCard VISA

TYPE RIGHT on your own MICROCOMPUTER

TYPE RIGHT is a complete course of 22 typing programs.

TYPE RIGHT has easy directions. The beginner will find all necessary instructions displayed on the screen, but there is extensive documentation for the programmer who wishes to make adaptations.

TYPE RIGHT is varied to keep your interest. You will be typing hundreds of real words, not just random letter drills. Your spelling and writing skills will improve. There are even five games!

TYPE RIGHT emphasizes accuracy first, then speed.

TYPE RIGHT computes words per minute the official way. (Errors are subtracted.)

TYPE RIGHT is a fantastic educational value at less than $2 per program.

BARRON ENTERPRISES
714 Willow Glen Road
Santa Barbara, CA 93105
(805) 687-5873

$29.95 PET (8K)
+ $1.50 shipping &
6% CA tax

☐ cassette ☐ diskette

4000-4999	**Enemy fire**
5000-5999	**Enemy ship & Supply ship appearances**
6000-6999	**Weapons and firing routine**
9500-9999	**Destroyed routine.**

When RUN, the program loops through each section, beginning with your fire routine, enemy fire and movement, new ships, and the screen routine. This is controlled by line 500.

The odds of a ship appearing are given in lines 5005 and 5010. The four ships are listed as subscripts of ET(x), with a value of one for light, two for medium, and three for heavy ships. ET(5) is the supply ship, and has a value of five if one is on the screen, and zero otherwise. EH(x) is the ship's hit probability, given by line 40, and ED(x) is the distance, given in line 30. To change the difficulty level of the game, change the values of the number following "FNA(x)" in lines 5005 and 5010. If a ship is scheduled to appear, line 5110 chooses the type. Light are the most probable.

The "radar screen" is generated by section 1500-1700 using an individual coordinate system. EX(x) and EY(x) define each enemy ship location. This is a slow method for generating the display, but offers the best memory saving.

GET statements are used throughout to avoid the inevitable RETURN hit.

If you wish to avoid wearing your fingers to the bone, I will be glad to duplicate the Commodore version of the program if you send $3.00, a blank tape, and an addressed mailer (no stamps) to:

Tim Parker
66 McKitrick Dr.
Kanata, ONT, Canada K2L1T7

Notes For Apple Owners:

Outpost is not too hard to adapt to the Apple since it is primarily a straightforward text display. The only problems are in the formatting of the display. Follow the suggestions given in **COMPUTE!**'s Listing Conventions, e.g. replace [CLEAR] with 'HOME', [LEFT] with CHR$(8), etc. Ignore all the VIC color controls such as WHT, CYN, BLK, etc. You may want to replace lines like 1690:

1690 PRINT "[HOME][6 DOWN]"

with

1690 VTAB 6

In line 1305, the underlined D's represent dashes, and the underlined Q in line 1630 is a "ball character." You can put whatever you like there, perhaps an asterisk.

Program 1: Commodore Version

```
10 PRINT"{CLEAR}":POKE36879,76
20 DEFFNA(X)=INT(RND(1)*X+1)
30 DEFFNB(Z)=INT(SQR((EX(G)-6)^2+(EY(G)-6)
   ^2))
40 DEFFNC(Z)=INT(1/(ED(G))*100+(C/2))
50 C=99:G=1:GOSUB5110:GOSUB2000
200 GOSUB5000:GOSUB1000:GOSUB6000:GOSUB3000
    :GOSUB4000
210 GOTO200
1000 PRINT"{HOME}{DOWN}{BLK}ENEMY 1   2   3
    4"
1020 PRINT"{WHT}TYPE ";
1030 FORG=1TO4
1040 IFET(G)=0THENPRINT"--- ";
1050 IFET(G)=1THENPRINT"LGT ";
1060 IFET(G)=2THENPRINT"MDM ";
1070 IFET(G)=3THENPRINT"HVY ";
1080 NEXT
1100 PRINT:PRINT"DIST":PRINT"PROB
       ":PRINT"ENGY                  "

1120 FORG=1TO4
1122 X=1+G*4
1124 PRINT"{04 UP}"
1126 PRINTSPC(X)ED(G)
1128 PRINTSPC(X)EH(G)
1130 PRINTSPC(X)EE(G)
1140 NEXTG
1300 PRINT:PRINTSPC(12);" {BLU}STATUS"
1305 PRINTSPC(12)" DDDDDD"
1310 PRINTSPC(11)" {CYN}ENGY:    {03 LEFT}";E

1320 PRINTSPC(11)"{YEL} COMP:    {03 LEFT}";C

1330 PRINTSPC(11)"{CYN} MAIN:    {03 LEFT}";M

1340 PRINTSPC(11)" SECN:    {03 LEFT}";S
1350 PRINTSPC(11)" TORP:";T
1360 PRINTSPC(11)"  VP :";VP
1400 PRINT:PRINTSPC(11);" {RED}C=CHARGE":PRI
    NT:PRINT
1500 PRINT"{HOME}{06 DOWN}"
1510 A=0
1520 FORY=1TO11
1530 FORX=1TO11
1540 FORG=1TO5
1550 IFY<>EY(G)THEN1620
1560 IFX<>EX(G)THEN1620
1570 A=1:IFG=1THENPRINT"{BLK}1";
1580 IFG=2THENPRINT"{BLK}2";
1590 IFG=3THENPRINT"{BLK}3";
1600 IFG=4THENPRINT"{BLK}4";
1610 IFG=5THENPRINT"{WHT}S";
1620 NEXTG
1630 IFX=6ANDY=6THENPRINT"{GRN}Q";:A=1
1640 IFA=1THENA=0:GOTO1660
1650 PRINT"{GRN}+";
1660 NEXTX
1670 PRINT
1680 NEXTY
1690 PRINT"{HOME}{06 DOWN}"
1700 FORA=1TO11
1710 PRINTSPC(11)" "
1720 NEXT
1990 PRINT:RETURN
2000 ET(5)=0:EX(5)=0:EY(5)=0
2010 E=99:M=99:S=99
2020 T=T+5:IFT>9THENT=9
2030 RETURN
3000 FORG=1TO5:IFET(G)>0THEN3100
```

```
3010 NEXTG:RETURN
3100 IFG<5ANDFNA(9)>5THEN3010
3200 IF EX(G)>6THENEX(G)=EX(G)-1
3210 IF EX(G)<6THENEX(G)=EX(G)+1
3220 IFEY(G)<6THENEY(G)=EY(G)+1
3230 IFEY(G)>6THENEY(G)=EY(G)-1
3240 IFET(5)=5ANDEY(5)=6ANDEX(5)=6THENGOSUB2
     000
3250 IFEY(G)=6ANDEX(G)=6THEN9500
3265 IFG<5ANDEX(G)=EX(5)ANDEY(G)=EY(5)THENET
     (5)=0:EX(5)=0:EY(5)=0
3300 ED(G)=FNB(1)
3330 EH(G)=FNC(0):IFEH(G)>99THENEH(G)=99
3400 GOTO3010
4000 PRINT"{UP}ENEMY FIRING & MOVING"
4010 FORG=1TO4:IFET(G)<>0THEN4100
4020 NEXTG:RETURN
4100 IFFNA(99)>(EE(G)+FNA(30))OREE(G)<10THEN
     4020
4110 E=E-FNA(5)*ET(G)
4150 EE(G)=EE(G)-FNA(10)
4160 IFFNA(10)=1THENC=C-FNA(25):IFC<1THEN950
     0
4170 IFFNA(10)=1THENM=M-FNA(25):IFM<0THENM=0

4180 IFFNA(10)=1THENS=S-FNA(25):IFS<0THENS=0

4200 IFE<0THEN9500
4210 GOTO4020
5000 G=FNA(5)
5005 IFG=5ANDET(5)=0ANDFNA(4)>1THENET(5)=5:G
     OTO5160
5010 IFG=5ORET(G)<>0ORFNA(9)>4THEN5400
5110 A=4-INT(LOG(FNA(50)+2))
5120 ET(G)=A:EE(G)=99
5160 EX(G)=FNA(11)
5170 EY(G)=FNA(11)
5180 A=FNA(4):IFA=1THENEY(G)=1
5190 IFA=2THENEY(G)=11
5200 IFA=3THENEX(G)=11
5210 IFA=4THENEX(G)=1
5300 ED(G)=FNB(1)
5320 EH(G)=FNC(0):IFEH(G)>99THENEH(G)=99
5400 RETURN
6000 PRINT"{BLK}WEAPON:              "
6010 GETA$:IFA$=""THEN6010
6020 IFA$="M"ANDM>0THENA=6:M=M-FNA(5):IFM<0T
     HENM=0
6025 IFA$="C"THENE=E+FNA(20):IFE>99THENE=99
6030 IFA$="C"THENRETURN
6035 IFA$="S"ANDS>0THENA=4:S=S-FNA(5):IFS<0T
     HENS=0
6040 IFA$="T"ANDT>0THENA=9:T=T-1
6060 IFA<3THENPRINT"{UP}BAD INPUT! WEAPON:":
     GOTO6010
6100 PRINT"{UP}TARGET NO:              "
6120 GET B$:IFB$=""THEN6120
6125 B=VAL(B$)
6130 IFET(B)=0THENPRINT"{UP}BAD DATA! TARGET
     :":GOTO6120
6200 IFFNA(99)>EH(B)THENPRINT"{UP}MISSED!
     ":FORZ=1TO1000:NEXT:RETURN
6210 EE(B)=INT(EE(B)-((A*FNA(15))/ET(B)))
6215 PRINT"{UP}{WHT}TARGET HIT        ":FORZ=1
     TO1000:NEXT
6220 IFEE(B)<1THEN6500
6230 E=E-FNA(5)
6300 RETURN
6500 VP=VP+ET(B)
6505 EX(B)=0:EY(B)=0
6510 ET(B)=0:EH(B)=0:ED(B)=0:EE(B)=0
6570 PRINT"{UP}{BLU}{REV}  TARGET DESTROYED!
     "
6575 FORA=1TO1000:NEXT
6580 RETURN
9500 POKE36879,110
9510 PRINT"{CLEAR}{WHT}{03 DOWN}        DESTROYE
     D!!!!!"
9550 PRINT"{04 DOWN}        SCORE= ";VP:PRINT:P
     RINT
9560 IFVP>HSTHENHS=VP
9580 PRINT"{02 DOWN} {GRN} *****************
     "
9590 PRINT" {GRN}  HIGH SCORE= ";HS
9600 PRINT" {GRN} *****************"
9605 PRINT"{04 DOWN}  {WHT}  ANOTHER GAME?"
9610 GETA$:IFA$=""THEN9610
9620 IFA$="Y"THENRUN
9630 STOP
```

Program 2: Atari Version

The game has changed very little in conversion to the Atari. Several sound effects were added to increase the excitement of the game. Also notice that the "radar screen" is set off in a green box by using player/missile graphics. If you have 8K of memory, you might want to delete lines 7000-7060 and just use:

7000 RETURN

```
10 GRAPHICS 0:POKE 752,1:GOTO 50
20 X=INT(RND(0)*X+1):RETURN
30 Z=INT(SQR((EX(G)-6)^2+(EY(G)-6)^2)):R
ETURN
40 Z=INT(1/(ED(G))*100+(C/2)):RETURN
50 DIM S$(11),R$(40):OPEN #1,4,0,"K":G=1

51 DIM ET(5),EH(5),EX(5),EE(5),EY(5),ED(
5)
52 R$="{RIGHT}":R$(40)="{RIGHT}":R$(2)=R
$
53 S$=R$(1,11):REM 11 SPACES
55 C=99:G=1:GOSUB 8000:GOSUB 5110:GOSUB
2000:GOSUB 7000
200 GOSUB 5000:GOSUB 1000:GOSUB 6000:GOS
UB 3000:GOSUB 4000
210 GOTO 200
1000 POSITION 2,1:? "|ENEMY|       1
  2      3      4"
1020 ? "TYPE      ";
1030 FOR G=1 TO 4
1040 IF ET(G)=0 THEN ? " ------";
1050 IF ET(G)=1 THEN ? " LIGHT ";
1060 IF ET(G)=2 THEN ? " MEDIUM";
1070 IF ET(G)=3 THEN ? " HEAVY ";
1080 NEXT G
1100 ? :? "Distance":? "Prob.
          ":? "Energy
     "

1120 FOR G=1 TO 4
1122 X=G*7+5
1124 ? "{4 UP}"
1126 ? R$(1,X);ED(G);" "
1128 ? R$(1,X);EH(G);" "
1130 ? R$(1,X);EE(G);" "
```

```
1140 NEXT G:? :POKE 201,11
1150 FOR I=8 TO 19:POSITION 2,I:? ,:NEXT
 I:POSITION 2,8
1300 ? S$;" |STATUS|"
1305 ? S$;" {6 R}":?
1310 ? S$;" ENERGY        {3 LEFT}";E
1320 ? S$;" COMPUTER      {3 LEFT}";C
1330 ? S$;" MAIN          {3 LEFT}";M
1340 ? S$;" SECONDARY     {3 LEFT}";S
1350 ? S$;" TORPS         {3 LEFT}";T
1360 ? S$;" V.P.          {3 LEFT}";VP
1400 ? :? S$;" C=CHARGE":? :?
1500 POSITION 2,8
1510 A=0
1520 FOR Y=1 TO 11
1530 FOR X=1 TO 11
1540 FOR G=1 TO 5
1550 IF Y<>EY(G) THEN 1620
1560 IF X<>EX(G) THEN 1620
1570 A=1:IF G<>5 THEN ? G;:FOR W=150 TO
0 STEP -10:SOUND 0,W,12,W:NEXT W
1610 IF G=5 THEN ? "|S|";:FOR W=150 TO 0
 STEP -10:SOUND 0,I,10,W/10:NEXT W
1620 NEXT G
1630 IF X=6 AND Y=6 THEN ? "{T}";:A=1
1640 IF A=1 THEN A=0:GOTO 1660
1650 ? "+";:POKE 53279,0
1660 SOUND 0,0,0,0:NEXT X
1670 ?
1680 NEXT Y
1690 POSITION 2,6
1700 FOR A=1 TO 11
1710 ? R$(1,11);" "
1720 NEXT A
1990 ? :RETURN
2000 ET(5)=0:EX(5)=0:EY(5)=0
2010 E=99:M=99:S=99
2020 T=T+5:IF T>9 THEN T=9
2030 RETURN
3000 FOR G=1 TO 5:IF ET(G)>0 THEN 3100
3010 NEXT G:RETURN
3100 IF G<5 AND INT(9*RND(0)+1)>5 THEN 3
010
3200 IF EX(G)>6 THEN EX(G)=EX(G)-1
3210 IF EX(G)<6 THEN EX(G)=EX(G)+1
3220 IF EY(G)<6 THEN EY(G)=EY(G)+1
3230 IF EY(G)>6 THEN EY(G)=EY(G)-1
3240 IF ET(5)=5 AND EY(5)=6 AND EX(5)=6
THEN GOSUB 2000
3250 IF EY(G)=6 AND EX(G)=6 THEN 9500
3265 IF G<5 AND EX(G)=EX(5) AND EY(G)=EY
(5) THEN ET(5)=0:EX(5)=0:EY(5)=0
3300 GOSUB 30:ED(G)=Z
3330 GOSUB 40:EH(G)=Z:IF EH(G)>99 THEN E
H(G)=99
3400 GOTO 3010
4000 ? "{UP}ENEMY FIRING & MOVING"
4010 FOR G=1 TO 4:IF ET(G)<>0 THEN 4100
4020 NEXT G:RETURN
4100 IF INT(99*RND(0)+1)>(EE(G)+INT(30*R
ND(0)+1)) OR EE(G)<10 THEN 4020
4110 E=E-INT(5*RND(0)+1)*ET(G)
4150 EE(G)=EE(G)-INT(10*RND(0)+1)
4160 IF RND(1)<0.1 THEN C=C-INT(25*RND(0
)+1):IF C<1 THEN 9500
4170 IF RND(1)<0.1 THEN M=M-INT(25*RND(0
)+1):IF M<0 THEN M=0
4180 IF RND(1)<0.1 THEN S=S-INT(25*RND(0
)+1):IF S<0 THEN S=0
4200 IF E<0 THEN 9500
4210 GOTO 4020
5000 G=INT(5*RND(0)+1)
5005 IF G=5 AND ET(5)=0 AND INT(4*RND(0)
+1)>1 THEN ET(5)=5:GOTO 5160
5010 IF G=5 OR ET(G)<>0 OR INT(9*RND(0)+
1)>6 THEN 5400
5110 A=4-INT(LOG(INT(50*RND(0)+1)+2))
5120 ET(G)=A:EE(G)=99
5160 EX(G)=INT(11*RND(0)+1)
5170 EY(G)=INT(11*RND(0)+1)
5180 A=INT(4*RND(0)+1):IF A=1 THEN EY(G)
=1
5190 IF A=2 THEN EY(G)=11
5200 IF A=3 THEN EX(G)=11
5210 IF A=4 THEN EX(G)=1
5300 GOSUB 30:ED(G)=Z
5320 GOSUB 40:EH(G)=Z:IF EH(G)>99 THEN E
H(G)=99
5400 RETURN
6000 ? :? :? "WEAPON:                   ":A=
0
6010 TRAP 6010:GET #1,K:TRAP 40000
6020 IF K=ASC("M") AND M>0 THEN A=6:M=M-
INT(5*RND(0)+1):IF M<0 THEN M=0
6025 IF K=ASC("C") THEN E=E+INT(20*RND(0
)+1):GOSUB 10800:IF E>99 THEN E=99
6030 IF K=ASC("C") THEN RETURN
6035 IF K=ASC("S") AND S>0 THEN A=4:S=S-
INT(5*RND(0)+1):IF S<0 THEN S=0
6040 IF K=ASC("T") AND T>0 THEN A=9:T=T-
1
6060 IF A<3 THEN ? "{UP}{BELL}BAD INPUT!
 WEAPON:":GOTO 6010
6100 ? "{UP}TARGET NO:          "
6120 TRAP 6120:GET #1,K:TRAP 40000
6125 B=K-48:B=B*(B>=1 AND B<=4)
6130 IF ET(B)=0 THEN ? "{UP}{BELL}BAD DA
TA! TARGET: ":GOTO 6120
6140 IF A=6 THEN GOSUB 10600
6150 IF A=4 THEN GOSUB 10500
6160 IF A=9 THEN GOSUB 10400
6200 IF INT(99*RND(0)+1)>EH(B) THEN ? "{
UP}MISSED!                  ":GOSUB 10100:
RETURN
6210 EE(B)=INT(EE(B)-((A*INT(15*RND(0)+1
))/ET(B)))
```

```
6215 ? "{UP}TARGET |HIT|           ":GOSU
B 10200
6220 IF EE(B)<1 THEN 6500
6230 E=E-INT(5*RND(0)+1)
6300 RETURN
6500 UP=UP+ET(B)
6505 EX(B)=0:EY(B)=0
6510 ET(B)=0:EH(B)=0:ED(B)=0:EE(B)=0
6570 ? "{UP}|  TARGET DESTROYED!  |":GOS
UB 10300
6580 RETURN
7000 REM LINES 7000-7060 ARE P/M GRAPHIC
S. DELETE FOR 8K MEMORY
7005 POKE 559,46:PMB=PEEK(106)-16:POKE 5
4279,PMB:PMB=PMB*256+512
7010 POKE 53248,0:POKE 53249,0
7020 POKE 704,192:POKE 705,192:P=47
7030 POKE 53256,3:POKE 53257,3
7035 FOR I=0 TO 63:POKE PMB+I,0:POKE PMB
+64+I,0:POKE PMB+128+I,0:POKE PMB+192+I,
0:NEXT I
7040 FOR I=0 TO 44:POKE PMB+P+I,255:POKE
 PMB+128+P+I,255:NEXT I
7050 POKE 53277,3:POKE 53248,55:POKE 532
49,68
7060 RETURN
8000 FOR I=0 TO 5:ET(I)=0:EH(I)=0:EX(I)=
0:EY(I)=0:ED(I)=0:EE(I)=0:NEXT I:RETURN

9500 GOSUB 10700
9510 ? "{CLEAR}{3 DOWN}You have been |DE
STROYED!|"
9520 POKE 53248,0:POKE 53249,0
9530 POKE 53277,0:POKE 559,34
9550 ? "{4 DOWN}   SCORE=";UP:? :?
9560 IF UP>HS THEN HS=UP
9580 ? "{2 DOWN}************************"

9590 ? "  High Score= ";HS
9600 ? "************************"
9605 ? "{4 DOWN} Another Game? ";
9630 TRAP 9630:GET #1,K:TRAP 40000
9640 IF K=ASC("Y") THEN RUN
9650 END
10000 REM SOUND SUBROUTINES
10100 REM MISSED!
10110 FOR W=100 TO 200 STEP 2:SOUND 0,W,
10,4:NEXT W:SOUND 0,0,0,0:RETURN
10200 REM HIT!
10210 FOR W=0 TO 30:SOUND 0,W*8,8,15-W/2
:SOUND 1,W*8,16,15-W/2:POKE 712,PEEK(537
70):NEXT W
10220 POKE 712,0:RETURN
10300 REM ENEMY DESTROYED!
10310 FOR W=0 TO 255 STEP 10
10320 POKE 704,PEEK(53770):POKE 705,PEEK
(704)
10330 SOUND 0,W,8,W/15:SOUND 1,PEEK(5377
0),0,15*RND(1)
10340 NEXT W:SOUND 0,0,0,0:SOUND 1,0,0,0

10350 POKE 704,192:POKE 705,192
10360 RETURN
10400 REM PHOTON TORPEDO
10410 FOR W=200 TO 100 STEP -2:SOUND 0,W
,2,4:NEXT W:FOR W=0 TO 100 STEP 6:SOUND
0,W,2,8:NEXT W
10420 SOUND 0,0,0,0:RETURN
10500 REM SECONDARY FIRE
10510 FOR I=1 TO 10:FOR W=0 TO 3:SOUND 0
,W,2,8:NEXT W:NEXT I:SOUND 0,0,0,0:RETUR
N
10600 REM MAIN FIRE
10610 FOR I=1 TO 10:FOR W=0 TO 3:SOUND 0
,W,2,8:SOUND 1,W*3,2,8:NEXT W:NEXT I:SOU
ND 0,0,0,0:SOUND 1,0,0,0:RETURN
10700 REM |YOUR| DEATH KNELL
10710 FOR W=0 TO 255 STEP 10
10720 POKE 710,PEEK(53770):POKE 712,PEEK
(53770)
10730 SOUND 0,W,0,15-W/17:SOUND 1,128+W/
2,8,8+7*RND(1)
10740 NEXT W:FOR W=0 TO 10:POKE 710,68:F
OR I=0 TO 5:NEXT I:POKE 710,0:NEXT W
10780 SOUND 0,0,0,0:SOUND 1,0,0,0
10790 SETCOLOR 2,9,4:SETCOLOR 4,0,0:RETU
RN
10800 REM RECHARGE
10810 FOR W=1 TO 20:SOUND 0,5+5*RND(0),1
0,8:NEXT W
10820 SOUND 0,0,0,0:RETURN
```

©

This issue we feature two articles on the increasingly popular "computer camp."

For a list of upcoming computer camps, see "Learning With Computers" elsewhere in this issue.

Guest Commentary:

Computer Camp

G. R. Boynton
University of Iowa
Iowa City, IO

It was the last day of our four weeks, and I had saved my sure winner for this day. It is a capital A which goes skipping across the screen from right to left. As soon as he saw it, Steve knew what he wanted and, with a little help, he produced a "banner" program which printed

››››**STEVE ON THE GO**›››

across the screen a thousand times. It is a relatively simple program, but Steve is only eleven. More important than how difficult or easy the program is what it says about Steve.

Last summer the Laboratory for Political Research at the University of Iowa ran a computer camp for four groups of seven junior high school students and a few, like Steve, who were younger and one or two who were in high school. For two years we have been busy installing microcomputers in the Lab and the department of political science. In the process we have purchased seven Commodore 2001s and 37 Commodore 8037s. The 2001s were used for program development before the 8032s were available and, when we got the 8032s the 2001s became surplus. I had never had a chance to work with junior high school students, and I wanted to see what that enthusiasm and energy was like. Hence, the computer camps.

Each group met for three hours a day for four weeks. My plan was very simple. Show them a lot of tricks that one can do with a PET. Have a lot of games that they can play. And get out of their way; turn them loose with a computer and see what happens. Cursor [*the monthly tape of programs from The Code Works*] very graciously permitted me to use their games (I received as a subscriber) in the computer camp on an experimental basis. I helped the campers write programs for between 30 minutes and an hour each day. We concentrated on relatively simple graphics programs because that is what they really liked. And then they were free to do what they wanted to do for the remainder of their three hours.

Literacy At A Low Price

Simple graphics programming has two advantages when working with junior high students. First, it motivates. They like making graphic displays, and that can be done rather easily on Commodore computers. Second, it eases the introduction to what are otherwise rather arcane subjects.

You have to learn something about variables and constants, strings and numbers, loops and conditionals to do any programming. But, if you are going to program graphics you also have to learn something about the difference between printing and POKEing to the screen. And this involves learning something about ASCII values for representing characters. And that this is only useful if you understand something about memory where the numbers are stored. This, in turn, leads immediately to a discussion of the memory map of the PET, keyboard buffers, and other esoterica. All these topics follow naturally, in the process of learning to put graphics displays on the screen and move them around. It's computer literacy at a very low price.

By the end of four weeks, most of the students had learned enough so that they could put ideas to work. That final program which Steve responded to is a simple idea. You print something, erase it, move, and print it again. There were also several rather nice applications of this idea. Josh produced a rocket which zoomed up the screen and then came down, landing on the moon. Gus printed his goodbye to his classmates by writing each line in a fancy box on the screen, erasing it, and then writing the next line.

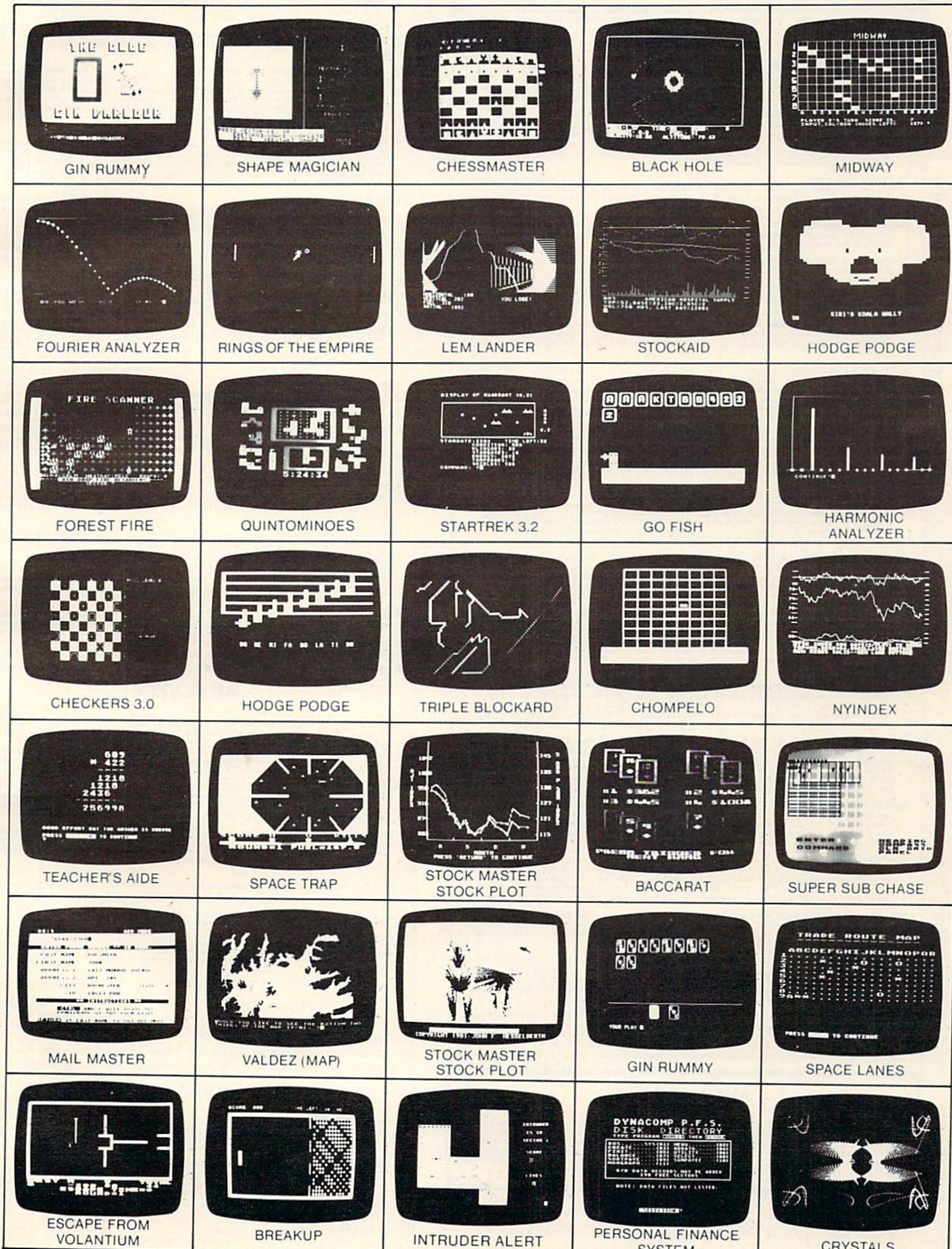
We did some other programming as well. We spent about a week building and using a subroutine that would break up a string into its constituent words. A fortune telling program resulted from that. And another program used the same subroutine to test knowledge of US and European capitols.

Programming is okay, but much of the appeal of the camp was the games. Cursor is a good collection of games of the most diverse types. Each student spent many hours playing these games. Fast action games were the most popular, but treasure hunt games, gambling games, and strategy games were popular as well.

My friends were having fun playing games and learning to write programs. And I learned some things about them.

One characteristic is that a majority of them charge ahead. Don't bother to read instructions.

It's hard to picture all of DYNACOMP's software



TURN TO NEXT PAGE....

BUSINESS and UTILITIES

NEW **MAILMASTER (Atari diskette only)** — **Price: $39.95 Diskette**

MAILMASTER is a very versatile software package for managing and manipulating mail lists and mini data bases. Each disk can hold over 600 customer entries containing name, address, 38 letter key words and a phone number. The display is marked so that entries may be made and edited with ease. The status (e.g., disk space left, options, etc.) is shown at all times. Labels may be printed 1,2 or 3 up, and all sorting (zip code and alphabetic) is performed by a fast machine language program.

PERSONAL FINANCE SYSTEM (Atari and North Star only) — **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.

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, commission 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.

TAX OPTIMIZER (North Star only) — **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!

NEW NEW **UTIL (Apple only, 48K)** — **Price: $19.95 Diskette**

UTIL is a disk-oriented utility system which permits examining and changing of the contents of DOS 3.2 and 3.3 diskettes at the bit (nibble or byte) level. With UTIL you can easily examine the contents of a diskette sector by sector, restructure the sector pointers, reallocate sectors (e.g. bad sectors may be "hidden"), and perform many other sophisticated operations. For the experienced programmer.

NEW **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.

NEW NEW **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 **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!

EDUCATION

HODGE PODGE (Apple only, 48K Applesoft or Integer BASIC) — **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.

NEW **TEACHERS' AIDE (Atari only)** — **Price: $13.95 Cassette/$17.95 Diskette**

TEACHERS' 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.

NEW **PHARMACOLOGY UPDATE (PET only)** — **Price: $169.95 Cassette/$149.95 Diskette**

This is DYNACOMP's first educational software entry for the medical profession (more are coming!). PHARMACOLOGY UPDATE was written by a R.N. as a masters project, with the aid of a practicing pharmacologist and an electronics instructor. This package comes in two parts. The first part is a 200 page manual which is divided into 11 sections. Each of these sections provides both concise information and probing questions. The second part consists of 11 programs that are keyed to the text and which test the degree of your understanding of the text material. This package has great educational value for the beginning student as well as the professional interested in an efficient way to review and update his or her knowledge. Available on cassettes (11) or diskette.

NEW **TEACHER'S GRADEBOOK (Apple 48K dual/single drive)** — **Price $49.95 Diskette**

TEACHER'S GRADEBOOK is a *complete* password protected record-keeping system for the classroom. It supports up to nine users, and each user may have data for up to nine classes on one disk (with up to 90 students per class). Typical information which can be entered, edited and processed includes rosters, absences and grades. Summary reports may be displayed (on the screen) or printed in various ways, with automatic weighted averaging and conversion to letter grades. This system has been tested ("goof-proofed") in the class environment and is both well-written and well-documented.

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. Dealer discount schedules are 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 and Osborne 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.

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 Cassette/$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 (eg: 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 (PARAFIT) (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 1. Use REGRESSION 1 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 I 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: $14.95 Cassette/$18.95 Diskette

All three collections are available for $39.95 (three cassettes) and $49.95 (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 I 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 and Dr. Dobbs.

NEW **SOFTNET (Apple II, 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 **MATCHNET (TRS-80 only)** — **Price: $19.95 Cassette/$23.95 Diskette**

It often takes days to iteratively optimize an L, Pi or T matching network for a particular application. Take a few minutes with MATCHNET and you will have the Q, frequency response and reflection coefficients for any of twelve matching networks. You input the source and load impedance and MATCHNET calculates the R, C, (and L) values and plots (and/or prints) the frequency response and reflection coefficients for each configuration. The reviewer of this program remembers when he use to do this by hand and loves MATCHNET!

ACTIVE CIRCUIT ANALYSIS (ACAP) (48K Apple only) — **Price: $25.95 Cassette/$29.95 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.

LOGIC SIMULATOR (Apple only; 48K RAM) — **Price: $24.95 Cassette/$28.95 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 using HIRES graphics. Save your breadboarding until the circuit is checked by LOGIC SIMULATOR.

NEW **NUMBERKRUNCHER (TRS-80 only)** — **Price: $69.95 Cassette/$73.95 Diskette**

This program is the most complete numerical analysis system available for the TRS-80. It can handle up to 255 data sets, each set having a six character name. It includes complete data editing facilities and convenient data input/output capability. The analyses available are multiple linear regression and correlation determination of residuals, data transformations and extensive graphics generation, including axis naming, and more. The supporting documentation is extremely well written and well organized, and includes appendices which describe the numerical procedures used in the program.

NEW **STATSORT (TRS-80 only)** — **Price: $39.95 Cassette/$43.95 Diskette**

STATSORT consists of several menu selected programs which allow the user to create (build, edit, merge), format and print files, (machine) sort them on any field, and numerically analyze (maximum, minimum, average, variance, standard deviation) tabulated data. STATSORT is well documented and easy to use. The cassette version can also be employed to create a data type which can be read by the Radio Shack Advanced Statistical Package.

NEW **STATTEST (TRS-80 only)** — **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.

Don't bother to plan very much. Just go. The book on BASIC that they bought did not get much of a workout.

They like some very simple things. Many of the boys engaged in a very short "insult" program.

```
10 print "garbage head";
20 goto 10
```

That will print garbage head continuously across the screen until you hit the STOP key. It even has a certain graphic appeal which grows out of the normal flow of the program.

They like simple graphics. They are very enthusiastic about games. They learn to program "in use." One of my colleagues noted that his son was learning to program more like learning a language by living in another country than the way languages are learned in schools. He could *do* it even though he did not find it easy to talk about it.

Games and programming were going on in these computer camps. But something else was going on that I had not fully expected. Persons between the age of eleven and fifteen were busy exploring and fleshing out their "self." Steve is an eleven-year-old who is on the go. It shows up in everything he does; including his banner program.

However, in his crashing ahead he never managed to produce anything neat. Gus is different. What he managed on the last day was a very aesthetically appealing display. That is an important difference between Gus and Steve. I could see the same thing going on in each of these young individuals. They were defining themselves in what they did with the computer. And the computer is flexible enough to permit this form of self-expression.

One more thing came out of this camp. There are now 28 more people for whom the computer will be understood as a personal tool.

It isn't necessary to understand machine language to add this useful search utility to your library of programs. Just type in the BASIC loader program and it will build the machine language routine for you. There are versions here for Apple II Plus, and both Upgrade and 4.0 PET/CBM BASICs.

Search For PET And Apple II Plus

Michael Erperstorfer
Vienna, Austria

Here is a useful utility program, Search, which enables you to find any string or number of BASIC keyword within a BASIC program. For example, if you've written a large program and want to find out all the places where a variable, NAME$, appears, or all examples of GOSUB – use Search. It will print out all line numbers where it finds the target.

To start the search, you type in a new BASIC line at line zero and follow it with a colon and the target of your search. To look for NAME$:

0:NAME$

To be able to look for numbers, the first character of line zero is ignored (that's why the colon is neces sary. To look for the number 102, you would type:

0:102

Program 1 is for the Apple II Plus. You can type the & key and hit RETURN because this is easier than typing CALL 768 every time you want to initiate a search. (The machine language routine must be linked at first, before any searches, with CALL 768.)

For PET/CBM 4.0 BASIC users, Program 2 will create the machine language routine which can then be used by typing in SYS 864 and hitting RETURN. PET/CBM Upgrade BASIC users should make the change to line 972 as indicated in Program 3.

Program 1.

```
10 REM FIND FOR APPLE II PLUS
700 FOR ADRES=768TO900:READ DATTA:POK
    E ADRES,DATTA:NEXT ADRES
768 DATA 169, 76, 141, 245, 3, 169
774 DATA 16, 141, 246, 3, 169, 3
780 DATA 141, 247, 3, 96, 162, 0
786 DATA 173, 1, 8, 133, 1, 173
792 DATA 2, 8, 133, 2, 160, 0
798 DATA 177, 1, 208, 6, 200, 177
804 DATA 1, 208, 1, 96, 160, 0
810 DATA 177, 1, 133, 3, 200, 177
816 DATA 1, 133, 4, 200, 177, 1
822 DATA 133, 117, 200, 177, 1, 133
828 DATA 118, 165, 1, 24, 105, 4
834 DATA 133, 1, 165, 2, 105, 0
840 DATA 133, 2, 160, 0, 177, 1
846 DATA 240, 28, 205, 6, 8, 240
852 DATA 4, 200, 76, 196, 3, 162
858 DATA 0, 232, 200, 189, 6, 8
864 DATA 240, 7, 209, 1, 240, 245
870 DATA 76, 76, 3, 32, 119, 3
876 DATA 165, 3, 133, 1, 165, 4
882 DATA 133, 2, 76, 28, 3, 169
888 DATA 163, 32, 253, 251, 32, 32
894 DATA 237, 169, 160, 32, 253, 251
900 DATA 96
```

Program 2.

```
10 REM FIND FOR 4.0 BASIC
800 FOR ADRES=864TO980:READ DATTA:POK
    E ADRES,DATTA:NEXT ADRES
864 DATA 162, 0, 173, 1, 4, 133
870 DATA 193, 173, 2, 4, 133, 194
876 DATA 160, 0, 177, 193, 208, 6
882 DATA 200, 177, 193, 208, 1, 96
888 DATA 160, 0, 177, 193, 133, 195
894 DATA 200, 177, 193, 133, 196, 200
900 DATA 177, 193, 133, 54, 200, 177
906 DATA 193, 133, 55, 165, 193, 24
912 DATA 105, 4, 133, 193, 165, 194
918 DATA 105, 0, 133, 194, 160, 0
924 DATA 177, 193, 240, 28, 205, 6
930 DATA 4, 240, 4, 200, 76, 156
936 DATA 3, 162, 0, 232, 200, 189
942 DATA 6, 4, 240, 7, 209, 193
948 DATA 240, 245, 76, 156, 3, 32
954 DATA 199, 3, 165, 195, 133, 193
960 DATA 165, 196, 133, 194, 76, 108
966 DATA 3, 169, 35, 32, 210, 255
972 DATA 32, 127, 207, 169, 32, 32
978 DATA 210, 255, 96
```

Program 3.

```
972 DATA 32, 213, 220, 169, 32, 32
```

©


computer
CBM™
Model 8032
commodore
NEECO
JUST A SAMPLE OF THE MANY PRODUCTS WE CARRY, CALL US FOR OUR NEW 80-PAGE CATALOG.
WE WILL MATCH SOME ADVERTISED PRICES ON CERTAIN PRODUCTS LISTED UNDER SIMILAR "IN STOCK" CONDITIONS.

Numerous ideas come to mind, especially for educational applications, when a VIC can be used to control a video disc machine.

A VIC Intelligent Video Disc System

Claire J. Carr and Everett Q. Carr
Herkimer, NY

The Commodore VIC-20 can control the Pioneer VP 1000 LaserDisc. What it takes is an adapter circuit attached to the VIC User Port. By adding a printed circuit board to the adapter it can be controlled by a PET computer. Adding the software and a suitable laser disc results in an "intelligent" video disc system. This new teaching tool is part of the information technology revolution that many predict will change schools radically during the next generation. We wanted to explore first-hand what could be done with an intelligent video disc system, possibly an important tool in the future of education.

As teachers, our major interest is to improve the art of teaching for the benefit of students. In an ordinary class, the range of student capability can be equal to the grade number. A fourth grade class can therefore contain students with a reading capability ranging from 1st to 8th grade level. In a 20 pupil class, students receive actual individual attention which amounts to only minutes a day. They get few trials at mastery of any topic. The wide spread in classroom capability, cultural biases of both city and rural schools, and the competition of TV can each damage a teacher's efforts to provide an atmosphere which motivates students.

Also, outstanding teaching is rarely transmitted. Few teachers have the resources to communicate beyond a local area. The use of an intelligent disc could lead to improved education. It actually leverages teacher time and permits an increase in contacts with students. This system can improve the accuracy of presentation and increase, by orders of magnitude, the number of possible trials a student encounters in achieving subject mastery. The system also requires student involvement and, with properly designed software, supplies immediate feedback to correct errors and speed mastery.

We had already built an adapter to connect between the PET 2000 (32K) and the Pioneer VP1000 LaserDisc video player.

What is significant about the video disc? It is the storage capacity, up to 54,000 individual picture frames on a half hour side of a LaserDisc. Each frame is numbered, encoded in the 17 spare lines of the TV frame. The Pioneer LaserDisc decodes the frame number and displays it on the TV screen on command. There are extended play versions of LaserDisc's operating at double the track density. Unfortunately, they skip putting the frame numbers on double density discs. One reason, it appears, is that, in one hour per side extended play made, the disc plays at constant linear velocity. The rate of disc rotation changes from 1800 RPM on the inner track to 600 RPM on the outer disc track. In the single play mode there is constant angular velocity with the disc held at 1800RPM.

It is straightforward to interface the VIC-20 to the Pioneer, VP1000 LaserDisc player. A block diagram of the VIC Intelligent Video Disc System is shown in Figure 1. The switch box shown in Figure 1 allows the use of a single TV monitor for the VIC and the Pioneer player. The switch box contains RF connectors for the computer, disc, and TV. We used the inexpensive Type F connectors.

Our VIC-to-Pioneer Adapter was built by wire wrap on a Radio Shack 0.1 inch grid printed circuit board. This board has contact fingers on 0.156 inch centers along one edge. A 12/24 contact edge connector can be soldered along this edge and the adapter is then a plug-in for the VIC User Port. We actually cut a Radio Shack 20/40 pin connector down to size for the VIC.

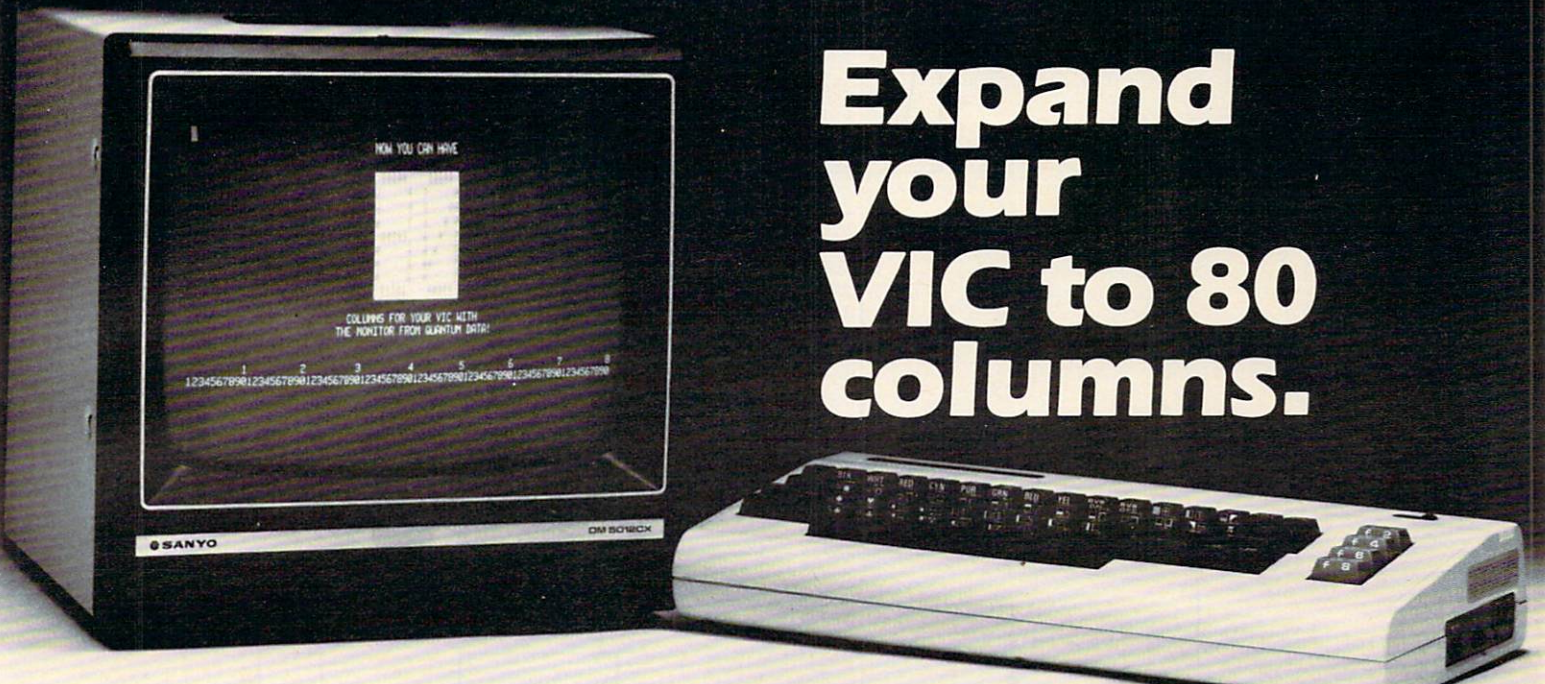
IC-1 in the Adapter produces a 38 kilohertz pulse chain containing the encoded signals for each of the operations of the video disc player.

A data pulse string is 10 cycles of a 38 Kilohertz clock that is generated from the piezo-ceramic chip, XI, Pioneer Electronics part number VSS-002. A code word is 10 bits long and is contained within 11 actual data pulses. It is the period between the 38 kilohertz data pulses which determine the logic code. A logical zero is a pulse period of 0.93 millisecond. A logical one is twice that period, 1.86 milliseconds. Five bits of the ten encoded are fixed, leaving five bits for up to 32 commands to the LaserDisc.

The code selection is determined by the combination of lines effectively pulled to ground by the chips IC-1 and IC-2, the CD4051B. These are CMOS analog switches with a low saturation resistance when switched by the control lines. The "B" after the device number 4051 means that the devices are buffered, increasing their resistance to (but not

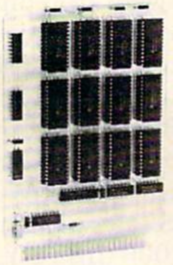
22-40-80 HIKE!

Expand your VIC to 80 columns.



Quantum Data's new Video Combo Cartridge brings you: 40 or 80 column display, plus 16K RAM and PROM socket. $299.00

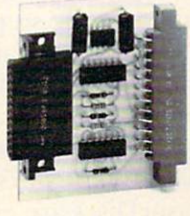
With the Video Combo Cartridge from Quantum Data you can now have 40 or 80 column display, 16K RAM and PROM all in one cartridge. It comes set for 40 column Display compatible with the VIC video modulator and your home T.V. Then, when you are ready to upgrade to 80 columns and a video monitor, just make a simple, no-cost change inside the cartridge. Instructions are provided. Also provided is a socket for a PROM, 16K of memory and AC adaptor. If you don't need memory, then 80 columns can be yours for only $199.50. A listing of the driver software is provided at no charge. A programmed PROM containing this software is also available for $19.95.



QDI expandor:

- Expands Basic user memory up to 24K in 8K steps
- PROMS may be mixed with RAM in 8K blocks
- 8K can be assigned to machine language area
- Plugs directly into VIC expansion port
- Low power, no additional power supply required
- Professional Quality, full buffering on all signals
- Small size: 6 x 4.5 inches.

16K expandor. $149.95
24K expandor. $199.95



QDI Printor RS-232 interface:

- Provides RS-232 voltage conversion for VIC serial port
- Allows use of a wide variety of RS-232 peripherals including printers, modems and voice synthesizers
- Low power CMOS circuitry requires no external power supply
- Small size: 2½ x 3 inches

Printor. $49.95

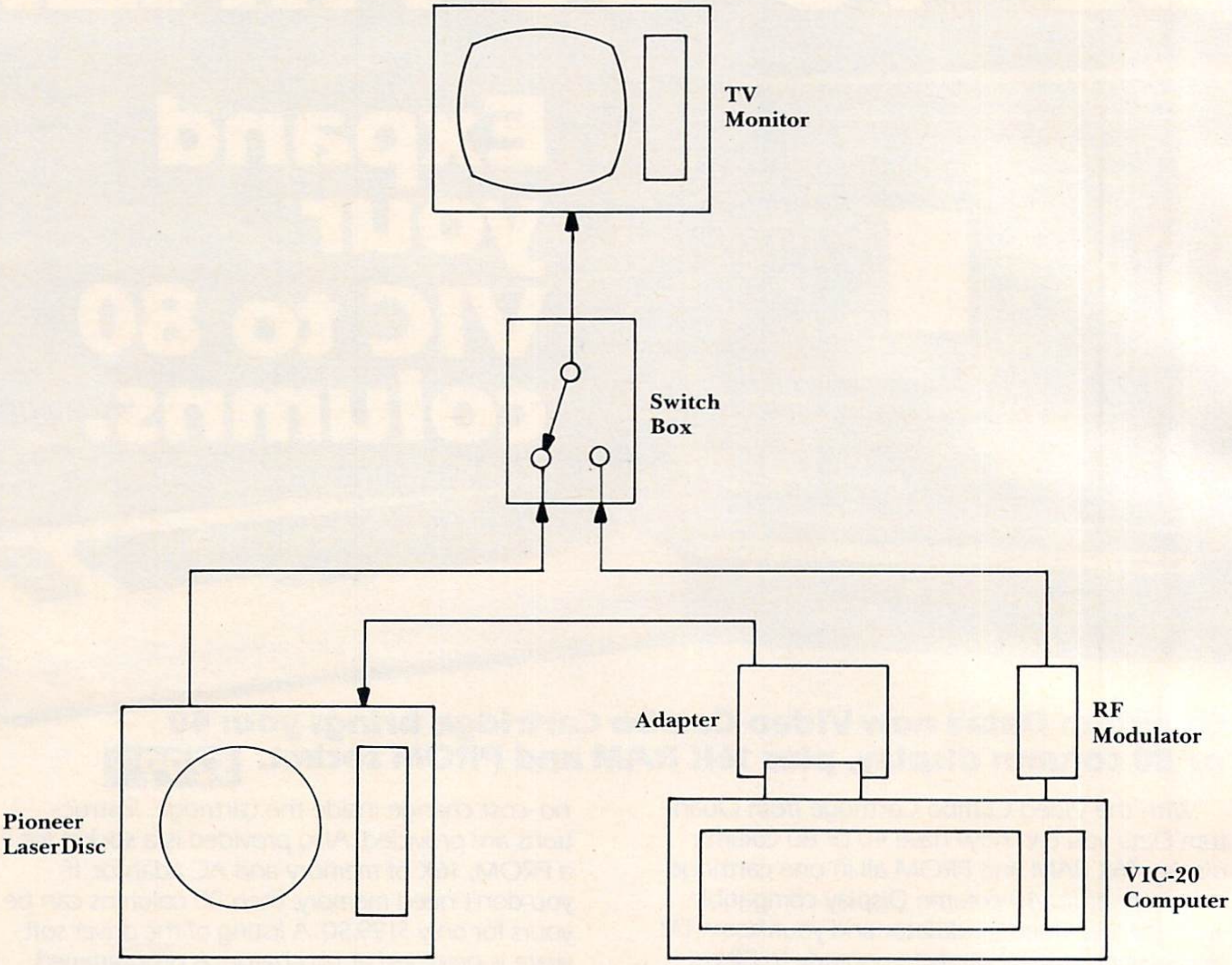
QDI Minimother:

- Adds 3 slots to the memory expansion port
- Removable card guides allow either boards or cartridges
- Requires no additional power supply
- Fused to protect VIC power supply from overload
- Simple plug-in installation

Minimother. $69.95

QUANTUM DATA, INC.
(714) 966-6553
DEALER HOT LINE (714) 754-1945
3001 Redhill Bldg. 4, Suite 105, Costa Mesa, CA 92626

Figure 1: VIC's Intelligent Video Disc System



totally protecting against) electrostatic charge and voltage transients.

The CD4051s are switched by the output from the 6522 VIA chip in the VIC computer by way of the User Port terminals C thru K. The transistors Q1 and Q2 buffer IC-1, provide pulse inversion and drive for the 30 feet of audio cable connected to the LaserDisc player. The LED, D1, also blinks out the coded pulse string indicating that the data pulse chains are being sent by the Adapter.

The LaserDisc player controls are a fantasy machine. Imagine any way you would want to manipulate a motion picture and the LaserDisc has a control key to do it. Here is a partial list:

CONTROL KEY		FUNCTION
SEARCH	(S)	Sets up player to search for picture frame and initiates search.
DIGITS	"0 - 9"	Enter frame number 0-54000.
PAUSE	(P)	Stops picture. Blanks screen.
PLAY	(G)	Starts picture.
FRAME	(F)	Displays or removes frame number.
STILL	(H)	Stops picture on frame number.

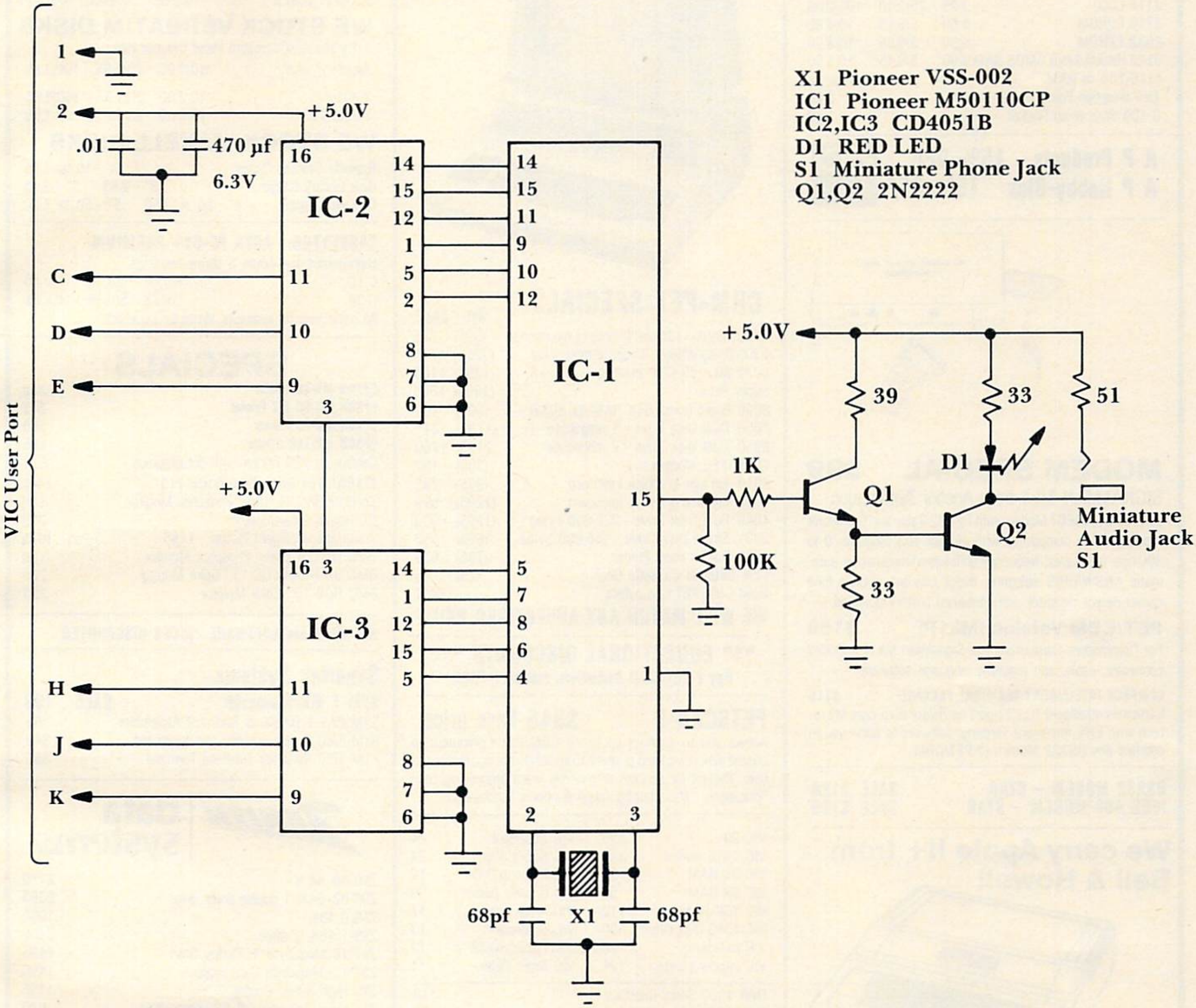
There are other functions as well: switching either of the audio channels, forward or reverse on single frame step, slow step, 3X scan and fast scan.

The program for the VIC-20 is called a "driver" and contains a simple routine in BASIC to control the LaserDisc player. You can build routines around it to make an Intelligent Video Disc system. With over 400 dealers already in the US sales of Pioneer LaserDiscs are estimated near 40,000 this year alone. Discounts on the $745 price are available, too.

Be forewarned. Dealers are having a difficult job keeping shelves stocked with discs. We bought 20 discs and the dealer said that that's about average.

```
1 REM VIC-DRIVE FOR LASER DISC"
10 D9=150
15 DIMTX(15)
20 GOSUB12000
30 INPUT"COMMAND STRING-->";CS$
40 PRINT
```

Figure 2: VIC To Pioneer Video Disc Adapter

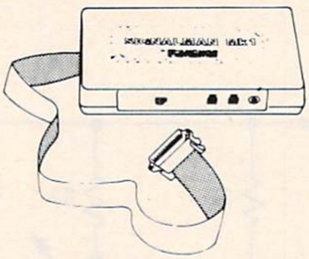



```
45 GOSUB10000
50 PRINT:PRINT
60 GOTO30
10000 FORQ1=1 TO LEN(CS$)
10010 Q2=ASC(MID$(CS$,Q1)):GOSUB110
    00
10020 NEXT
10030 RETURN
11000 IFQ2>57 OR Q2<48 THEN 11020
11010 Q3=TX(Q2-47):GOTO 11060
11020 IFQ2>72 OR Q2<70 THEN 11040
11030 Q3=TX(Q2-59):GOTO11060
11040 Q3=0
11045 IF Q2=80 THEN Q3=TX(14)
11050 IF Q2=83 THEN Q3=TX(15)
11060 PRINT CHR$(Q2);:POKE 37136,Q3
11070 FOR J= 1 TO D9: NEXT
11080 POKE 36136,0
11090 FOR J= 1 TO D9:NEXT
11100 RETURN
12000 DATA 86,22,38,70,54,82,18,34,
    66
12010 DATA 50,81,52,69,53,83
12020 FOR J=1 TO 15:READTX(J):NEXT
12030 POKE 37138,255
12040 RETURN
13010 REM *** COMMAND SUMMARY ***
13020 REM NUMBERS "0" TO "9"
13030 REM SEARCH   "S"
13040 REM PAUSE    "P"
13050 REM PLAY     "G" (FOR "GO")
13060 REM STILL    "H" (FOR "HALT")©
```

SPECIALS on INTEGRATED 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
2114-L300		2.25	25/2.10	100/2.00
2716 EPROM		5.50	5/5.15	10/4.80
2532 EPROM		9.90	5/9.45	10/8.90
6116 Hitachi 2Kx8 CMOS RAM		9.90	5/9.45	10/8.90
4116-200 ns RAM				8 for 15
Zero Insertion Force 24 pin Socket				2.00
S-100 Wire Wrap Socket				2.40

A P Products 15% OFF
A P Hobby-Blox 15% OFF



MODEM SPECIAL $99

SIGNALMAN Mk1 from Anchor Automation

DIRECT CONNECT Modem with RS232 Cable and Connector included. Fully compatible with all Bell 103 modems. 0 to 300 bps, full duplex, frequency shift keyed modulation, automatic ANSW/ORIG selection, direct connect, audible tone carrier detect indicator, self-contained battery powered.

PET/CBM Version (Mk1P) $169
For Commodore Computers, the Signalman Mk1P includes connector, cable, and machine language software

COMPACK INTELLIGENT TERMINAL PACKAGE $115
Combines intelligent RS232 port hardware from cgrs Microtech with EHS Intelligent Terminal software to allow you to connect any RS232 Modem to PET/CBM.

RS232 MODEM – STAR SALE $128
IEEE 488 MODEM – STAR SALE $199

We carry Apple II+ from Bell & Howell

16K RAM Card for Apple 95

fullFORTH+ for Apple II $75
A full-featured FORTH with enhancements. Conforms to FORTH Interest Group standards.

REVERSAL (Spracklen) 32K Apple	28.00
Data Manager (Lutus) 24K Apple	40.00
Data-Graph (Boyd) 48K Apple	40.00
Apple II User's Guide (Osborne)	12.00
Introduction to Pascal (Sybex)	10.30
Pascal Handbook (Sybex)	12.00
Musical Applications of Micros (Chamberlin)	20.00
User Guide to the Unix System	13
6502 Assembly Language Subroutines	11
PET Fun and Games	8

commodore

CBM-PET SPECIALS

	list	SALE
8023 Printer – 136 col, 150cps bi-directional	(995)	775
8300 Daisy Wheel – 40 cps bi-directional	(2250)	1750
8032 80 x 25 CRT, business keyboard	(1495)	1100
Super Pet	(1995)	1600
8096 Board (extra 64K RAM for 8032)	(500)	400
8050 Dual Disk Drive - 1 megabyte	(1795)	1345
8250 Dual Disk Drive - 2 megabyte	(2195)	1760
CBM IEEE Modem	(395)	199
4016 full size graphics keyboard	(995)	795
4032 full size graphics keyboard	(1295)	999
4040 Dual Disk Drive - 330,000 bytes	(1295)	999
2031 Single Disk Drive - 165,000 bytes	(695)	550
4022 Tractor Feed Printer	(795)	630
C2N External Cassette Deck	(75)	65
Used CBM/PET Computers		CALL

WE WILL MATCH ANY ADVERTISED PRICE

***** EDUCATIONAL DISCOUNTS *****
Buy 2 PET/CBM Computers, receive 1 FREE

PETSCAN I $345 base price.
Allows you to connect up to 20 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 3 computers.

VIC 20	259	VIC Superslot	24
VIC 1515 Printer	335	VIC Super Alien	24
VIC 3K RAM	36	VIC Maze in 3-D	12
VIC 8K RAM	54	VIC Cosmic Debris	9
VIC 16K RAM	112	VIC Amok (UMI)	17
VIC 1540 Disk Drive	500	VIC Snakman	13
VIC Invaders	24	VIC Rubik's Cube	13
VIC Jupiter Lander	24	VIC Night Rider	11

TNW 1000 Serial Interface	110
TNW 488/103 with DAA	450
Compute!'s First Book of PET /CBM	11
WordPro 3 Plus - 32K CBM, disk, printer	195
WordPro 4 Plus - 8032, disk, printer	300
VISICALC for PET, ATARI, or APPLE	190
SM-KIT - PET 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	8
IEEE-Parallel Printer Interface for PET	110
IEEE-RS232 Printer Interface for PET	120
The PET Revealed	12
Library of PET Subroutines	12

4 PART HARMONY MUSIC SYSTEM for PET
The Visible Music Monitor, by Frank Levinson, allows you to easily enter, display, edit, and play 4 part harmony music. Includes whole notes thru 64ths (with dotted and triplets), tempo change, key signature, transpose, etc. The KL-4M unit includes D to A converter and amplifier ready to hook to your speaker.
KL-4M Music Board with VMM Program $60

DISK SPECIALS

SCOTCH (3M) 5"	10/2.45	50/2.35	100/2.30
SCOTCH (3M) 8"	10/2.60	50/2.45	100/2.40

WE STOCK VERBATIM DISKS
Try the new Verbatim Head Cleaner Kits

BASF 5" or 8"	10/2.00	20/1.95	100/1.85
Wabash 5"	10/2.00	20/1.95	100/1.85
Wabash 8"	10/2.00	20/1.95	100/1.85

WE STOCK MAXELL DISKS

Diskette Storage Pages		10 for 3.95
Disk Library Cases	8" - 2.85	5" - 2.15
Disk Hub Rings	8" - 50 @ 7.50	5" - 50 @ 6.00

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

C-10	10/.56	50/.50	100/.48
C-30	10/.73	50/.68	100/.66

All other lengths available. Write for price list.

SPECIALS

EPSON MX-80 Printer	450
EPSON MX-80 F/T Printer	545
EPSON MX-70 Printer	300
EPSON MX-100 Printer	695
Centronics 739 Printer with dot graphics	575
STARWRITER Daisy Wheel Printer F10	1445
Zenith ZVM-121 Green Phosphor Monitor	119
DC Hayes Smartmodem	230
Watanabe Intelligent Plotter 1195 6-Pen	1445
BMC BM12A Green Phosphor Monitor	100
BMC BM1400BLUC 13" Color Monitor	285
BMC RGB 13" Color Monitor	329

ALL BOOK and SOFTWARE PRICES DISCOUNTED

Synertek Systems

SYM-1 Microcomputer SALE	**199**
SYM BAS-1 BASIC or RAE 1/2 Assembler	85
KTM-2/80 Synertek Video and Keyboard	349
KTM-3/80 Synertek Tubeless Terminal	385

ZENITH data systems

Z90-80 64 K	2170
Z90-82 64K, 1 double dens. drive	2395
Z89-0 48K	1950
Z89-1 48K, 1 drive	2150
Z67 10 Megabyte + Floppy Drive	4495
Z37 1.3 Megabyte Dual Floppy	1495
Z25 High Speed Printer	1195
Z19 Video Terminal (VT-52 compatible)	670
ZVM-121 Green Phosphor Monitor	119
All Zenith Software discounted	

ATARI SPECIALS

800 Computer	675	Microsoft BASIC	72
400 - 16K	330	**MISSILE COMMAND**	32
810 Disk Drive	440	**ASTEROIDS**	32
825 Printer	629	**STAR RAIDERS**	37
850 Interface	170	Space Invaders	32
Inside Atari DOS	18	Music Composer	45
Paddle Pair	19	Chess	30
Joystick Pair	19	Super Breakout	30
16K RAM	69	**PAC-MAN**	36
32K RAM	99	**CENTIPEDE**	36
Pilot	68	First Book of Atari	11

Write for prices on other Atari items.

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.

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, Relocater, 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	$85
RAM/ROM - - 8K	120
Battery Backup Option	30

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 (disk or tape)	$ 40
SuperGraphics in ROM	55

Volume discounts available on ROM version for schools.

FLEX-FILE for PET/CBM Computers

Self Calculating
DATA BASE
REPORT WRITER
MAILING LIST

FLEX-FILE is a set of flexible, friendly programs to allow you to set up and maintain a data base. Print files with a versatile Report Writer or a Mail Label routine. Programmers will find it easy to add subroutines to their own programs to make use of Data Base files.

RANDOM ACCESS DATA BASE

Record size limit is 250 characters. The number of records per disk is limited only by the size of each record and the number of records per disk is limited only by the size of each record and the amount of 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. The Find command locates any record when you enter all (or a portion of) the desired key. Field lengths may vary from record to record to allow maximum packing of information. Files may be sorted by any field, and any field may be specified as a key. Sequential files from other programs may be converted to Flex-File format, and Flex-File records may be converted to sequential (WordPro, PaperMate, other word processors may also use Flex-File data). Maximum record size, fields per record, and order of fields may be changed at any time.

MAILING LABELS

With typical record size of 127 characters, each disk can handle over 1000 records (about 2800 with 8050 drive). 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 two or three fields may be joined together on one line (like first name, last name, and title). A "type of customer" field allows selective printing.

REPORT WRITER

Print any field in any column. For numeric fields, use decimal point justification (and round to any accuracy). Define any column as a series of mathematical functions performed on other columns. These functions include arithmetic operations and various log and trig functions. Pass results of operations such as running total from row to row. At the end of the report, print total and/or average for any column. Complete record selection, including field within range, pattern match, and logical functions can be specified individually or in combination with other parameters.

FLEX-FILE by Michael Riley — $60

Please specify equipment configuration when ordering.

COLOR CHART Color Video Board $125

Up to 8 colors and high resolution graphics (128 x 192 in 2 colors). Generates composite video for use with monitor.

PEDISK II from cgrs Microtech

5" 40 track, 1 drive, 143K	$525
5" 40 track, 1 drive, 286K	690
8" IBM 3740 format, 77 track, 250K	995

PROGRAM YOUR OWN EPROMS

Branding Iron for PET/CBM — $79

EPROM Programmer with software for all ROM versions. Includes all hardware and software to program or copy 2716 and 2532 EPROMs.

CBM Software

Legal Time Accounting Package	445
Medical Accounting Package	
PORTMAKER DUAL RS232 SERIAL PORT	63

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

Dow Jones Portfolio Management	129
Personal Tax Calculator	65
Wordcraft 80 Wordprocessor Package	325
Pascal Development Package	235
Assembler Development Package	99
Intelligent Terminal Emulator	30
Softpac-1 (Competitive Software)	29

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.

Available soon:

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. 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.

For writing text, Paper-Mate has a definable keyboard so you can use 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 16/32K CBM/PET machines, with any printer, and with either cassette or disk.

To order Paper-Mate, please specify configuration.

Paper-Mate on disk or tape — 40.00

BASIC INTERPRETER — $200

Designed to support the CBM 8096 (8032 with add-on 64K board). A full interpreter implementation to automatically take advantage of the extra memory available to the 8032.

BPI General Ledger – 8032/8050	300
BPI Accounts Receivable – 8032/8050	300

Hayden Software

Complex Mathematics	12.70
Engineering Mathematics	12.70
General Mathematics	12.70
MCAP:Circuit Analysis Program	21.00
Energy Miser	24.50

CASH MANAGEMENT SYSTEM — $45

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

MICRO-REVERSI for PET by Michael Riley — **10**
super machine language version of Othello

Tunnel Vision / Kat & Mouse by Michael Riley — **10**
two excellent machine language maze programs

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.

Some Similarities Between Applesoft And PET BASIC

Garry Kiziak
Burlington, Ontario

I wonder how often a PET owner will pick up a magazine article and seeing that it applies to the Apple say "Oh! This isn't of any use to me.". Similarly, how many Apple owners will pick up a PET article and say the same thing? In fact, there is much to be learned about the PET from Apple articles and vice-versa. In many cases translating a useful utility or idea from the one computer to the other simply involves chaining a few addresses or some simple modification.

For example, a very useful idea appeared in Volume 3 Issue #1 of The Transactor dealing with the idea of reading data from a particular DATA statement. The following short PET program demonstrates how it works.

```
10 DATA FIRST, SECOND, THIRD
20 DATA FOURTH
30 READ A$, B$
40 POKE 62, PEEK(119) : POKE63, PEEK(120)
50 READ A,B
60 DATA 1, 2, 3, 4
70 PRINT A$, B$, A, B
```

In this program, line 40 causes the READ statement in line 50 to get its data from the next DATA statement (i.e. line 60) instead of from the next data item which would have been in line 10.

To get this program to work on an Apple, it is only necessary to change the addresses in line 40. The following will do the trick.

```
40 POKE 125, PEEK(184) : POKE 126, PEEK(185)
```

In **COMPUTE!**, May, 1981, #12, Craig Peterson offered an elegant routine for the Apple that would allow you to input anything into a string (including commas, colons, etc.) without getting the EXTRA IGNORED error message. That routine is reproduced here.

```
1000 CALL 54572
1010 FOR B=512 TO 751 : IF PEEK(B) <> 0 THEN
     NEXT
1020 IN$="" : POKE PEEK(131)+256*PEEK(132)+
     1,0 : POKE PEEK(131)+256*PEEK(132+2,2 :
     POKE PEEK(131)+256*PEEK(132),B-512 : IN$
     = MID$(IN$,1) : RETURN
```

Once again, to get this to work on the PET, certain addresses will have to be changed. The necessary changes are as follows:

```
1000 SYS 48117
1010 FOR B=512 TO 592 : IF PEEK(B) <> 0 THEN
     NEXT
1020 IN$="" : POKE PEEK(68)+256*PEEK(69)+
     1,0 : POKE PEEK(68)+256*PEEK(69)+2,2
1030 POKE PEEK(68)+256*PEEK(69),B-512 : IN$
     = MID$(IN$,1) : RETURN
```

Notice that line 1020 had to be split into two lines. This is because the maximum length of a line in PET BASIC (including line numbers) is 80 characters. On the Apple, it is 239 characters.

The use of this subroutine instead of the traditional INPUT statement on the PET has an additional advantage – an empty response (i.e. simply pressing <RETURN>) does not break out of the program. Instead, the program continues and IN$ = "".

The above routine will prompt with a question mark and then a flashing cursor just like the regular INPUT statement. If you would rather that the question mark did not appear, then simply change line 1000 to:

```
1000 SYS 46306
```

Of course, not all programs will be as easy to change, but many will – even machine language programs. The trick is to find the correct change of addresses.

Recently, while writing a program for the Apple to draw the graph of practically any curve, I found it necessary to write a short machine language program which would change a line in the program to whatever you wanted. Specifically, it would be used to enter the equation of the curve to be graphed without going through the process of stopping the program, entering a new line, then typing in GOTO 550 or some similar process. This CHANGE routine would allow you to enter the equation just like in a regular INPUT statement. An assembly listing of the program is given below (Program 1). Here is an example of a machine language program that can be easily modified for the PET. The modified assembly listing is given in Program 2. Notice that the only changes required are in the addresses to the external ROM routines or zero page locations and to the IOSAVE and IORESTore routines of the Apple which had to be simulated on the PET.

To illustrate how this routine works on the

PET, get into the monitor (i.e. SYS 4) and type in .M 033A 0399 . Then use the cursor to edit the displayed screen as indicated in the memory dump in Program 3. Get back into BASIC and type in the short program in Program 4. RUN it, type in anything that you want, and watch line 100 change. RUN it again as often as you like and line 100 changes to whatever you dictate.

Now RUN the program again, but this time type in X=T:Y=SIN(T) in response to the prompt. Notice the momentary display of the message EXTRA IGNORED. In the listing you will see that line 100 has been changed correctly, but IN$ is only equal to X=T. If you happen to need the correct value of IN$ later on in the program, this would be totally unacceptable.

Now type in Program 5, RUN it, and type in your responses as before. This time, when you type in X=T:Y=SIN(T), you will notice that line 100 gets changed correctly as does variable IN$.

A good question would be: "How do you know what to change the various addresses to?". Personally, I have found two excellent sources. The first is the article "Applesoft Internal Entry Points" which appeared in the original *Apple Orchard*. The second source is the *PET/CBM Personal Computer Guide* (second edition). Pages 476-493 have Hex Addresses and Label References for most of the zero page addresses and ROM routines in the new BASIC 4.0 (and BASIC 3.0) ROMs. Even most of the names of the routines from these two sources are the same. So, the next time you see an Apple article or a PET article, don't put it off as not applying to you, make it work for you and learn by the experiences of others.

Program 1.

```
SOURCE FILE: CHANGE - APPLE
----- NEXT OBJECT FILE NAME IS CHANGE -
      APPLE.OBJ0
0341:                   ORG  $341
009B:          LOWTR    EQU  $9B
00B8:          TXTPTR   EQU  $B8
DEBE:          CHKCOM   EQU  $DEBE
DD7B:          FRMEVL   EQU  $DD7B
E752:          GETADR   EQU  $E752
FF4A:          IOSAVE   EQU  $FF4A
FF3F:          IOREST   EQU  $FF3F
D56C:          CRUNCH   EQU  $D56C
D61A:          FNDLIN   EQU  $D61A
D412:          ERROR    EQU  $D412
0341:          ;
0341:          ; SAVE REGISTERS
0341:          ;
0341:20 4A FF           JSR  IOSAVE
0344:          ;
0344:          ; GET THE LINE NUMBER
0344:          ;
0344:20 BE DE           JSR  CHKCOM
0347:20 7B DD           JSR  FRMEVL
034A:20 52 E7           JSR  GETADR
034D:          ;
034D:          ; SAVE TEXT POINTER
                 TEMPORARILY
034D:          ;
034D:A5 B8              LDA  TXTPTR
034F:8D 9A 03           STA  TEMPTXT
0352:A5 B9              LDA  TXTPTR+1
0354:8D 9B 03           STA  TEMPTXT+1
0357:          ;
0357:          ; TOKENIZE THE INPUT
                 BUFFER
0357:          ;
0357:A9 00              LDA  #$00
0359:85 B8              STA  TXTPTR
035B:A9 02              LDA  #$2
035D:85 B9              STA  TXTPTR+1
035F:A2 FF              LDX  #$FF
0361:A0 04              LDY  #$4
0363:20 6C D5           JSR  CRUNCH
0366:          ;
0366:          ; FIND THE LINE IN THE
                 BASIC PROGRAM
0366:          ;
0366:20 1A D6           JSR  FNDLIN
0369:90 2A              BCC  NOPE
036B:          ;
036B:          ; CHANGE IT TO THE NEW
                 LINE STORED IN THE
                 INPUT BUFFER
036B:          ;
036B:A0 04              LDY  #$04
036D:B9 FC 01  BEGIN    LDA  $1FC,Y
0370:F0 09              BEQ  DONE
0372:91 9B              STA  (LOWTR),Y
0374:C8                 INY
0375:D0 F6              BNE  BEGIN
0377:A2 B0              LDX  #$B0
0379:D0 1C              BNE  ERR
037B:          ;
037B:          ; FILL UP THE LINE WITH
                 COLONS
037B:          ;
037B:A2 3A     DONE     LDX  #$3A
037D:B1 9B     START    LDA  (LOWTR),Y
037F:F0 06              BEQ  LAST
0381:8A                 TXA
0382:91 9B              STA  (LOWTR),Y
0384:C8                 INY
0385:D0 F6              BNE  START
0387:          ;
0387:          ; RESTORE TEXT POINTER
0387:          ;
0387:AD 9A 03  LAST     LDA  TEMPTXT
038A:85 B8              STA  TXTPTR
038C:AD 9B 03           LDA  TEMPTXT+1
038F:85 B9              STA  TXTPTR+1
0391:          ;
0391:          ; RESTORE REGISTERS
0391:          ;
0391:20 3F FF           JSR  IOREST
0394:60                 RTS
0395:A2 5A     NOPE     LDX  #$5A
0397:4C 12 D4  ERR      JMP  ERROR
039A:          TEMPTXT  DS   $2
```

Commodore Computers Bloom with CP/M® and Multi-User Capability.

COMMODORE DEALERS

Expand your customer base by including the vast range of CP/M software development and applications!

Small Systems Engineering provides everything you need to enter the CP/M marketplace. Excellent dealer margins. Advertising and marketing support. CP/M software on Commodore formats. Corvus products. Watch for innovative products from SSE to support your Commodore Computer line.

Now your Commodore can blossom into a full-blown CP/M-based or multi-user business system with the addition of 1 or 2 innovative peripherals. SOFTBOX alone gives you complete CP/M capability. HARDBOX interfaces to the Corvus Winchester disk drive, and multiplexing units, for high storage capacity and multi-user capability.

RUN CP/M WITH SOFTBOX

Add SOFTBOX. The Z80-based computer which connects to your Commodore PET or CBM through the IEEE bus.

Now you have the ability to run any CP/M software or application, giving you a universe of computing capabilities you could only wish for until now.

Simply by plugging the SMALL SYSTEMS SOFTBOX into your Commodore Computer's IEEE-488 port and loading the CP/M disk, your computer will run under the world's most popular disk operating system. No internal connections or modifications to your computer are necessary.

Applications packages designed to work with specific terminals (like Lear Siegler ADM3A, Televideo 912 or Hazeltine 1500) need no modification to work with your computer's screen, since the SOFTBOX allows your computer to emulate any of these devices.

Interfaces that come with your SOFTBOX let you add Corvus hard disk, with capabilities to 80 MBytes of storage. You can run a printer, and even to talk to another CP/M based system.

Available for immediate shipment. 90-day Warranty. Sugested retail $895

Call or write for our free CP/M Software Catalog.

HARDBOX FOR LARGE STORAGE & MULTI-USER CAPABILITY

The HardBox introduces your PET software to the world of high-capacity, high speed disk storage. It acts as an intelligent controller for up to four Corvus Winchester disk drives. Each Corvus drive can have a capacity of 6, 10 or 20 million bytes.

The HardBox is designed to be software-compatible with PET DOS version 1 or 2, so that it will work with existing programs. The HardBox is designed to appear to the PET as a fast, high-capacity floppy disk unit.

When used with the Corvus range of drives, the HardBox provides much more than just a hard disk interface. By using an 8-way multiplexing unit, up to eight PET users, each equipped with a HardBox, may simultaneously address the same hard disk. By using two levels of multiplexing, up to 64 users may be accommodated. The hard disk may be divided logically into user areas.

A comprehensive set of utility programs are provided with the hardbox for file transfer, hard disk configuration, mirror backup, system maintenance etc.

Available for immediate shipment. 90 day Warranty. Suggested retail $695.

SSE

SMALL SYSTEMS ENGINEERING
222 B View St.
Mountain View, California 94041

(415) 964-8201

Program 2.

```
SOURCE FILE: CHANGE - PET
----- NEXT OBJECT FILE NAME IS CHANGE -
      PET.OBJO
033A:               ORG    $33A
005C:        LOWTR  EQU    $5C
0077:        TXTPTR EQU    $77
BEF5:        CHKCOM EQU    $BEF5
BD98:        FRMEVL EQU    $BD98
C92D:        GETADR EQU    $C92D
B4FB:        CRUNCH EQU    $B4FB
B5A3:        FNDLIN EQU    $B5A3
B3CF:        ERROR  EQU    $B3CF
033A:        ;
033A:        ; SAVE REGISTERS
033A:        ;
033A:48             PHA
033B:08             PHP
033C:8A             TXA
033D:48             PHA
033E:98             TYA
033F:48             PHA
0340:        ;
0340:        ; GET THE LINE NUMBER
0340:        ;
0340:20 F5 BE       JSR    CHKCOM
0343:20 98 BD       JSR    FRMEVL
0346:20 2D C9       JSR    GETADR
0349:        ;
0349:        ; SAVE TEXT POINTER
               TEMPORARILY
0349:        ;
0349:A5 77          LDA    TXTPTR
034B:8D 99 03       STA    TEMPTXT
034E:A5 78          LDA    TXTPTR+1
0350:8D 9A 03       STA    TEMPTXT+1
0353:        ;
0353:        ; TOKENIZE THE INPUT
               BUFFER
0353:        ;
0353:A9 00          LDA    #$00
0355:85 77          STA    TXTPTR
0357:A9 02          LDA    #$2
0359:85 78          STA    TXTPTR+1
035B:A2 FF          LDX    #$FF
035D:A0 04          LDY    #$4
035F:20 FB B4       JSR    CRUNCH
0362:        ;
0362:        ; FIND THE LINE IN THE
               BASIC PROGRAM
0362:        ;
0362:20 A3 B5       JSR    FNDLIN
0365:90 2D          BCC    NOPE
0367:        ;
0367:        ; CHANGE IT TO THE NEW
               LINE STORED IN THE
               INPUT BUFFER
0367:        ;
0367:A0 04          LDY    #$04
0369:B9 FC 01 BEGIN LDA    $1FC,Y
036C:F0 09          BEQ    DONE
036E:91 5C          STA    (LOWTR),Y
0370:C8             INY
0371:D0 F6          BNE    BEGIN
0373:A2 B0          LDX    #$B0
0375:D0 1F          BNE    ERR
0377:        ;
0377:        ; FILL UP THE LINE WITH
               COLONS
0377:        ;
0377:A2 3A   DONE   LDX    #$3A
0379:B1 5C   START  LDA    (LOWTR),Y
037B:F0 06          BEQ    LAST
037D:8A             TXA
037E:91 5C          STA    (LOWTR),Y
0380:C8             INY
0381:D0 F6          BNE    START
0383:        ;
0383:        ; RESTORE TEXT POINTER
0383:        ;
0383:AD 99 03 LAST  LDA    TEMPTXT
0386:85 77          STA    TXTPTR
0388:AD 9A 03       LDA    TEMPTXT+1
038B:85 78          STA    TXTPTR+1
038D:        ;
038D:        ; RESTORE REGISTERS
038D:        ;
038D:68             PLA
038E:A8             TAY
038F:68             PLA
0390:AA             TAX
0391:28             PLP
0392:68             PLA
0393:60             RTS
0394:A2 5A   NOPE   LDX    #$5A
0396:4C CF B3 ERR   JMP    ERROR
0399:        TEMPTXT DS    $2
```

Program 3.

```
.:033A 48 08 8A 48 98 48 20 F5
.:0342 BE 20 98 BD 20 2D C9 A5
.:034A 77 8D 99 03 A5 78 8D 9A
.:0352 03 A9 00 85 77 A9 02 85
.:035A 78 A2 FF A0 04 20 FB B4
.:0362 20 A3 B5 90 2D A0 04 B9
.:036A FC 01 F0 09 91 5C C8 D0
.:0372 F6 A2 B0 D0 1F A2 3A B1
.:037A 5C F0 06 8A 91 5C C8 D0
.:0382 F6 AD 99 03 85 77 AD 9A
.:038A 03 85 78 68 A8 68 AA 28
.:0392 68 60 A2 5A 4C CF B3 EF
.
```

Program 4.

```
10 INPUT"{CLEAR}CHANGE TO ";IN$
20 CHANGE=826:LINE=100
30 SYSCHANGE,LINE
40 PRINT"{CLEAR}IN$ EQUALS :"IN$:L
   IST
100 :::::::::::::::::::::::::::::::::
    ::::::::::::::::::::::::::::::
    ::::::::::::::::::::
```

Program 5.

```
10 PRINT"{CLEAR}CHANGE TO ";:GOSUB
   1000
20 CHANGE=826:LINE=100
30 SYSCHANGE,LINE
40 PRINT"{CLEAR}IN$ EQUALS: "IN$:L
   IST
100 :::::::::::::::::::::::::::::::::
   ::::::::::::::::::::::::::
   :::::::::::::::::::
1000 SYS48117
1010 FORB=512TO592:IFPEEK(B)<>0THEN
   NEXT
1020 IN$="":POKEPEEK(68)+256*PEEK(6
   9)+1,0:POKEPEEK(68)+256*PEE
   K(69)+2,2
1030 POKEPEEK(68)+256*PEEK(69),B-51
   2:IN$=MID$(IN$,1):RETURN
```

©

The Beginner's Page

Files Versus Programs

Richard Mansfield
Assistant Editor

A friend of mine bought a computer a couple of months ago. I asked him what he found to be the most baffling, the toughest thing to learn to do with it. "Files," he said without hesitation, "I can't make the dang things work."

Files were also the most confusing thing to me when I first bought a computer. In fact, several years ago, handling files mystified nearly everyone who hadn't had previous experience with computers. Some of the early newsletters for home computerists were full of discussions and techniques on how to make files work.

They are somewhat more tricky than most programming techniques – there is more responsibility left up to the programmer. OPEN, CLOSE, PRINT# and INPUT# are so useful, however, that they deserve to be studied a little until they are understood.

Because file handling (also called *data base management*) requires a bit of explanation, we can look this month at the general differences between programs and files. In the July issue, we'll get down to specific filing techniques.

Telling Them Apart

The first step is to realize that tapes or disks store two different things – programs or files. (Some books refer to programs saved on tape or disk as "program files," but that terminology is worse than redundant, it's also confusing.) A BASIC program is a list of "lines" and each line contains instructions to the computer. These instructions are to be carried out during a RUN of the program. That is, the instructions are followed in order, from the lowest line number to the highest, when you type the word RUN. A data file, by contrast, is raw information like a page in a telephone book, without any instructions about what to do with that information.

When programs are SAVEd onto a disk or tape they can later be LOADed back into the computer to be RUN at any time in the future. Any programs you write into the computer will stay there only as long as the computer is turned on. So, to build a library of programs, you must SAVE them on tape or disk. They are SAVEd as if the tape or disk (let's just call them "magnetic memory") were given a photo of the program that was in the computer at the time of the SAVE. BASIC keeps track of how large a program is, where it starts and ends in the computer's memory cells, so it knows just what to "photograph" when you ask for a SAVE.

You, however, are far more responsible for handling the storage of *files*. BASIC doesn't supervise their storage or recall nearly as completely as it does with programs. You must do several things to create a file on magnetic memory and several things to get it back into the computer later. You establish the size of the file, the divisions between items in the file (called *delimiters*), and the order of the items. We'll illustrate this next month, but first let's *see*, visually, how programs and files differ:

A typical recipe will have both a "file" and a "program" in it:

MEATLOAF

1 lb. Hamburger
1 cup bread crumbs
1/2 cup milk
spices

1. Mix ingredients.
2. Form into loaf.
3. Bake 45 minutes at 325.

Steps one through three are clearly a "program" of sorts. The first clue is that each item starts with a number, indicating the order in which the steps are performed. The ingredients – standing by themselves as raw data – are a file. Just as the ingredients "file" in the example above is *acted upon* by the cooking instructions "program," a computer program acts upon a data file.

On Magnetic Memory

Here's a simple program which will create a tape file on a Commodore computer:

```
10 DATA AAA,BBB,CCC
20 OPEN 1,1,1,"FILE"
30 FOR I = 1 TO 3
40 READ D$
50 PRINT#1,D$
60 NEXT I
70 CLOSE 1
```

PRINT# (usually pronounced "print-number") is an entirely different command from PRINT and the punctuation, as usual in programming, must be exact. Line 40 is interesting because we keep READing D$ over and over to use it as a temporary holding place until we can PRINT# it to a magnetic memory. D$ isn't anything in itself (it varies, it's a *variable*). READ will pick out each datum from the DATA line in turn, keeping track of the last one that was READ.

In any case, after this program is RUN, the magnetic memory would contain a file. If we could look at that file on the tape the way we would look

The Hard & Soft of It: Atari from ASAP

If you're into Atari, get into ASAP. When it comes to Atari, ASAP has it all — computers, a full line of accessories, and one of the most complete lines of software on the market. Whether you use your Atari for work or play, make ASAP your source. Call today.

The **Atari® 800™** Computer features color graphics and English characters with truly high resolution, high quality sound, expandable memory and sleek modular appearance.

Atari® 400™ — 16K also available: **$349.00.**

OPTIONAL ACCESSORIES	**PRICE**
ATARI® 410™ Program Recorder	80.00
ATARI® 810™ Disk Drive	455.00
ATARI® 822™ Thermal Printer	299.00
ATARI® 820™ 40-column Dot Matrix Impact Printer	279.00
ATARI® 825™ 80-column Dot Matrix Impact Printer	625.00
ATARI® 830™ Acoustic Modem	159.00
ATARI® 850™ Interface Module	169.00
ATARI® Paddle (CX30-04) and Joystick (CX40-04)	17.95
ASAP 16K RAM Module	50.00

COMPLETE SOFTWARE LIBRARY INCLUDES THESE POPULAR UNITS:

Atari®	
Educational System ROM	19.95
Video Easel ROM	25.00
Music Composer ROM	42.00
Assembler/Editor ROM	45.00
Mortgage Loan Analysis Cassette	15.95
Stock Analysis Disk	19.95
Stock Charting	22.95
Bond Analysis Disk	22.95
Mailing List Cassette	19.95
Touch Typing (2 Cass)	19.95
Graph It (2 Cass)	17.95
Word Processor	119.00
Personal Finance	64.95
Microsoft Basic	75.00
Basketball ROM	27.00
Super Breakout ROM	30.00
Computer Chess ROM	32.00
3D Tic Tac Toe ROM	25.00
Star Raiders ROM	36.00
Kingdom Cassette	12.95
Blackjack Cassette	12.95
Biorhythm Cassette	12.95
Energy Czar Cassette	12.95
Telelink ROM	24.00
Space Invaders ROM	30.00
Scram	18.95
Asteriods	32.00
Missile Command	32.00
Caverns of Mars Disk	32.00
Centipede Disk	35.00
Pac-Man	call
Arcade Plus	
Ghost Hunter Cassette	25.00
Ghost Hunter Disk	30.00
Datasoft	
Atari Mailing List Disk	19.95
Atari Character Generator Disk	16.95
Text Wizard Disk	89.95
Micropainter Album 1 Disk	16.95
Micropainter Album 2 Disk	16.95
Le Stick Accessory	30.00
On-Line Systems	
HI-RES Adv #0 — Mission: Asteroid Disk	19.95
HI-RES Adv #2 — Wiz & Princess Disk	25.00
Roadwork Disk	29.95
Jawbreaker Disk	25.00
Softporn Adventure (X-rated) Disk	25.00
The Next Step Disk	29.95
Personal Software	
Visicalc Disk	169.00
FOR HOME ENTERTAINMENT SYSTEMS:	
Activision	
Dragster Cartridge	18.50
Boxing Cartridge	18.50
Checkers Cartridge	18.50
Fishing Derby Cartridge	18.50
Skiing Cartridge	18.50
Bridge Cartridge	23.50
Tennis Cartridge	18.50
Laser Blast Cartridge	18.50
Freeway Cartridge	18.50
Kaboom! Cartridge	18.50
Stampede Cartridge	18.50
Ice Hockey Cartridge	23.50

ALL PRICES SUBJECT TO CHANGE WITHOUT NOTICE. CALL FOR BEST PRICE

ASAP offers a 15-day buyer protection policy: full money-back guarantee if not totally satisfied.

Ordering information: name, address, phone; ship by: UPS or Mail. Shipping charge: add $2.90 up to 1 lb. (UPS blue), U.S. Mail add $1.50 (U.S. only) ($25.00 minimum order).

Terms: We accept cash, check, money orders, Visa and Master Charge (U.S. funds only). Tax: 6% Calif. res., COD's and terms available on approval (School PO's Accepted).

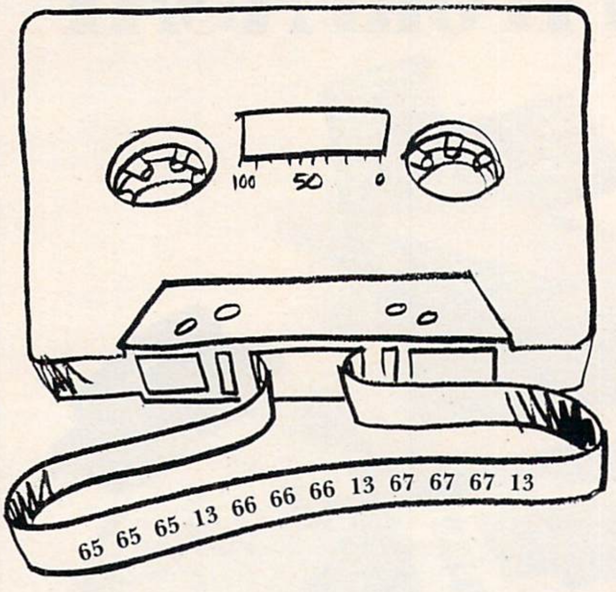
ASAP announces new Candian office. Call for address & phone number.

Toll free outside California: **(800) 421-7701.** Inside California: **(213) 595-6431 (714) 891-2663.**

asap computer products, inc.

1198 E. Willow St., Signal Hill, CA 90806 at a photograph, we would see a row of numbers. The number 65 stands for the letter A and 13 represents a carriage return. Here's what the photograph would look like:

Figure 1: A File

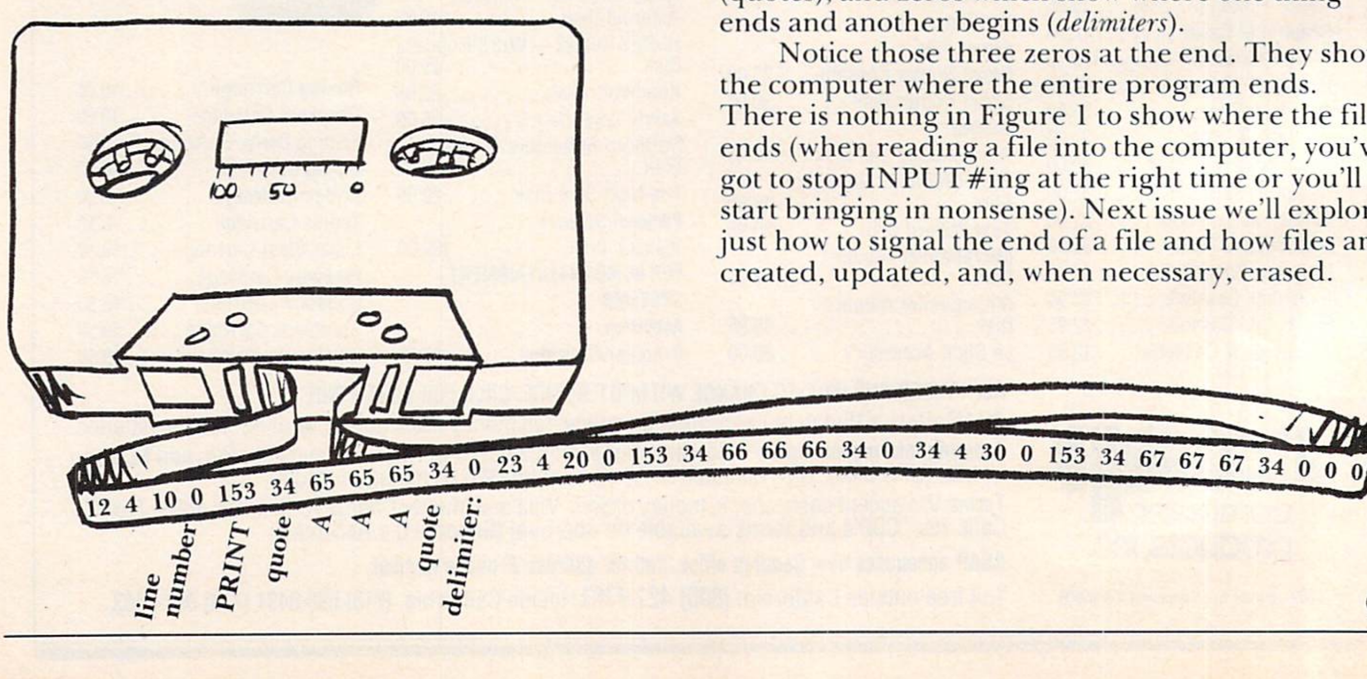


A program (with its line numbers and instructions) would be longer than a file containing the same data:

```
10 PRINT "AAA"
20 PRINT "BBB"
30 PRINT "CCC"
```

The Microsoft BASIC version of this short program would look like this in the computer's memory or on magnetic memory after a SAVE:

Figure 2: The Program



We still find the AAA's, BBB's, and CCC's in there, but surrounded by line numbers, the 34's (quotes), and zeros which show where one thing ends and another begins (*delimiters*).

Notice those three zeros at the end. They show the computer where the entire program ends. There is nothing in Figure 1 to show where the file ends (when reading a file into the computer, you've got to stop INPUT#ing at the right time or you'll start bringing in nonsense). Next issue we'll explore just how to signal the end of a file and how files are created, updated, and, when necessary, erased.

©

This article takes you step by step through the design of a graphics dump routine.

Copy Atari Graphics To Your Printer

Harry A. Straw
Wilmington, DE

Let's look at some techniques involved in copying the graphics window to a printer.

First of all, we have to match the number of columns in the graphics display to the number of columns on the printer. My printer is 80 columns, so GRAPHICS 4 and GRAPHICS 5 (both 80 column modes) are a natural fit. If your printer is different, don't worry. We'll come back to that question later. Let's take the easy situation first, then modify the program as necessary.

For an 80 column printer, the simplest case is GRAPHICS 4: 80 columns by 40 rows, just two colors. We must make the graphics cursor look at each pixel on the screen and tell us whether it is blank (background color) or displays a spot of light. This is easy to do with a double FOR-NEXT loop and Atari's LOCATE command. See Listing 1. At the start, Y=0 (line 10). This corresponds to the top row of the screen. Line 20 then increases X from 0 to 79, one step at a time. Line 30, LOCATE X,Y,Z, causes the cursor to move across all 80 columns (X values) for this value of Y. It also returns the value of Z, the pixel color, at each location. If Z=0, this is background color, a blank spot. If Z=1, a spot of color is found on the screen. The cursor then moves on to the next column (NEXT X). When it reaches the end of the row (X=79), it moves to the next row (Y=1) and starts a new horizontal scan (X=0 to 79 again). See Program 1.

Now all we have to do is tell the printer what to print. If the screen is blank at a given location (Z=0), we tell the printer to print a blank space, and then go on to the next X:

```
40 IF Z=0 THEN LPRINT CHR$(32);: GOTO 60
```

Don't forget the semicolon. It prevents carriage return. If there is a spot of color at this location, Z is not zero, and we want the printer to print something. You can select any character you want by consulting the list of ASCII characters and numbers for your printer. Let's pick the *, ASCII number 42, and add:

```
50 LPRINT CHR$(42);
```

Now we have a complete program for copying the GRAPHICS 4 graphics window to an 80 column printer. (Program 2.)

In case you have trouble with no carriage return, try adding:

```
65 IF X=79 THEN LPRINT CHR$(13)
```

If you get no line feed, make it:

```
65 IF X=79 THEN LPRINT CHR$(10)
```

If necessary, use both:

```
65 IF X=79 THEN LPRINT CHR$(13): LPRINT
   CHR$(10)
```

With Graphics 5

We can easily expand our program to take care of GRAPHICS 5, a four-color mode. The same FOR-NEXT loops and LOCATE statement work. All we need to do is to select a different printer character for each color, and add some more IF-THEN statements to make the printer character correspond to the color at each location. One setup for three characters plus the blank is shown in Program 3. You can change the printed characters to suit yourself.

This program uses high line numbers, starting at 31000, so it can be merged with a program already in RAM without line number interference. Line 31000 is insurance: it makes sure that the printer head starts at the left-hand margin.

It is convenient to record this program on a cassette using the LIST"C command. You can then enter it into RAM using the ENTER"C command without destroying your main program already in RAM. Adding:

```
line no. GOSUB 31000
```

to your main program will now cause your graphics display to print out. Program 3 will work for GRAPHICS 4 and GRAPHICS 5.

What if you have a 40 column printer? Use GRAPHICS 3 (40 columns by 20 rows) to set up your display. Change lines 31010 and 31020 to match the cursor scan to the graphics display:

```
31010 FOR Y=0 TO 19
31020 FOR X=0 TO 39
```

and line 31080 to:

```
31080 IF X=39 THEN LPRINT CHR$(13)
```

A problem can turn up if the last statement in your main program is Atari's XIO "fill" command:

```
XIO 18,#6,0,0,"S:"
```

It makes the computer think that ports are open to read, not write, and it shows

```
ERROR-131, IOCB write only
```

when the printer subroutine starts. You can fix this by inserting the following line after the last XIO statement:

PLOT X,Y: DRAWTO X+1,Y

where X,Y and X+1,Y are any two points within your filled area so the screen display is not changed. The print routine then runs fine.

What can you do with this? Copy graphs or charts of data, or plan ahead for your 1981 Christmas cards. Try running Program 4 along with Program 3!

Line 29999 of Program 4 is not a necessary part of the printer copying routine. It merely provides a line to which the program can RETURN from line 31110, and ends the run. Without line 29999, an ERROR message will appear, but only after the printout has been completed.

Program 1.

```
10 FOR Y=0 TO 39
20 FOR X=0 TO 79
30 LOCATE X,Y,Z
60 NEXT X
70 NEXT Y
```

Program 2.

```
COPY GRAPHICS 4 TO PRINTER

10 FOR Y=0 TO 39
20 FOR X=0 TO 79
30 LOCATE X,Y,Z
40 IF Z=0 THEN LPRINT CHR$(32);: GOTO 60
50 LPRINT CHR$(42);
60 NEXT X
70 NEXT Y
```

Program 3.

```
GRAPHICS 4 OR 5
4 COLORS

COPY GRAPHICS TO PRINTER

USE 'GOSUB 31000' IN MAIN PROGRAM

31000 LPRINT CHR$(13)
31010 FOR Y=0 TO 39
31020 FOR X=0 TO 79
31030 LOCATE X,Y,Z
31040 IF Z=0 THEN LPRINT CHR$(32);:GOTO
   31080
31045 REM - COLOR 1, Z=0 - BACKGROUND
31050 IF Z=1 THEN LPRINT CHR$(42);:GOTO
   31080
31055 REM - COLOR 2, Z=1
31060 IF Z=2 THEN LPRINT CHR$(43);:GOTO
   31080
31065 REM - COLOR 3, Z=2
31070 LPRINT CHR$(111):
31075 REM - COLOR 4, Z=3
31080 IF X=79 THEN LPRINT CHR$(13)
31090 NEXT X
31100 NEXT Y
31110 RETURN
```

Program 4.

```
10 GRAPHICS 5
20 COLOR 1
30 PLOT 55,31:DRAWTO 40,7
40 POKE 765,1
50 POSITION 25,31
60 XIO 18,#6,0,0,"S:"
70 COLOR 3
80 FOR X=39 TO 41
90 PLOT X,32:DRAWTO X,39
100 NEXT X
110 COLOR 2
120 PLOT 40,6
130 PLOT 38,7:DRAWTO 42,7
140 PLOT 39,8:PLOT 41,8
150 GOSUB 31000
160 LPRINT CHR$(10):LPRINT CHR$(10)
170 LPRINT "                              MERR
 Y CHRISTMAS !!"
29999 GOTO 29999
```

©

Santa Cruz Educational Software * Tricky Tutorials * Santa Cruz Educational Software * Tricky Tutorials * Santa Cruz Educational Software * Tricky Tutorials *

ATARI (tm) GRAPHICS AND SOUND MADE EASY!

SANTA CRUZ EDUCATIONAL SOFTWARE HAS WRITTEN A SERIES OF AFFORDABLE PROGRAMS THAT DEMONSTRATES MANY OF THE SPECIAL "TRICKS" THAT ONLY THE ATARI 400/800 COMPUTER IS CAPABLE OF DOING. WE OFFER EVERYTHING FROM A PROGRAM THAT DOES FANTASTIC HIGH RESOLUTION GRAPHICS TO ONE THAT ACTUALLY DIALS YOUR PHONE.....

★★★

MASTER MEMORY MAP(tm) — This is really the key to using the ATARI'S capabilities. We start out by explaining how to PEEK and POKE values into memory so that even new programmers can use this. Then: we give you over 15 pages of the memory locations that are the most useful. The information is condensed from both the ATARI'S Operating System Manual and various articles and programs. It is, of course, useful even for experienced programmers as a reference. Also, we highly suggest that dealers offer this Memory Map to customers who request to be told how to use the power of the machine. We guarantee it will answer many of the questions you have about the machine.

$6.95

TRICKY TUTORIALS(tm)

#1: DISPLAY LISTS — This program teaches you how to alter the program in the ATARI that controls the format of the screen. For example: when you say graphics 8 the machine responds with a large graphics 8 area at the top of the screen and a small text area at the bottom. Now, you will be able to mix the various modes on the screen at the same time. Just think how nice your programs could look with a mix of large and small text, and both high and low resolution graphics. this program has many examples plus does all of the difficult caluculations! **$19.95**

#2: HORIZONTAL/VERITCAL SCROLLING — The information you put on the screen, either graphics or text, can be moved up, down or sideways. This can make some nice effects. You could move only the text on the bottom half of the screen or perhaps create a map and then move smoothly over it by using the joystick **$19.95**

#3: PAGE FLIPPING — Normally you have to redraw the screen every time you change the picture or text. Now you can learn how to have the computer draw the next page you want to see while you are still looking at the previous page, then flip to it instantly. You won't see it being drawn, so a complicated picture can seem to just appear. Depending on your memory size and how complicated the picture, you could flip between many pages, thus allowing animation or other special effects with your text. **$19.95**

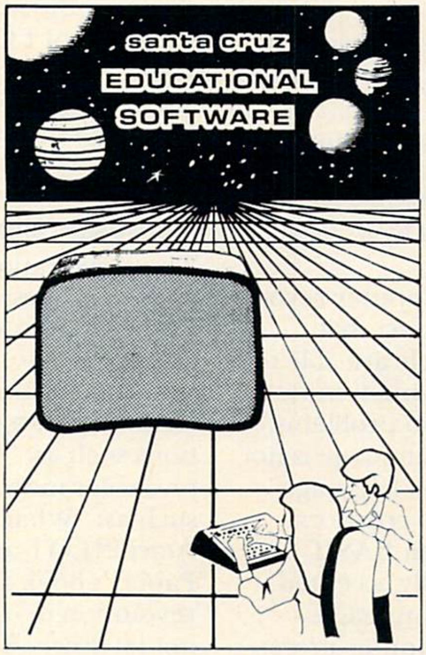
#4: BASICS OF ANIMATION — Shows you how to animate simple shapes using the PRINT and PLOT commands, and also has nice little PLAYER/MISSILE Graphics demo to learn. This would be an excellent way to start making your programs come alive on the screen. Recommended for new owners **$19.95**

#5: PLAYER MISSILE GRAPHICS — Learn to write your own animated games and business applications! The Tutorial begins with many small examples that complement the 50 page manual, then gradually builds up to a complete game as everything you need to know is explained fully. Also included are two machine language routines that you can use to animate Players from Basic. Next we include one of the best editors currently available, one for Playfield shapes, and one for Player shapes. All of the examples and games run with 16K Tape or 32K Disk. The two utilities require 32K to run. **$29.95**

#6: SOUND — From explaining how to create single notes, to demonstrating complex four channel sound effects, this newest tutorial is great. Even those experienced with ATARI's sound capabilities will find the menu of sound effects a needed reference that can be used whenever you are in the need of a special sound for your programs. Everyone will learn something new! Written by Jerry White. **$19.95**

Tricky Tutorials (except #5) require 16K memory for cassette orders and 24K for disk. The price is $19.95 each. Tricky Tutorials #1 to 6 in a notebook cost only $99.95, a $30 savings.

THE GRAPHICS MACHINE!! — Turn your computer into an incredible graphics tool with advanced commands like circle, box, fill, polygon, line, help, etc. 3 colors in graphics 8 with instant text!!! Create colorful business charts or beautiful drawings and then save or retrieve them from disk in 5 SECONDS. YES, it's that fast. Needs all 48K, disk, and costs **$19.95**



MINI-WORD PROCESSOR — This is for those of you who have a printer, but don't want to spend $100 or more for a fancy word processor. It is suitable for simple editing of text, accepts most control characters for your printer, and text is stored on disk for easy retrieval. Holds 2½ typed pages at a time. Requires 32K, disk or tape. **$19.95**

BOB'S BUSINESS — 14 small business type programs for home or office, all chosen from a nice menu. Supports printed output. 169 sectors of output require 16K tape, or 32K disk. **$14.95**

KID'S #1 — Includes the following: 1) TREASURE — search for the lost treasure while trying to keep from falling into the sea. Nice graphics if you find it!; 2)DIALOGUE — talk back to the computer about four subjects; 3) MATH QUIZ — Nice musical and graphical rewards for good scores. Parents input the level of difficulty.

KID'S #2 — A spelling quiz, a "scrabble" type game, and a version of Touch with the computer giving all the directions! Both Kid's programs require 16K tape or 24K disk and cost **$14.95** each.

MINI-DATABASE/DIALER — This unique new program stores and edits up to 8 lines of information such as name, address, and phone numbers, or messages, inventories or anything you want. It has the usual sort, search, and print options, but it also has an unusual feature: If your files include phone numbers and you have a touch-tone phone, the program will DIAL THE PHONE NUMBERS FOR YOU! This is perfect for those who make a lot of calls like salesmen, teens, or those trying to get through to busy numbers (acts as an auto-redialer). It is also a lot of fun to use. Requires 16K cassette or 24K disk and costs **$24.95**

FONETONE — For those who only want to store name and phone numbers and have the dialer feature as above, we offer this reduced version. Same memory requirements, but only costs **$14.95.** Don't forget you must have a touch-tone phone.

PLAYER PIANO — Turns your keyboard into a mini-piano and more. Multiple menu options provide the ability to create your own songs, save or load data files using cassette or diskette, fix or change any of up to 400 notes in memory, and play all or part of a song. The screen displays the keyboard and indicates each key as it is played from a data file or the notes you type. You don't have to be a musician to enjoy this educational and entertaining program. Requires 24K cassette or 32K disk. **$14.95**

BOWLERS DATABASE — Provides the league bowler with the ability to record and retrieve bowling scores providing permanent records. The data may then be analyzed by the program and displayed or printed in summary or detail form. Data may be stored on cassette or diskette and updated quickly and efficiently. The program proivdes such information as highest and lowest scores by individual game, (first, second, and third games throughout the season), high and low series, current average, and more. The program listing and documentation provided are a tutorial on ATARI basic and record keeeping. Requires 16K for cassette or 24K for disk. **$14.95**

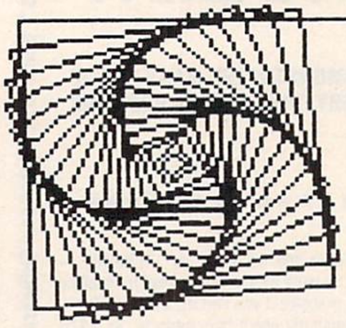
#7: DISK UTILITIES TUTORIAL — This Tutorial both offers you several programs to aid in the use of your 810 Disk drive, and teaches you how these utilities were written at the same time. The utilities included are a Menu program, a Disk inspector (look at sectors on the disk), a Disk Directory printout program, a Format utility to quickly format large numbers of disks, and a utility that allows your programs to start running by just turning on the 810 (Autorun.sys). Requires 32K of memory and costs only **$29.95**

★★★

SEE YOUR LOCAL DEALER
OR
ORDER FROM:

SANTA CRUZ EDUCATIONAL SOFTWARE
5425 Jigger Dr., Soquel, CA 95073
C.O.D. Orders call (408) 476-4901
***Include $3.00 postage ($1.00 for Memory Map)**
***In California, include 6.5% tax**
Visa and MasterCard Accepted

Santa Cruz Educational Software * Tricky Tutorials * Santa Cruz Educational Software * Tricky Tutorials * Santa Cruz Educational Software * Tricky *



Friends Of The Turtle

David D. Thornburg
Los Altos, CA

The Computer Faire Of The Turtle...

While some might argue with the exact date, I place the start of the Personal Computer Revolution in 1978. That was the year in which affordable desk-top computers were first made available to the general public. The big sellers that year, Commodore, Apple, and Radio Shack, are still going strong – as are other companies who joined in the explosion of enthusiasm which greeted these products.

But during these past years there was another revolution brewing – a revolution in computer languages which promised to make the newly affordable computer easy to program by its largely non-technical owners.

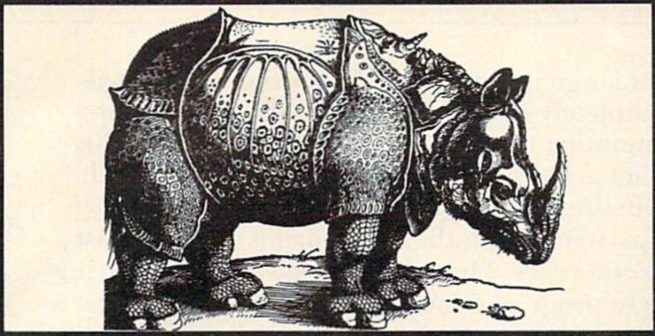
The mainstay of the personal computer revolution was the language BASIC. The fact that many hundreds of thousands of people are able to write programs in this language is strong testimony to its effectiveness. But BASIC has two problems. First, the threshold for learning the language is not very low and, second, the power of the language isn't large enough to invite the user to create extremely sophisticated programs. When BASIC was the only language in town, it was gladly accepted. After all, one alternative – assembly language – doesn't appeal to many first-time computer users; and more powerful structured languages such as PASCAL seem too complex for people interested in balancing checkbooks or generating games.

But, for a decade before 1978, research in university and industrial laboratories was pointing the way to a new type of computer language – a language with a low threshold for learning and a power so great that it could continue to serve the needs of its user at any level of sophistication. One such language, LOGO, was developed and studied on the East Coast, primarily at MIT. While research showed that this language was easy for children to learn and powerful enough for advanced applications, one problem remained – LOGO needed a lot of memory in which to run. As a result, most potential users had to be content either with reading articles about the language or, more recently, with reading Seymour Papert's book, *Mindstorms*.

And then, last year, the seeds of the new revolution began to sprout. Atari released its version of PILOT – a powerful yet simple text manipulation language which had been enhanced by the addition of a graphics environment similar to that in LOGO. At about the same time, Texas Instruments released a version of LOGO which had been compressed to fit on a memory expanded TI 99/4. With these two products, it was evident that a new class of computer language was starting to appear on small affordable computers.

Increasing Literacy

The acceleration of this trend was most evident at the 7th West Coast Computer Faire held in San Francisco this March. This show was packed by attendees who, in my estimation, were the most computer literate group to ever attend this show. In past years an exhibitor was likely to hear questions such as: "Why can't I receive television signals on a color monitor?" This year I was asked questions such as: "What are the major differences between Atari PILOT and LOGO?" Many people had read Papert's book and were fully prepared for the revolution in user friendly languages. They were not let down. The presence of Seymour Papert as keynote speaker and the booths providing information on YPLA (Young People's LOGO Association), FOLLLK (Friends of LISP, LOGO, and Logic for Kids), and FOTT (Friends of the Turtle) set the tone for the release of several versions of LOGO for the Apple II. A special exhibit on the fourth floor of the Faire devoted considerable space to the demonstration of Apple's own LOGO product which was developed by LOGO Computer Systems, Inc. (LCSI). In addition to the language, other exhibits included a prototype of a "sprite" board for the Apple which allows the computer to control four animated turtles at once. Two floors down, another version of Apple LOGO was being offered by Terrapin – a company known previously for its computer controlled robot turtles. (I have copies of both the LCSI and Terrapin LOGO, and



Be sure to enter Krell's giant

$30,000

EDUCATIONAL SOFTWARE CONTEST

WRITE FOR DETAILS. LUDWIG BRAUN. D.E.E.
DIRECTOR OF EDUCATIONAL PRODUCT DEVELOPMENT

College Board SAT* Prep Series

TRS-80, APPLE, PET, ATARI, CP/M, PDP-11 Based on past exams, presents material on the same level of difficulty and in the same form used in the S.A.T.

S.A.T., P.S.A.T., N.M.S.Q.T. — Educator Edition set includes 25 programs covering Vocabulary, Word Relationships, Reading comprehension, Sentence Completion, and Mathematics. $229.95
INDEPENDENT TESTS OF S.A.T. SERIES PERFORMANCE SHOW A MEAN TOTAL INCREASE OF 70 POINTS IN STUDENTS' SCORES.

GRADUATE RECORD EXAM Series — Educator Edition includes 28 programs covering Vocabulary, Word Relationships, Reading Comprehension, Sentence Completion, Mathematics, Analytical Reasoning and Logical Diagrams. $289.95

Micro-Deutsch

MICRO-DEUTSCH set includes 24 grammar lessons, covering all materials of an introductory German course. Four test units also included. Grammar lessons use substitution transformation drills, item ordering, translations and verb drills. Drill vocabulary based on frequency lists. Suitable for use with any high school or college textbook. Extensively field tested at SUNY, Stony Brook. APPLE*, PET*. $179.95

Isaac Newton + F.G. Newton

Perhaps the most fascinating and valuable educational game ever devised - ISAAC NEWTON challenges the players to assemble evidence and discern the underlying "Laws of Nature" that have produced this evidence. Players propose experiments to determine if new data conform to the "Laws of Nature". FULL GRAPHICS NEWTON - presents all data in graphic form. This game is suitable for children. Players may select difficulty levels challenging to the most skilled adults. Both Games $49.95

Pythagoras and the Dragon

Mathematics in a fantasy game context. Based on THE SWORD OF ZEDEK introduces Pythagoras as a mentor to the player. When called on for aid, Pythagoras poses math questions and depending on the speed and accuracy of response, confers secret information. With Pythagoras as an ally, the quest to overthrow Ra, The Master of Evil, assumes a new dimention of complexity. Depending on the level chosen, problems range from arithmetic through plane geometry.
APPLE, TRS80, PET, ATARI* 32K $39.95

ALSO AVAILABLE:

Time Traveler
Odyssey in Time
Sword of Zedek
Krell Game Pak
Super Star &
All Time Super Star Baseball

LOGO

© M.I.T.

Krell's LOGO for APPLE II*
Includes: our new low price and

1. Two copies of Krell's LOGO for Apple II*

2. A UTILITY disk containing M.I.T.'s extremely valuable demonstration material, a series of pre-defined shapes which may be substituted for the LOGO TURTLE, and Krell's timesaving library of pre-defined procedures.
3. ALICE IN LOGOLAND, a twenty program tutorial series for intermediate and experienced programmers new to the LOGO language.
4. The technical manual LOGO for Apple II* by H. Abelson & L. Klotz. This is the official technical manual issued by M.I.T.
5. LOGO FOR THE APPLE II* by Harold Abelson, Byte Books. A complete instructional manual for intermediate and advanced users of LOGO.
6. THE ALICE IN LOGOLAND PRIMER, a step by step, lavishly illustrated introduction to LOGO for those who are new to both programming and to LOGO, (available in May and sent free to all purchasers of Krell's LOGO for Apple II*).
7. A comprehensive wall chart that portrays, explains and graphically illustrates the LOGO commands in action.
8. A one-year's free subscription to the LOGO & EDUCATIONAL COMPUTING NEWSLETTER. (Regularly $30.00 per year). A new authoritative source of information about the structure, classroom application and capabilities of LOGO, plus an up to the minute forum on the most significant new ideas and issues in educational computing. **$149.95**

Competency/Proficiency Assessment Examination Preparation Series

A COMPREHENSIVE MULTI-MODULE TUTORIAL AND EXAMINATION SIMULATION PACKAGE

- Diagnostic analysis
- Prescription of individual study plans
- Systematic instruction in all facets of math and verbal skills
- Unlimited drill and practice with selection by skill area or level of difficulty
- Worksheet generation and performance monitoring
- Practice exam simulations
- A complete record management system

Krell's unique logical design provides personalized instruction for each student according to individual needs.

APPLE, ATARI, COMMODORE CBM/PET, CP/M, I.B.M.
RADIO SHACK TRS-80, TEXAS INSTRUMENTS 99/4(A)*
CALIFORNIA, N.Y., NATIONAL EDS. CALL FOR DETAILS & PRICES

Krell Software Corp.

"The State of the Art in Educational Computing"

1320 STONY BROOK ROAD / STONY BROOK NY 11790 / (516) 751-5139

Krell Software Corp. has no official ties with the College Entrance Examination Board or the Educational Testing Service. Krell is, however, a supplier of products to the E.T.S. N.Y.S. residents add sales tax.
PROGRAMS AVAILABLE FOR THE TRS80, APPLE II, PET & ATARI.
*Trademarks of Apple Comp Corp., Tandy Corp., Commodore Corp., Atari Corp.

will report my opinions to you in a later column. At first glance, they are each terrific!)

Just as some people feel that IBM has legitimized the personal computer by entering the market themselves, one got the feeling that the presence of LOGO on the Apple (with its massive installed base) was going to help wean people from BASIC faster than might otherwise be expected.

Product and information booths were only one source of information on this topic. In addition, no less than a dozen demonstrations, workshops, tutorials and speeches were devoted to user friendly languages.

While I spent much of my time helping Addison Wesley show my book on turtle geometry (yes, Atari PILOT fans, *Picture This!* is now at your local bookstore!), I was still able to talk with many attendees and visit the other booths. Those people who knew about LOGO could hardly wait to see a version on their own computer.

Decked out in my Friends of the Turtle T-shirt, I visited with Rich Pattis who was demonstrating software supporting his excellent book, *Karel the Robot* (Wiley) – a book devoted to introducing people to programming through the medium of the turtle. While geared towards the beginning PASCAL programmer, Pattis' work shows a sensitivity that is characteristic of the user friendly languages such as PILOT and LOGO. The people from FOLLLK were acting as guides to the host of LOGO-based exhibits and talks. Larry Muller and his dad, Jim, demonstrated TI LOGO at the YPLA booth. Recent price reductions in both the TI 99/4A computer and in the TI LOGO cartridge have further fanned the flames on a product whose sales were already heating up quite nicely.

The Computer As A Mudpie

Considering the booths, talks, workshops, and enthusiastic attendees, it was clear that this year's Computer Faire was the focal point of the new revolution – the user friendly languages had come home at last. People were lined up against the walls to hear Papert's keynote address. While much of his talk was devoted to describing the function of the World Center for Informatics and the Human Resource (in Paris, France), he also talked about his view of the computer as a "mudpie" – a tool with which children could (through languages such as LOGO) make discoveries on their own and with which they could acquire for themselves information which was previously "taught" to them by teachers. It was easy to be swept along in the belief that we were witnessing the onset of a revolution which promises to be as significant as the advent of the personal computer itself.

I can state, without equivocation, my belief that languages such as LOGO and PILOT will completely displace BASIC as the popular programming medium in the next five years. This belief arises not from my own excitement with something new, but from the results of my own experiences with these languages over the past several years. I have had the pleasure of sharing these programming environments with children from second to sixth grade, as well as with teachers, college students, and artists. The enthusiasm expressed by these varied audiences is enormous.

And each of you who calls yourself a Friend of the Turtle is sharing in this new age of computing – in this new level of power now being unlocked in the Apple, Atari and TI computers all over the world.

Let the revolution continue!

For more information on LOGO and other user friendly languages, contact:

Young People's LOGO Association
1208 Hillsdale Dr.
Richardson, TX 75081

The FOLLLK Foundation
c/o Social and Information Service
HLL 382
San Francisco State University
1600 Holloway Ave.
San Francisco, CA 94132

and, of course,

Friends of the Turtle
P.O. Box 1317
Los Altos, CA 94022

©

The World Inside The Computer

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!**

Last issue we looked at the history, and some of the future possibilities, for the microelectronics world. We closed with the question, "What does all this have to do with our children?"...

Architects Of The Micro World

Our journey into the world inside the computer might be like some exotic travel story. It might be like Jack London telling tales from the far north, or like Gulliver describing his voyage to Lilliput. It might be far removed from our everyday experience and the concerns of us and our children.

Except it's not.

All of the changes, the fascinating developments in chip design and technology, might not be expected for many years, and these new designers might all be faceless adult experts, hidden away inside corporate and university labs, performing mysterious feats of electronic alchemy.

Except they're not.

Just a couple years ago, Lynn Conway of Xerox and Carver Mead of Caltech wrote a book called *An Introduction to VLSI Systems* (Addison-Wesley, 1980). In it, and in college courses they taught, Conway and Mead called for a new generation of chip designers – architects of the micro world. Conway and Mead proved that, using automated drafting tools (*CAD/CAM* – Computer-Aided Design/Computer-Aided Manufacturing), even graduate students at universities could design custom-made computer chips. And they didn't have to be engineers or experts in computer logic.

Young people, by the thousands, are following the lead of Conway and Meade's students, and are designing custom-made chips and revolutionary new kinds of computers. They treat the basic components – the transistors, gates and memory cells – like "bristle blocks." They sit in front of a computer keyboard, press buttons, and the computer fits the bristle blocks together, and displays the mazelike circuit on a color picture screen. When the chip is finished, the student and his teacher send the design to a "silicon foundry" – a regional center that prints the chip on a sliver of silicon.

Within days, the student has the chip back and can plug it into a circuit board and turn on the power. The chip might be a new kind of computer, a "graphics engine" specializing in high-speed, animated color pictures, or a music-synthesis chip capable of making the computer sound like a bass guitar or a pipe organ.

For two decades, chip design was done exclusively by experts at major corporations, such as Intel, Texas Instruments, and Motorola. Now, all this has changed, due to the new CAD/CAM tools, to the new microminiaturization (VLSI – Very Large-Scale Integration) of circuits, and to Conway and Meade's "paint-by-numbers" approach to chip design.

Today, computer *software* is being written and

Designing new computer chips with the aid of intelligent programs and color graphics.

CREDIT: courtesy of Henry Fuchs and the Department of Computer Science, University of North Carolina. Photo by Mike Pique.

sold by thousands of people, all over the world. It is a cottage industry performed in the home by a lot of low-budget suppliers. Big companies are becoming software publishers and distributors for the software "authors."

Similarly, in just a few years we will probably see a cottage industry of chip authors. CAD/CAM equipment will be rented or cheap enough to install in the home. "Intelligent" chip design programs, like Xerox's *Palladio*, will work with young chip designers. The programs will check the young people's designs for mistakes, suggest new designs of their own, and explain the trade-offs between different designs.

Chip architects are now appearing in colleges and universities. They will soon appear in high school. They will design thousands, even millions of new computers. The best computers will find their way to the marketplace. Large "chip foundries" will print the chips. Large "chip publishers" will market them.

"It's like painting with numbers." Inventing a new computer may someday be as easy as writing a game program in BASIC.

CREDIT: Courtesy of Floyd James, Henry Fuchs and the Department of Computer Science, University of North Carolina. Photo by Jim Erickson, Raleigh *News & Observer*.

The Playdough Computer

What do we mean by "personal computers", or by "personal computing"?

Until now, we meant *one person, one machine* (Portia Isaacson's definition).

Personal computing has made the computer accessible to the average person, the non-expert. Even little kids can sit down in front of the family's home computer and use it to learn or play a game.

But personal computing will soon have an added meaning. It will mean *computing on a machine we designed ourself.* Even young people will be able to fashion their own computers, like they build an airplane or castle from playdough or clay. Everyone will have the opportunity to become a computer designer – a computer artist, an architect of the micro world.

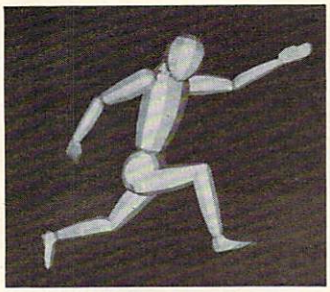
Naturally, airplanes made of clay do not fly. And computers made by young children may not compute. Still, they will be an important exercise in creativity. And, unlike models made from clay or plastic, simple computer models can gradually become more sophisticated and realistic. They can be a first step toward building real chips and computers that work.

Many of today's most innovative, most successful software entrepreneurs are young people, of college or high-school age. In the near future, young people will be some of the most creative chip architects, artists, and inventors.

Your Kids: The Computer's Creators

The purpose of these first columns has been to demonstrate the relevance and the reality of the world inside the computer – to all of us and to our children.

Although the surface of the computer will continue to remain the same, the computer's insides will soon change dramatically. New computers will



Creating the tiny inhabitants of the world inside the computer, using advanced computer graphics, animation, and Artificial Intelligence. Of course, the inhabitants need not be human. They might be sprites or turtles – or whatever else you can imagine.

CREDIT: Charles Csuri and Ohio State University Computer Graphics Laboratory.

soon be created with capabilities exceeding our wildest dreams. The revolutionary new computers will be molded from youthful imaginations. Young people will be the computers' creators.

Fantasy, Turtles, and Sprites

The present trend toward black-box computers is positive, in that it is encouraging millions of people to try personal computing. But don't let this trend "distance" you or your children from the world inside the computer. The real action is taking place under the computer's "hood."

How do you and your children learn more about the world inside the computer? How can you learn *today*?

Computer micro-worlds are so enchanting, so exciting that they can be a pleasure to learn about and explore. Pretend that you and your children are pioneers. I hope these first columns have given you some ideas about where you can focus your pioneering.

But what about a pioneering vehicle – a land rover, jeep, or canoe? After all, sophisticated computer-design tools, such as CAD/CAM machines, still aren't off-the-shelf items, sold at your local Radio Shack for $19.95. How can you explore the world inside the computer without the proper tools?

Fortunately, the first tools are now appearing. They are, first of all, what Bob Albrecht calls the "Rainbow Computers" – the sound-and-color machines that retail for less than five hundred dollars. The Sinclair ZX 81A, the low-cost VIC machines, the TI 99/4A, the TRS-80 Color Computer, and the Atari 400 are all rainbow computers. They provide an environment for simulating the world inside the computer. Using the rainbow computers, you can build models of that world with computer programs.

What kind of programs? You can use a language like BASIC. But new languages are now appearing that make superior micro-world building tools. These languages are Atari PILOT and the various versions of LOGO, including Apple LOGO and TI LOGO.

When you write a program in BASIC, you are like a chef inventing a recipe for a new food dish. The recipe itself is the list of steps you must follow (the *algorithm*) to get the program to perform some function. To make the recipe work, you need to add ingredients in a precise amount and in the proper order. In a program, these ingredients are your *data*.

But programming in the new languages – especially LOGO – is quite different. You are no longer a chef working with a recipe and ingredients. Now you are a band leader conducting a band of jazz musicians, or a film director overseeing the efforts of a diverse bunch of live actors.

When you program in BASIC, you are dealing with inert objects and structures, and the computer marches in lockstep, doing one thing at a time.

On the other hand, when you program in LOGO, an *actors* language, you treat the computer like a tiny world. You are the world's creator. You define the world's laws. You populate the world with tiny creatures. TI's LOGO world comes with "canned" sprites. Various LOGO, PILOT, and PASCAL worlds come with prepackaged, artistic turtles.*

And, using these languages, you can also create other beings drawn purely from your own imagination.

When you type RUN, you turn the world *on*. You breathe life into your creatures and set them in motion. All together. All at the same time.

In upcoming columns, I will develop "computer world" programs written in BASIC, in Atari PILOT and in Apple and TI LOGO. Also, you should consult David Thornburg's column, "Friends of the Turtle," which appears each month in **COMPUTE!** (And take a look at the references at the end of this month's column.)

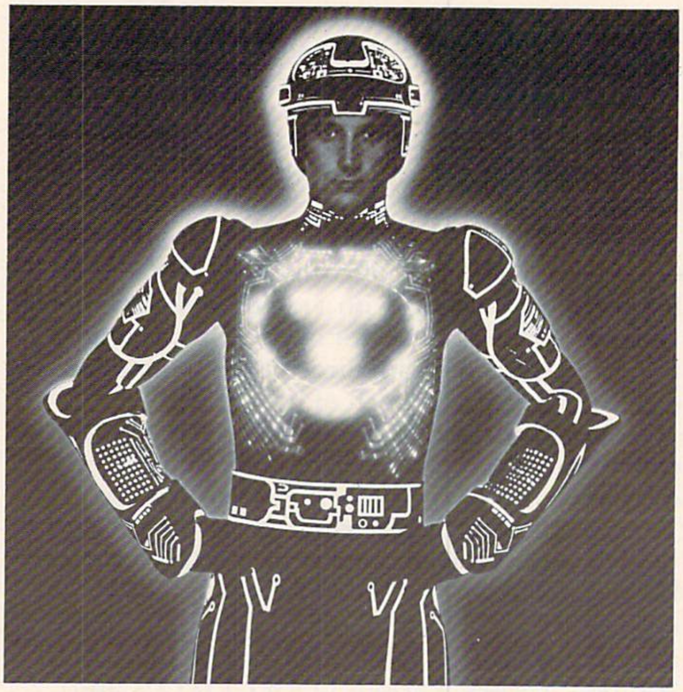
The Computer World Goes to Hollywood

The world inside the computer is a little known and obscure place. But it won't be for long.

This summer (on July 9th), Walt Disney Productions will be releasing the long-awaited film, *TRON*. The hero of TRON is a young computer genius who gets transported to the world inside the computer – Hollywood style.

The hero, Flynn, is the owner of a video game arcade and the inventor of fabulous new game

*I am excited about Alan Kay's new job as Chief Scientist at Atari. Kay is the brilliant scientist who helped create Xerox's Smalltalk language system, and the Xerox Altos and Star computers. Smalltalk is one of the most powerful "Actor" languages. It lets you easily create computer worlds filled with active, interacting beings, processes, and events. Two Smalltalk-type machines, code-named, "Mackintosh" and "Lisa," are under development at Apple Computer Company. And it is likely that at Atari, Kay will be developing Smalltalk-type languages that will run on Atari computers.

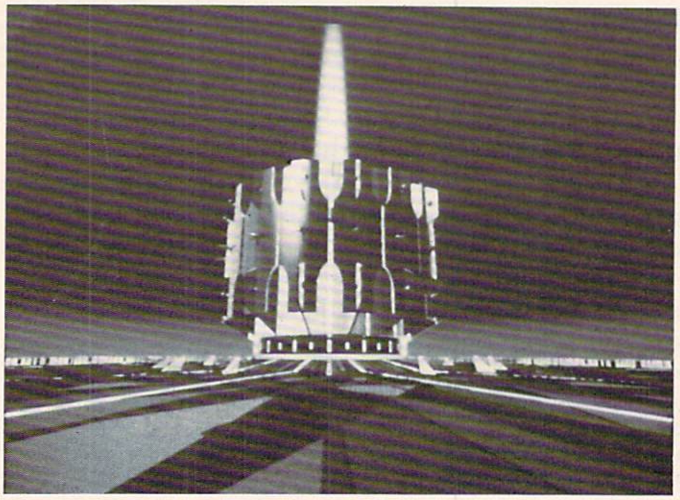


What do inhabitants of the world inside the computer look like? If you entered a computer, what would you look like? This is Walt Disney Productions' answer in its new movie TRON.

CREDIT: Courtesy of Walt Disney Productions. Copyright 1982, Walt Disney Productions.

programs. But the programs are stolen by a video game company. To recover his programs, Flynn breaks into the company's computer system but is caught by the computer's intelligent control program. The evil program, MCP, uses a laser to blast Flynn into electronic particles.

Somehow Flynn survives. When he awakens, he is no longer in the real world. Instead, he has entered the world inside the computer. But he has entered the world as a condemned prisoner. The control program places him on a huge video game grid and sentences him to die.



The fantasy world inside the computer in Walt Disney's new movie TRON.

CREDIT: Courtesy of Walt Disney Productions. Copyright 1982, Walt Disney Productions.

Several leading computer graphics firms have helped Disney produce the film. Their vision of the world inside the computer is sure to be startling, enchanting, and exciting.

But it is only one vision. There are endless others. And you and your family can create them.

The metaphor of the "world inside the computer" will soon become widely known. Already, primitive worlds exist in the form of arcade and home video games. The mazes, rockets, monsters, and little people you see on the game screen emerge from their silicon "homes" when the game is turned on. They are swiftly becoming more lifelike and realistic.

A Rabbi In A Box

Have you ever been to a novelty store and seen the Thing-in-a-Box? The box is really a small, black, plastic cube. You flip an ON switch, and the cube begins whirring strangely. Then the top of the cube opens, and a little green hand pokes out, knocks the switch to OFF, and, in the blink of an eye, disappears back inside the cube.

After watching the box in action, you have the strong impression that someone – or something – is living inside.

We may soon have the same impression about our personal computer.

Personal computers may soon have Artificial Intelligence (AI) programs and AI chips. Your computer may become an "intelligent assistant" that can think, reason, even learn. It will be a "fluent" computer that can carry on a casual conversation with you in normal English. If your family or friends won't listen to you, you will be able to turn on the computer and have a good heart-to-heart discussion with a machine.

AI research, tools, and technology are hot items. After almost four decades of research, AI professors have tied their bedsheets and pillow cases together into a rope, and have descended the ivory tower. Suddenly, at the same time their feet touch the ground, AI is becoming a big business.

What kind of big business? The hottest item in AI is known as *knowledge engineering.* AI scientists have created "expert systems" – chip-sized electronic clones of human experts. Already, these intelligent programs have been put to work diagnosing lung diseases, locating mineral deposits, and designing new clothing fashions. For a relatively small price, you will soon be able to have a world-class expert working alongside you, even if you are a lonely country doctor, or an oceanologist on an oil platform in the turbulent North Atlantic.

Putting a human-like, intelligent program on a chip is like having a wise slave who lives in a package of gum, or like a genie who lives inside a music box. We will very shortly see experts-in-a-box, advisors-in-a-box, and teachers-in-a-box. The knowledge, expertise, and personality of a doctor, a minister, a psychoanalyst, or a lawyer can be "mined" and captured on a home computer in the form of a small, inexpensive chip. AI companies like Cognitive Systems, Inc., Computer Thought Corporation, and Machine Intelligence Corporation are already developing the first AI software for personal computers.

"Know thyself," is an ancient prescription and a unique ability of human beings. Only humans seem to have a strongly developed sense of self-consciousness and self-awareness.

But not for long.

New computer chips are being developed with "expert" programs that know how to design other computer chips. These chips are being fed huge quantities of knowledge. (Some use optical scanners and hungrily read all sorts of technical papers and reports). What do they learn? They are learning about themselves, and about others of their kind. They are being endowed with a primitive sense of self-awareness.

Imagine someday when entire micro universes inside the computer – the chip neighborhoods, cities, and worlds – all have a sense of self-knowledge and self-consciousness. This awareness will only partly be human. Much of it will be non-human, perhaps even alien. It will be suited to the chips' electronic, inorganic bodies and to the complicated knowledge pattern stored in their local and global memories.

One day you and your children may design your own chips and create your own real computer worlds. Yet, even today, you can imagine them, then implement them in PASCAL, PILOT, BASIC or LOGO. Right now the worlds are fanciful, the blue-sky stuff of fantasy and magic. But by inventing them and building them into programs, you and your kids will gain insight into the real computer worlds that await you. These worlds and the tools to fashion them already exist as blueprints and prototypes in scientists' labs. Before you know it, they will enter your office, your classroom, and your home.

Additional Reading

Anacker, W. (IBM) Computing at 4 Degrees Kelvin. *IEEE Spectrum.* May 1979. Page 26.

Business Week (Cover Story). Artificial Intelligence. March 8, 1982. Page 66.

Business Week (Special Report). The '80s Look in Chips: Custom, Not Standard. January 18, 1982. Page 36D.

Business Week (Cover Story). Japan's Strategy for the '80s. December 14, 1981. Page 39.

Carroll, Lewis. *The Annotated Alice* (Notes and Introduction by Martin Gardner). New York: Clarkson N. Potter, Inc. 1960.

Computer Design Liquid Immersion Technology Cools Supercomputer's Circuits. February 1982. Page 58.

Conway, Lynn and Carver Mead. *An Introduction to VLSI Systems.* Reading, Massachusetts: Addison-Wesley Publishing Company, 1980.

Cromie, William J. and Harold A. Rodgers. The Big Squeeze: Packing More Circuitry into Less Space. *Technology Illustrated.* February/March 1982. Page 55.

D'Ignazio, Fred. (Illustrations by Stan Gilliam.) *Katie and the Computer.* Morristown, NJ: Creative Computing Press. 1979.

D'Ignazio, Fred. *Small Computers: Exploring Their Technology and Future.* New York: Franklin Watts, Inc. 1981.

Matisoo, Juri. The Superconducting Computer. *Scientific American.* May 1980. Page 50.

Science. Are VLSI Microcircuits Too Hard to Design? July 11, 1980. Page 258.

Price, Derek de Solla. 3-D Intelligence. *Science Digest.* January 1982. Page 40.

Sweetnam, George. The Incredible Shrinking Microcircuit. *Science Digest.* January 1982. Page 80.

PILOT And LOGO Addresses

Apple LOGO. Terrapin, Inc., 678 Massachusetts Avenue #205, Cambridge, Massachusetts 02139. (617/492-8816)

Atari PILOT. Atari, Inc., 1196 Borregas Avenue, Sunnyvale, CA 94086. (408/745-2000)

TI LOGO. Texas Instruments Inc., P.O. Box 10508, M/S 5849, Lubbock, TX 79408. (806/741-2978)

Young Peoples' LOGO Association (YPLA), 1208 Hillsdale Drive, Richardson, TX 75081.

Books And Articles

Abelson, Harold. *Logo.* Peterborough, New Hampshire: Byte Books (McGraw-Hill). 1982.

Abelson, Harold and Andrea diSessa. *Turtle Geometry: The Computer as a Medium for Exploring Mathematics.* Cambridge, Massachusetts: The MIT Press. 1981.

BYTE. Special Issue on Smalltalk. August 1981.

Papert, Seymour. *Mindstorms: Children, Computers, and Powerful Ideas.* New York: Basic Books, Inc. 1980.

©

A painless, no-POKE method for mastering Atari high resolution, four-color Graphics from BASIC.

Graphics 8 In Four Colors Using Artifacts

David Diamond
DiamondSoft Co.
Mt. Laurel, NJ

Contrary to what the *Atari BASIC Reference Manual* states, Graphics Mode 8 is a true four-color mode (five colors if you count the border). Other articles have shown you how to obtain 16 or 128 colors by PEEKing, POKEing, and using machine language subroutines to fake out the operating system. This article is different. You can paint with four colors using simple, straightforward BASIC programming.

You probably have noticed that patterns drawn in Graphics Mode 8 often contain spurious colors. Atari stretches your television's resolution to its limits and the extra hues do sneak in.

The spurious colors seem random because they are appearing within a random pattern. They are, however, well behaved. They can be harnessed, controlled, and used for brilliant displays.

Before I get into the details, try the following demonstration program:

```
10 GRAPHICS 8 : COLOR 1
15 R=50
20 FOR X= -R TO R STEP 2
30 Y=SQR(R*R-X*X):REM FORMULA FOR A
   CIRCLE
40 PLOT 100+X, 100+Y : DRAWTO 100+X, 100-Y :
   REM CIRCLE #1
50 PLOT 151+X, 100+Y : DRAWTO 151+X, 100-Y :
   REM Circle #2
60 NEXT X : FOR I=1 TO 350 : NEXT I
70 FOR C=0 TO 15
80 SETCOLOR 2,C,4 : SETCOLOR 4,15-C,8
85 FOR I=1 to 350 : NEXT I
90 NEXT C
```

Surprise! You have five vivid, solid colors on the screen at the same time. Now let's take a look at that program:

Line 10 – Straight Graphics 8. Standard color defaults.

Line 15 – "R" is the radius of a circle.

Line 30 – This is the formula for a circle: $X^2 + Y^2 = R^2$. ("R*R" is a little faster than "R^2").

Line 40 – This draws the first circle. It is vertically cross-hatched to fill it in with a solid color.

Line 50 – This draws the second circle. *But why is it a different color* from the first circle?

Line 20 – Ah, here begins the secret: "STEP 2." Before reading further, change it to "STEP 1," and rerun the program.

Lines 40,50 – Here is the second half of the secret: "100+X" is an *even* offset. "151+X" is an *odd* offset. Change both occurrences of "151" to "150" on Line 50, and see what happens. (Remember to set Line 20 back to "STEP 2").

Lines 70-90 – These lines show you the wide range of color combinations available. Of course, when varying the luminance level, there will be even more.

Alternating Colored Fields

Without any additional programming lines, the circles can easily be changed into beach balls with alternating bands of color. Make sure that Line 20 says "STEP 2," and change lines 40 and 50 as follows:

```
40 PLOT 98+X, 100+Y : DRAWTO 101+X, 100-Y :
   REM Circle #1
50 PLOT 147+X, 100+Y : DRAWTO 150+X, 100-Y :
   REM Circle #2
```

Changing the slope of the cross-hatching by a single horizontal point will add or remove one band of color. Increment the DRAWTO's by one horizontal point, and see what happens:

```
40 PLOT 98+X, 100+Y : DRAWTO 102+X, 100-Y :
   REM Circle #1
50 PLOT 147+X, 100+Y : DRAWTO 151+X, 100-Y :
   REM Circle #2
```

Although the quirk that provides us with the extra colors seems somewhat magical, the reason for the varied solid colors is not. Remember that the "colored-in" areas are really comprised of finely separated, vertical lines. To better see what is happening, spread those lines out into a large grid for easier inspection:

```
10 GRAPHICS 8 : COLOR 1
20 FOR X=10 TO 160 STEP 15
30 PLOT X, 1 : DRAWTO X, 160
40 PLOT 1, X : DRAWTO 160, X
50 NEXT X
```

This isolates your three colors. The even column vertical lines are one color. Odd column vertical lines are a second color. Horizontal lines are the third color. (The background is the fourth, and the border is the fifth).

Line 20 controls the colors. Try "FOR X=10 TO 160 STEP 14" and try "FOR X=9 TO 160 STEP 14".

CALL TOLL FREE
800 424-2738

the PROGRAM STORE

Save $$ on Popular Programs for Atari

LIMITED TIME OFFER

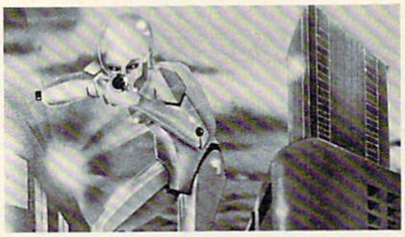
TUMBLEBUG

By Bob Bishop from DataSoft
Travelling through an Egyptian maze, you gather precious Tanna leaves while avoiding the wandering beetles. The exciting element is in the Hi-Res magnification of each area as you progress through the arcade maze.

48K disk...~~$29.95~~ Save 20%

THRU JULY 1 ONLY: pay just $23.96

CROSSFIRE



The aliens have taken over the city and everyone left peacibly -- except you. You are surrounded by the laser firing hordes. In this exciting arcade game your chances of survival are slim, but you can make it -- if you don't get caught in the crossfire!

48K disk...~~$29.95~~ Save 20%

THRU JULY 1 ONLY: pay just $23.96

From Broderbund
As you climb the building, little apples chase you about. You must dig a hole in his path, wait for him to drop in, then bash him over the head with your shovel before he crawls out and gets you! The action is frenzied, and just when you think it's under control, advanced monsters come to get you! Colorful and fun.

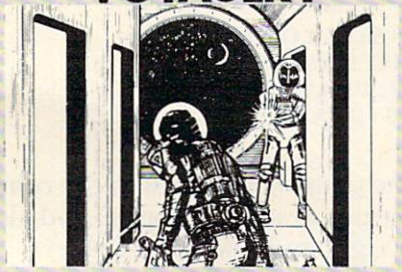
48K disk...~~$29.95~~ Save 20%

NOW THRU JULY 1 ONLY: pay just $23.96

By Bob Bishop from DataSoft
It's like a jigsaw puzzle for Rubic's Cube lovers! Using Hi-Res drawings, the program divides the screen into small blocks. The blocks are then scrambled and you try to reconstruct the picture. Challenging & fun!

48K disk...$29.95

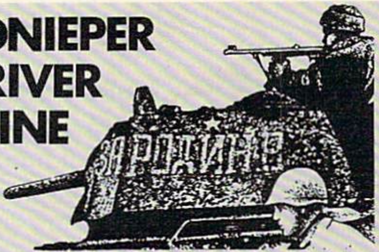
VOYAGER I



From Avalon Hill
A graphic science fiction game that puts you on board a spaceship infested with killer robots! Your job is to clear the four level, 144 location ship of the robots and arm it to self--destruct. The high speed graphics are represented in a 3-D perspective that represents your eye's view. Instant switching to floor plan maps aids your movement. The action is fast and furious!

32K tape...$20.00 48K disk...$25.00

DNIEPER RIVER LINE



From Avalon Hill
It's 1943 and your German troops are dug in along the Dnieper River. Theb rumble of distant artillery fire reinforces your suspicions: the Russians are coming! This WWII simulation is fought using the colorful map, playing pieces and your computer. Can you hold the line?

48K tape...$25.00 48K disk...$30.00

EASTERN FRONT

By Chris Crawford from APE
A map-based simulation of Operation Barbossa, the german invasion of Russia. A complex and accurate war game, EASTERN FRONT pits you, as German Commander, against the terrain, the weather, and the Russian forces. Good use of colorful graphics and sound, and well written documentation make this game a winner!

16K tape or 32K disk...$29.95

MONEY MANAGER

By Andrew Bartorillo from Acorn
This financial management tool can assist you in budgeting, tax deductions and checkbook balancing. Stores up to 130 entries per month (more with additional disk drives) with as many as 99 user-defined categories. One of the most practical home-oriented programs available.

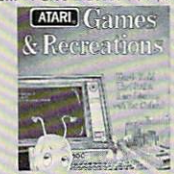
32K disk...$39.95

PASCAL

ATARI PASCAL LANGUAGE SYSTEM

From Atari Program Exchange (APX)
Atari has long promised the release of Pascal. This version has all the major features of this structured language although it is not completely compatible with standard UCSD Pascal. Apparently Atari has released this program through APX because it "...is seriously restricted owing to memory limitations and diskette capacity and performance." The user manual is good for advanced Pascal programmers. Requires two disk drives and a text editor.

48K 2-disk drives..........$49.95
Atari Program-Text Editor...$39.95



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.

Paperback...$14.95

ATARI ASSEMBLER

By Don & Kurt Inman
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 programing in clear, easy steps. Includes full listing and description of 6502 mnemonics and addressing modes.

Paperback...$12.95

Visit our other stores: Seven Corners Center · Falls Church, VA & W. Bell Plaza · 6600 Security Blvd. · Baltimore, MD

TO ORDER CALL TOLL FREE 800 424-2738 For information Call (202) 363-9797

MAIL ORDERS: Send check or M.O. for total purchase price, plus $1.00 postage & handling. DC, MD & VA: add sales tax. Charge cards: include all embossed information on the card.

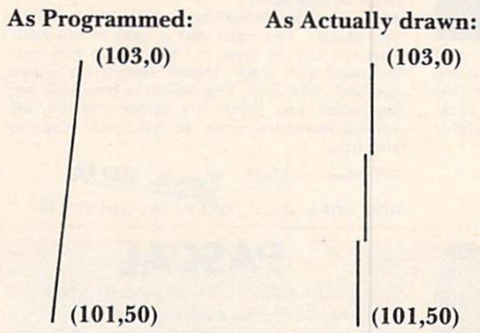
THE PROGRAM STORE
4200 Wisconsin Avenue NW, Dept. U206Box 9609
Washington, D.C. 20016

 Prices Subject to Change

When two adjacent lines touch each other ("FOR X= ... STEP 1"), the two colors blend into the official color for Graphics Mode 8. Another way to look at it is that there are no longer separate lines when they touch, but rather a solid field of pixels.

The Alternating Color Phenomena

The beach ball display, with its alternating bands of color, takes advantage of the fact that, with a pixel matrix, one cannot draw nearly vertical pure diagonal lines. Instead a series of shorter vertical lines are drawn, as shown below:



You can see that the three vertical line segments are drawn on odd, even, and odd columns, respectively, thus alternating colors.

Why Multiple Colors

The horizontal resolution limit of a television set is about 160 unique points. This is because on any one line of the television tube surface there are 160 sets of phosphor points which emit light when struck by the scanning electron beam. Each set actually contains three separate phosphor points – one that glows blue when struck, one that glows green, and one that glows red. Combinations of these dots in various intensities creates the myriad of colors available.

Atari, in order to provide finer resolutions than 160 bytes across, plots 320 points across the screen – two for each set of color dots. (This is referred to as a *half color cycle,* or a *half color clock.*) Thus, even-column points will turn on the left portion of the three color phosphors, and odd-column points will turn on the right portion, producing alternating colors. The effect is referred to as *artifacting.*

Diagonal Lines

Diagonal lines, ranging from vertical to almost 45 degrees, contain vertical components, and are therefore subject to the artifacting effects described above. However, when these lines are drawn on top of a "...STEP 2" solid colored field (such is demonstrated in the above programs), much of the spurious color effect is minimized, so that the "official" color for Graphics Mode 8 will be seen. If the background is dark, a medium intensity line will appear light (whitish). If the background is bright, a medium intensity line will appear dark (often a rich chocolate brown).

The bold splashes of multiple solid colored shapes can thus be combined with the more delicate effects of intersecting diagonal lines, as in the following demonstration program:

```
10 GRAPHICS 8 : COLOR 1
20 SETCOLOR 4, 15, 10 : SETCOLOR 2, 0, 15
30 FOR A=20 TO 140 STEP 2
40 IF A=100 THEN A=101
50 PLOT 65,20 : DRAWTO A,1 : DRAWTO A,A :
   DRAWTO A+30,70
60 DRAWTO 65,A : DRAWTO 30,A+40 :
   DRAWTO 65,20
70 NEXT A
80 FOR I=1 TO 350 : NEXT I
90 FOR COLOR=0 TO 15
100 SETCOLOR 2, COLOR, 5 : SETCOLOR 4,
    15-COLOR, 10
110 FOR I=1 TO 350 : NEXT I
120 NEXT COLOR
```

Moire Patterns

No discussion of multiple colors would be complete without mentioning color moire patterns. There are two types of moire patterns. One type is the secondary pattern produced by the intersection of diagonal lines, such as is illustrated by DEMO Program 2, above. This type is not dependent on color for its effect. The second type is the subtle and delicate designs produced by shifts in color along diagonal lines. This type is dependent on the artifacting effect, and is illustrated in the following program:

```
10 GRAPHICS 8 : COLOR 1
20 FOR A=0 TO 319 STEP 3
30 PLOT 0,159 : DRAWTO A, 0
40 PLOT 319, 0 : DRAWTO 319-A, 159
50 NEXT A
```

Notice that the pattern is whitest in the center, where the lines are not as steeply sloped, and also toward the upper right and lower left corners, where the lines are closest together. In addition to the white and the two artifacted colors, you may notice a fourth and fifth color along the top and bottom sections of the pattern. These extra colors are formed by a *visual* blending of the two artifacted colors. It is caused by the fact that the alternating colored areas are so close together that the eye has difficulty resolving them (a trick used by the Impressionists).

You can combine the various effects discussed in this article. Experiment with different color and intensity combinations. Blend in some dynamic color changes. You have a palette that any artist would envy. ©

Using Atari Joysticks With Your VIC

Christopher J. Flynn
Herndon, VA

Description

What is the most inexpensive peripheral that you can buy for your VIC? A color television? Certainly not. Memory expansion? Probably not. No, a joystick. What? You mean one of those gadgets for playing games? That's right!

Perhaps you didn't realize it, but your VIC can use the very same joysticks that are found on the Atari and Sears video games. Absolutely no hardware modifications are needed at all. Best of all, your local Sears store will sell you a joystick for about ten dollars. How about that! It may turn out, however that the Commodore joystick will differ from the Atari joystick in subtle ways.

Here we will concentrate on showing you how to use an Atari or Sears joystick. Since the VIC joysticks are not available yet (as of this writing), anything we might say about them would only be speculation. So we won't deal too much with what might be.

To give you an idea of the capabilities of the joystick, we've included a demonstration program called Doodle. It's a fast-paced game in full sound and color designed for drawing patterns with the joystick. Your kids will love it – if they can get it away from you.

Before we get into the details, an acknowledgement is due. (Creative Software of California deserves credit for pointing out to me that Atari joysticks are usable on the VIC.)

How We Do It

Figures 1 and 2 compare the VIC joystick socket with the Atari's. The similarities are striking. (Actually, if we ever need to, the Atari joystick can be quite easily rewired. If you take one apart, you'll find that the wires are connected with solderless terminals.)

So let's go with the joystick the way it is. We need to do a little exploratory surgery first. Since I've already done this, please just follow my description. You don't need to do this to your VIC. First we gently open up VIC's case. Armed with our trusty ohmmeter, we trace the joystick connections. We assume that they must reach the 6522 VIA I/O chips. So that's where we start looking. Voila! Tracing all the connections, we find that the joystick switches do indeed go to the 6522s.

Finally, we determine that the joystick is connected as follows:

6522 #1	E	?	?	?	?	?	?	?
	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
6522 #2	?	?	F	W	S	N	?	?
	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0

E, W, S, and N represent the four compass directions. F represents the fire button. We won't be concerned with the fire button in this article.

How do we use this information in a program? What we generally have to do is read each I/O port and test the appropriate bits. Then our program can take any action needed. And there are some complications. Don't forget that the 6522 has data direction registers which program each bit for an input or output operation. Also, the signals from the joystick are in what is called an "active low" state. That is, if the joystick is pointing, say, north, the north bit will be low or zero. The other three directions will be high or ones.

That probably sounds a lot harder than it is. We can actually use BASIC to obtain the joystick readings pretty easily. The BASIC statements shown here are the key to using joysticks on the VIC.

```
POKE 37154,127
V1 = PEEK(37152) AND 128
V2 = PEEK(37151) AND 28
POKE 37154,255
JS = V1/16 + V2/4
JS = (NOT JS) AND 15
```

These statements read the I/O ports and manipulate the bits. We end up with a bit configuration like this:

```
O O O O E W S N
```

The least significant four bits in the variable JS thus correspond to the four joystick switches. Normally, this would mean that JS could range in value from 0 to 15. In practice, JS will take on values from 0 to 10. This is because some bit patterns just aren't possible. With a properly functioning joystick, you can't press the north and south switches at the same time, for example.

The following table shows the values that JS will assume for each of the valid joystick positions.

Direction	JS Value	Delta X	Delta Y
Neutral	0	0	0
N	1	0	-1
S	2	0	1
Can't occur	3	0	0
W	4	-1	0
NW	5	-1	-1
SW	6	-1	1
Can't occur	7	0	0
E	8	1	0
NW	9	1	-1
SE	10	1	1

Note that JS is 0 in the neutral position. This gives us a handy way to test for joystick movement.

Delta X and Delta Y are variables which will help us if we're trying to move an object around the screen. Suppose we're using an X and Y coordinate system like this:

```
              X
   0 1 2 3 4 5 ... 21
   1
   2
Y  3
   4
   .
   .
   .
   22
```

Y represents a row number and X represents a position within a row. When the joystick moves, we want to update the values of X and Y so they indicate the new position. We can do this again easily in BASIC:

```
X=X+DX(JS)
Y=Y+DY(JS)
```

DX and DY are arrays where we've saved the list of values for Delta X and Delta Y.

An example will show how this works. Let us assume that we have an object at X=7 and Y=5. We test the joystick and determine that it has moved. Let's assume that it's pointing north. From our table, we know that JS will contain 1. So, the new positions of X and Y will be:

```
X=7+DX(1)
Y=5+DY(1)   or

X=7
Y=4
```

Thus, our object is moved up one line closer to the top of the screen. There was no forward or backward horizontal change.

One last detail we need to think about is how to convert X and Y into something VIC understands. As you know, we can POKE things into VIC's screen memory. But we need a memory location for that. Again, BASIC helps us out:

```
P=22*Y+X
```

That little formula will convert valid X and Y values into a number ranging from 0 to 505. Next, we must add P to the screen and color memory starting locations:

```
POKE 7680+P, code
POKE 38400+P, color
```

Use any screen code and color that you wish.

Doodling

We've covered joysticks pretty quickly; we've only discussed the highlights. There are many other details involved. The best way to pick these up is to study Program 1 and to enjoy the Doodle game.

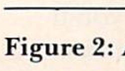
Doodle is a lot of fun to play. The object is just to enjoy yourself. When you start Doodle, it will display instructions on how to use the special function keys.

Key	Message	Description
f1	TO QUIT	Ends the game.
f3	MOVE CURSOR	The cursor moves, but does draw a line. Erases any objects that it crosses.
f5	DRAW LINE	The cursor moves and draws a line.
f7	CLEAR SCREEN	The screen is cleared and the cursor is centered. VIC is ready to doodle again.

You may press any key at any time while doodling. For interesting effects, alternate the f3 and f5 keys. By doing this properly, you can enclose a figure within another figure without any intersecting lines.

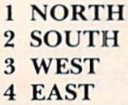
Comparison Of VIC And Atari Joystick Sockets (as viewed from the outside)

Figure 1: VIC Joystick Socket



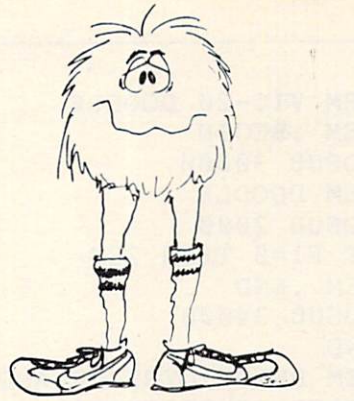
1 JOY 0	6 LIGHT PEN
2 JOY 1	7 +5V
3 JOY 2	8 GROUND
4 JOY 3	9 POT X
5 POT Y	

Figure 2: Atari Joystick Socket



1 NORTH	6 FIRE BUTTON
2 SOUTH	7 NO CONNECTION
3 WEST	8 GROUND
4 EAST	9 NO CONNECTION
5 NO CONNECTION	

VIXEL™



Vixel Volume One

Fire

Fly a water-dropping helicopter, and try to put out the high-rise fire before it spreads.

Draw

Be an artist! This high-resolution drawing program makes it easy to create pictures on the screen, and then save them on tape.

Race

Race the computer, head-on! Simple but lots of fun.

Vixel Volume Two

Superfont

Design programmable characters on your VIC-20 with this easy-to-use program. Especially useful when creating animations, since you can edit four characters at once as a 2×2 block. SUPERFONT creates DATA statements after you have finished designing the characters.

Safari

You are a photographer on an African Safari in this great game. The jungle animals run past as you try to snap their pictures. An excellent example of how to use large blocks of programmable characters on the VIC to create animation effects.

Quix

How good is your memory? QUIX presents patterns of color and sound that gradually get longer and harder to remember.

The programs Fire, Draw and Race are available on VIXEL Volume One for only $12.95. The programs Superfont, Safari and Quix are available on VIXEL Volume Two which is also $12.95. Both Vixel #1 and Vixel #2 work with the standard 5K VIC-20. Foreign orders add $3.50 for shipping. CA residents add 6% tax. VISA and MasterCard welcome. Please add $1.00 shipping for credit card orders.

(805) 683-1585 THE CODE WORKS Box 550
Goleta, California 93116
USA

```
100 REM VIC-20 DOODLE
200 REM .BEGIN
210 GOSUB 30000
220 REM DOODLE
230 GOSUB 2000
240 IF F1=0 THEN 230
250 REM .END
260 GOSUB 34000
270 END
300 REM READ JOYSTICK AND K.B. (***
    ATARI JOYSTICK***)
310 POKE DD,127
320 V1=PEEK(R1)AND128
330 V2=PEEK(R2)AND28
340 POKEDD,255
350 JS=V1/16+V2/4
360 JS=(NOT JS)AND15
370 GET A$:IF A$<>"" THEN GOSUB 400

380 RETURN
400 REM SERVICE K.P.
410 A=ASC(A$)
420 IF A=133 THEN F1=1
430 IF A=134 THEN CH=32
440 IF A=135 THEN CH=32+128
450 IF A=136 THEN GOSUB 32000
460 RETURN
500 REM CHOOSE COLOR
510 CL=INT(RND(1)*8)
520 IF CL=1 THEN 510
530 RETURN
600 REM VERIFY X&Y
610 IF X<0 THEN X=0
620 IF X>21 THEN X=21
630 IF Y<0 THEN Y=0
640 IF Y>22 THEN Y=22
650 RETURN
700 REM X&Y TO ADDR
710 P=22*Y+X
720 POKE VA+P,BT
730 POKE CA+P,CL
740 RETURN
800 REM SET NOISE AND  VOLUME
810 POKE VL,(3+INT(RND(1)*6))
820 POKE S3,(128+INT(RND(1)*110))
830 RETURN
2000 REM DOODLE
2010 TL=TI+60*2.5
2020 GOSUB 300:REM POLL JOYSTICK
2030 IF F1 THEN RETURN
2040 IF JS<>0 THEN 2070
2050 IF TI<TL THEN 2020
2060 GOSUB 800:GOTO 2010
2070 POKE VL,15:POKE S3,220
2080 FOR Z=1 TO 100:NEXT
2090 POKE S3,0:GOSUB 800
2100 REM CLEAR OR FILL CURSOR SPOT
2110 BT=CH
2120 GOSUB 500:REM GET COLOR
2130 GOSUB 700:REM STORE BT
2140 REM NEW CURSOR POSITION
2150 X=X+DX(JS)
2160 Y=Y+DY(JS)
2170 GOSUB 600:REM CHECK X&Y
2180 REM SET CURSOR
2190 BT=CS:CL=0
2200 GOSUB 700:REM STORE BT
2210 RETURN
30000 REM .BEGIN
30010 PRINT CHR$(147);
30020 PRINT SPC(8);"VIC-20"
30030 PRINT
30040 PRINT SPC(5);"D O O D L E"
30050 PRINT:PRINT CHR$(158)
30080 PRINT CHR$(31):PRINT
30090 PRINT "PRESS:":PRINT
30100 PRINT "F1- TO QUIT"
30110 PRINT "F3- MOVE CURSOR"
30120 PRINT "F5- DRAW LINE"
30130 PRINT "F7- CLEAR SCREEN"
30140 PRINT:PRINT
30150 PRINT"ATARI/SEARS JOYSTICK"
30160 PRINT"PLUGGED IN ?"
30170 REM VARIABLES
30180 REM JOYSTICK
30190 DD=37154:R1=37152:R2=37151
30200 REM VIDEO AND SOUND
30210 VA=7680:CA=38400:BG=36879
30220 VL=36878:S3=36876
30225 CS=90:CH=32+128:Z=RND(-TI)
30230 REM DELTA X, DELTA Y FOR JOYSTI
  CK
30240 DIM DX(10),DY(10)
30250 FOR I=0 TO 10:READ DX(I):NEXT
30260 FOR I=0 TO 10:READ DY(I):NEXT
30270 DATA 0,0,0,0,-1,-1,-1,0,1,1,1
30280 DATA 0,-1,1,0,0,-1,1,0,0,-1,1
30290 FOR Z=1 TO 4000:NEXT
30300 REM INITIAL CONFIGURATION
30310 POKE BG,25:REM WH-WH
30320 GOSUB 800:REM GET NOISE
30330 GOSUB 32000:REM CLEAR
30340 RETURN
32000 REM CLEAR SCREEN
32010 PRINT CHR$(147);
32020 X=10:Y=10:BT=CS:CL=0
32030 GOSUB 700
32040 RETURN
34000 REM.END
34010 PRINT CHR$(147);
34020 POKE BG,27
34030 PRINT:PRINT
34040 PRINT "SO LONG!"
34050 PRINT:PRINT
34060 POKE VL,0:POKE S3,0
34070 RETURN
```

©

This program was written in Microsoft BASIC. We have included notes on adapting it to the Atari.

Analysis Of Variance

Anselm Wachtel
Pittsburgh, PA

Suppose you wanted to find out which department store chain sells some item at the lowest price. You could simply go to a store representing each chain and compare prices. Or for greater accuracy, you could go to a number of stores of each chain and compare the averages. But now there's a problem: the differences between individual stores of any one chain may differ from each other by about as much as the averages differ from each other. What can you now say about the pricing by the chains? Those differences may simply reflect normal scatter, i.e. chance, and therefore be insignificant.

This problem is very difficult to handle by "inspection," but is readily solved by a *One-Way Analysis of Variance* (ANOVA). Without going into the details of statistics, let's just say that this technique compares the scatter of data within each group of data (in our case the prices of individual stores within any one chain) with the overall scatter of all data. Scatter (variability) is measured by what is called variance. It is estimated by summing the squares of differences between data and the average, and dividing the sum by the number of data, minus one. Number -1 is called the degrees of freedom, abbreviated DF in the program.

The program subtracts the "treatment sum of squares" from sum of squares of the overall mean to arrive at an "error sum of squares" which, divided with its degrees of freedom, represents the variance due to chance.

The Degree Of Confidence

Finally, the ratio of variances, associated with treatments and error, yields the *F-statistic*. It's up to you to decide on the degree of confidence, i.e. in how many cases out of 100 future pricings, you can expect the differences between chains to be real. You need a table of F-values which you enter with the number of degrees of freedom associated with each statistic and find a number. If your F value is greater, then the chains can be expected to be different in, say, 95% of future pricings. Naturally, the more confidence you demand, the less chance you have of finding that your results are significant to that degree.

I have deliberately structured the program to require data entry with DATA statements rather than INPUT. The DATA become part of the program and can be edited easily. Simply type:

line # DATA xx,xx,xx,xx,999,yy,yy,yy,999,zz,zz,zz,zz, zz,999,9999

and RUN. xx,yy,zz represent individual prices with any one chain contained within the 999s.

Instead of using a table, you might wish to use the F-distribution program in *Some Common Basic Programs*, page 140, by Lon Poole and Mary Borchers (A. Osborne and Assoc., Inc.). Entering your F-value and degrees of freedom returns the confidence level directly (called percentile). Naturally, this could be incorporated as a subroutine.

Atari Notes

These are the modifications necessary to adapt Analysis of Variance to the Atari:

```
115 DIMSP$(40):SP$=" ":SP$(40)=" ":SP$
    (2)=SP$
140 ?"---------------":REM 15 DASHES
150 (DELETE)
300 ?"T";K;"=";INT(H*100+.5)/100
370 ?"SOURCE";SP$(1,6);"SSQ";SP$(1,9);"DF";
    SP$(1,7);"MS"
380 (DELETE)
400 ?" CRUDE";:POKE85,8:PRINT Q1;:
    POKE85,23:?N1
410 ?" COR.F";:POKE85,8:PRINTG;:POKE85,
    24:?1
420 ?" TOTAL";:POKE85,8:PRINTC;:POKE85,
    23:?N1-1
430 ?" TREAT";:POKE85,8:PRINTT2;:POKE85,
    23:?D1;:POKE85,31:?INT(M1*100+.5)/100
440 ?" ERROR";:POKE85,8:?E;:POKE85,23:
    ?D2;:POKE85,31:?INT(M2*100+.5)/100
460 ?"F(";D1;" AND ";D2;" DEGREES OF
    FREEDOM)=";INT(F*100+.5)/100
```

```
0 PRINT"{CLEAR}":GOTO480
100 REM ONE WAY ANALYSIS OF VARIANCE
110 REM A. WACHTEL, PITTSBURGH, PA 15235
120 PRINT"{CLEAR}"
130 PRINT"TREATMENT MEANS"
140 PRINT"###############"
150 DEF FNA(X)=INT(X*100+.5)/100
160 S1=0:Q1=0:T1=0:N1=0:K=0
170 N=0:S=0:Q=0
180 READ Y
190 IF Y=999 THEN 250
200 IF Y=9999 THEN 320
210 S=S+Y
220 Q=Q+Y*Y
230 N=N+1
240 GOTO 180
250 S1=S1+S:Q1=Q1+Q:N1=N1+N
260 H=S/N
270 T=S*S/N
```

```
280 T1=T1+T
290 K=K+1
300 PRINT"T"K"="FNA(H)
310 GOTO 170
320 G=S1*S1/N1
330 C=Q1-G:T2=T1-G:E=C-T2
340 D1=K-1:D2=N1-K
350 M1=T2/D1:M2=E/D2:F=M1/M2
360 PRINT
370 PRINT"SOURCE";SPC(6);"SSQ";SPC(9);"DF";
    SPC(7);"MS"
380 PRINT"######";SPC(6);"###";SPC(9);"##";
    SPC(7);"##"
390 PRINT
400 PRINT" CRUDE";TAB(8)Q1;TAB(23)N1
410 PRINT" COR.F";TAB(8)G;TAB(24)"1"
420 PRINT" TOTAL";TAB(8)C;TAB(23)N1-1
430 PRINT" TREAT";TAB(8)T2;TAB(23)D1;TAB(31
    )FNA(M1)
440 PRINT" ERROR";TAB(8)E;TAB(23)D2;TAB(31)
    FNA(M2)
450 PRINT
460 PRINT"F("D1"AND"D2"DEGREES OF FREEDOM)=
    "FNA(F)
470 GOTO 530
480 PRINT"USE LINE 0 AND LINES UP TO 119 TO

490 PRINT"ENTER DATA. PLACE 999 AT THE END
500 PRINT"OF EACH TREATMENT SERIES.
510 PRINT"PLACE 9999 AFTER THE LAST 999.
520 PRINT"(AVOID 999 OR 9999 AS DATA).
530 END
```

©

Atari Dice Simulation

W. C. McLachlan
Dallas

So you wanted to program a board game utilizing dice? Well, you've come to the right place: Atari's graphics and string handling capabilities make it easy.

It is possible to simulate dice in graphics modes 3 through 7 (even 8), but it is cumbersome. It is much easier and faster in mode 0. The following program uses one large string (DICE$) to store the die faces one through six. Each die is divided into nine parts. By effectively utilizing Atari's cursor controls, we can build the parts into a realistic looking die.

```
100 GR.0
110 DIM DICE$(102): POKE 752,1:REM TURN
  OFF CURSOR
120 DICE$="AAACDDDABACDDDAAABAACDDDAAACD
DDAABBAACDDDABACDDDAABBAACDDDAAACDDDBABB
AACDDDABACDDDBABBBBCDDDAAACDDDBBB"
130 X=20 : Y=10: REM X AND Y ARE INITIAL
  DIE POSITION
140 DICE = INT(6*RND(0)+1): REM RANDOM N
  UMBER
150 A=DICE*17-16:REM A=SUBSTRING START P
  OSITION
160 COUNT=COUNT+1
170 IF X>25 THEN X=20
180 POSITION X,Y
190 PRINT DICE$(A,A+16): REM PRINT THE S
  UBSTRING
200 X=X+5: REM MOVE OVER 5 SPACES TO PRI
  NT NEXT DIE
210 SO. 0,10,2,15:S0.0,0,0,0: REM SOME D
  ICE ROLLING NOISE
220 IF COUNT <> 20 THEN 140
```

To enter line 120, the following code prevails:

A = (REVERSE VIDEO) SPACE
B = (REVERSE VIDEO) CTRL-. (PERIOD)
C = ESC-CTRL-DOWN ARROW
D = ESC-CTRL-LEFT ARROW

You'll note that the first 17 characters constitute die face #1, the second 17 – die face #2, etc. All six faces *will* fit on one logical line, although it's a tight fit. Now to the program:

Line 140 selects a number from one to six. After the number is selected, it will be necessary to tell the computer what part of the string contains the die face corresponding to it. This is the function of line 150. Next, the substring is printed (line 190) at the position determined in line 180. Since two dice are displayed, lines 170 and 210 are required to select the die position. Lines 160 and 220 combine to roll each die ten times to simulate a dice "throw." Easy!

This routine has been expanded just to make it clearer. You can compact it to four logical lines and store it away as a subroutine in your programs. [*The word* logical *means "as the computer would see it." In other words, four "logical lines" might break into seven or eight lines on your TV because it can only show 40 characters per line. There would still only be four "logical lines," however, since there would only be four line numbers (to the computer). Sometimes, for example, sector 15 on a disk might be located* physically *before sector 14. They would still be* logical *14,15.*]

Other additions, such as adding more dice, or using joysticks to initiate a dice roll, are made easily.

Atari BASIC allows for extremely fast string handling. This is just one example of how to use a string to its fullest advantage. Remember, you can use any of the edit commands (tab, cursor up, cursor right, backspace, line delete, etc.) in a string by first pressing the ESC key. Usually, you can save a considerable amount of time and unnecessary program steps by employing string edit commands. ©

Jim Butterfield responds to a number of recent queries about Commodore computers.

PET Miscellany

Jim Butterfield
Toronto

Q: How do I get my CBM disk to do things (like direct access) from machine language?

A: The disk doesn't know or care who's giving it instructions: BASIC or machine language. All you really need to do is to send to the disk (or receive from disk) exactly the same information that BASIC would send. Some commands go to the command channel (secondary address 15) and some to the data channel, so you'll need to keep things sorted out and know the command channel formats. See a separate article on Machine Language and the CBM Disk [**COMPUTE!**, *March, 1982, #22, pg. 139]*.

Q: A previous article says that SYS 54386 on 4.0 systems will *call* the Machine Language Monitor (as opposed to *breaking* to the Monitor). What is the BASIC Upgrade ROM address for this? I'd like to get that C* to show instead of that B*.

A: Upgrade ROM may call the monitor with SYS 64785. It's not needed quite so badly in earlier ROMs since they do not turn off the CMD that's in effect.

Q: I see that memory locations 1001 to 1012 decimal are used in the Fat 40. Are there any that are useful to know to the average programmer? For example, what does "New Key Marker," location 1001, do?

A: The interrupt working values in the Fat 40 aren't really too important to the end user; you need to know that they're there so that you can leave those locations alone. I don't plan to comment in any depth on them ... if you want to play, be my guest. The New Key Marker spots if a new key is pressed, so that when you hold down cursor-right and then press cursor-down instantly, it still knows to pause before taking off again. (Note that it goofs up if you use the shift key to change cursor-left to cursor-right.)

Q: More questions on memory maps. The 80 column machine has a lot of constants used near the top of zero page for windows and things. What is this space used for in the 40 column ones?

A: The 80-column locations are mostly used in the 40-column job for the "line wrap table" which keeps track of single versus double lines. You don't need to do this in 80 columns, of course.

Q: I have a screen dump program which prints a hard copy of everything on the screen to the printer. At the end of each line it sends CHR$(0) five times before it proceeds to the next line. Why?

A: Some printers need time to return the carriage to the left hand side of the page. To allow for this, some programs add "pad" characters after the RETURN/LINEFEED to give this time; CHR$(0) doesn't print, but fills in the time.

Q: What is the difference between ASCII and "true ASCII"?

A: True ASCII is a standard 7-bit code. Upper case A is decimal 65, and lower case a is decimal 97. In the PET, an 8-bit code is used; if you're in Text mode then upper case A is 193 and lower case a is 65. To go to a standard device or a communications line, you'll need to do a translation from PET-ASCII to true-ASCII.

Q: I have a program which does a "soft" change of the device number of a particular device. How does it do it?

A: Some devices – notably disk units – place their device ID's into RAM at the time of power-up. If you can download to the device's memory, and know the locations, you can change these memory values and, thus, the disk ID.

Q: I'd like to be able to check to see if the disk is busy. If the disk is doing a Header (New), Collect (Verify), or other lengthy jobs, I'd like to know when it is finished. Trying to use the disk when it's busy causes my program to lock up until the disk is free.

A: Try using the following code:

```
POKE 59456, PEEK(59456) AND 251
X=PEEK(59456) AND 64
POKE 59456, PEEK(59456) OR 4
```

At this point, X will have a value of zero if the disk is busy, and 64 otherwise. ©

Ms. Deal has been working with Commodore machines for several years. Many of the tidbits below represent questions people have asked her. You're bound to learn something you didn't know.

Bits And Pieces

Elizabeth Deal
Malvern, PA

• It's sometimes hard to tell where the screen boundaries are. Annoying, if the cursor jumps the line or the screen scrolls when you don't want it to. Cheap solution: fill screen with the full white square character and put borders on the picture using 1/8" or 1/16" vinyl tape sold in stationary or art supply stores. Do it on the right and bottom edges of the screen. Watch for parallax!

• Don't use SYS4 within a running BASIC program (to get to the monitor). Location 4 is busy at that time and you'll crash. Use SYS1024 or *call* the Monitor instead (64785 on Upgrade).

• Most, if not all, tape load errors are preventable. Keep tapes clean. Clean the cassette unit immediately after getting the first VERIFY ERROR or a bad ST value. Use Radio Shack freon head cleaner or grain alcohol and Q tip. No rubbing alcohol. Don't permit the Q-tip to get tangled up in moving parts inside.

• If you can't find any pencils in the house turn to your PET. They are all under there. A real "world computer" would have legs either too small for pencils to roll under or big enough for a person's hand, if you want my opinion.

• Recent Commodore printers (4022) have a trace of descenders and work quietly, but slower, than the old ones. It is easier to flip character sets than previously. They don't get stuck any more. Surprise – the values are opposite from those in the PET itself. Quote from the manual: to set text mode (upper and lower case) 10 poke59468,12: open7,4,7: print#7: close7, and to set graphics mode (graphics and upper case) 10 poke59468,14: open8,4,8: print#8: close8. Note that two secondary addresses are used, 7 and 8.

• Both Osborne *PET/CBM GUIDES*, the red and white books, incorrectly describe several aspects of array storage on the PET. The general logic is all right, but many details have been mangled. BASIC 4 users should be aware of the fact that their character strings occupy two more bytes of storage per string than previous BASICs. All users should note that the Osborne books, in text and/or some illustrations, reverse the low-high order of addresses, and that the description of storage of character strings is in error.

• Update on Partitions: to set up partitions in the PET, it is always necessary to reset the "beginning of BASIC" pointer and to put zeros in the first three bytes of a partition (NEW). Assuming two partitions, the way to get going is via the monitor. For instance, to set up a partition at $6000 (24576 decimal), type SYS4 and:

```
.M   0028 0028
.:   0028 01 60 03 60 03 60 03 60
.M   6000 6000
.:   6000 00 00 00 xx xx xx xx xx
```

To reenter a previously established partition, it's a good idea to check the presence of that zero in the first position, otherwise BASIC can't work. Do the check while in the Monitor changing your pointers. Otherwise, if you exit with the zero missing, BASIC will not function and you probably won't be able to get back into the monitor without a reset (power turned off, then back on).

Existing program files can be loaded into a partition using "Toolkit" 's APPEND command (tape only). Existing programs in ASCII format can be so loaded via the XEC command of "Power" (tape and disk).

Saving programs from a partition has to be done by the Monitor's .S command in case of the Toolkit. In Power, saving a program as an ASCII file (relocateable, by definition) does the job automatically.

For many applications a method described in **COMPUTE!**, November, 1981, #18, "Inverse Partitioning" should simplify the task.

• Supermon (SM) [*a machine language monitor extension program which appeared in* **COMPUTE!**, *December, 1981, #19*] is handy in resetting the top of BASIC pointer in case of continuing shrinkage of your PET's memory while you POKE and rePOKE some undebugged machine code program. Assume that SM is the SYS address given by the loader during original setup. Doing SYS SM at any time will set the pointers to the original condition. By the same token, if you have some code below (lower address) Supermon, don't use its SYS command, use PET's SYS4. PET recognizes Supermon's commands by checking a pointer at $03FA-03FB. If you crash and reset the PET, this pointer has to be fixed by SYS SM.

• Wedge relocates itself to the top of the PET, next to the screen. The code uses ROM routines and contains no location-sensitive addresses. It can be block-moved anywhere in memory with the Supermon's Transfer command, so long as you notify the CHRGET routine of the move: change $0071-0072 (low byte-high byte of Wedge's address) to chips...chips...chips...chips...chips...chips...

Command-O or Command-O-Pro?

It's called the Command-O-Pro in Europe, Command-O in the U.S.

But whatever you call it, this 4K byte ROM will provide your CBM BASIC 4.0 (4016, 4032) and 8032 computers with 20 additional commands including 10 Toolkit program editing and debugging commands and 10 additional commands for screening, formatting and disk file manipulating. *(And our technical writer dug up 39 additional commands in the course of doing a 76-page manual!)*

Extends Commodore's 8032 advanced screen editing features to the ultimate. **SCROLL** up and down, insert or delete entire lines, delete the characters to the left or right of the cursor, select **TEXT** or **GRAPHICS** mode or ring the PET's bell. Redefine the window to adjust it by size and position on your screen. And define any key to equal a sequence of up to 90 key strokes.

Resides in hexadecimal address **$9000**, the rightmost empty socket in 4016 and 4032 or the rearmost in 8032. If space conflicts, Socket-2-ME available at a very special price.

Command-O from Skyles Electric Works $75.00
Complete with **Socket-2-ME** 95.00

Limited Time: FREE Disk-O-Pro.

Special: You can get this Disk-O-Pro **absolutely free**... when you order the new Commodore 2031 single disk drive from Skyles. (See page 8.)

"The PET with the new PicChip...is like home movies."

The new ROM that took Europe by storm is available only from Skyles Electric Works in the U.S. and Canada.

PicChip, a ROM extension of the BASIC Version III (BASIC 2), BASIC 4.0 or BASIC 8032 interpreter. Over 40 commands that allow you to create programs with dynamic graphics displays: plots, bar graphs, pictures; and rolling, scrolling, shifting and inverting. All instantly and easily added to your BASIC program.

The address for the 2000/3000 (which would require PicChip module PC2), for the 4000 (PC4) and for the 8000 (PC8) is $A000...unless you have a Mikro, Word Pro 3 or 4, or Jinsam, which occupy that same address. In those cases, you will need the PicChip on an interface board... for the 2000/3000 series, PCB2, for the 4000, PCA4, and for the 8000, PCA8. In all cases, the Mikro or WordPro would be switchable manually using the Skyles Socket-2-ME...available at a very special price.

PicChip from Skyles Electric Works $60.00
(Please indicate PC2, PC4, PC8)
Complete with Socket-2-ME 80.00
(Please indicate PCB2, PCA4, PCA8)

Disk-O-Pro ROM:

25 new commands at $75.00, that's only $3.00 a command!

The powerful chip for all "classic" PETs. In any PET with Version III (BASIC 2.0) ROMs (### **COMMODORE BASIC** ###) Disk-O-Pro will give 19 software compatible disk instructions*: 15 identical with the new BASIC 4.0 (or with 8032 ROMs) compatible with both old and new DOS. Plus 4 additional disk commands...including program appending (**MERGE**), program overlaying (**MERGE #**) and **PRINT USING**, allowing formatting output of strings and numbers on the PET screen or on any printer.

* **NOTE:** *Old DOS doesn't recognize 3 commands.*

Plus softtouch key (**SET**) which allows you to define a key to equal a sequence of up to 80 keystrokes; **SCROLL** whereby all keys repeat as well as slow scrolling and extra editing features; **BEEP** which allows you to play music on your PET.

Completely compatible with the BASIC Programmer's Toolkit. *(The chip resides in the socket at hexadecimal address* **$9000**, *the rightmost empty socket in most PETS.)* And for the owners of "classic" (or old) PETs, we do have interface boards.

(For those owning a BASIC 4.0 or 8032, even though the Disk-O-Pro may not be suitable, the Command-O is. We have never abandoned a PET owner.)

Complete with 84-page manual written by Greg Yob.

Disk-O-Pro from Skyles Electric Works $75.00
Complete with **Interface Board** (for "Classic" PETs) 95.00

The New Mikro:

Now you can program in machine language as easily as writing BASIC.

4K machine language assembler ROM plugs into your main board. Does all the machine language work for you; all you have to do is start laying down the code.

Retains all the great screen editing features of the PET... even all the Command-O or Toolkit commands. Write your own machine language subroutine. The program you write is the source code you can save. Machine language monitor saves the object code. Not as professional as the Skyles MacroTeA...not as expensive, either.

A great learning experience for those new to machine language programming but who want to master it easily. Superb 50-page manual—including brief introduction to machine language programming—included. But we also recommend several of the books in the Skyles Book Barn.

Skyles Mikro Machine Language Assembler
MK 2/3 for 2000 or 3000 Series PETs $80.00
MK 4/8 for 4000 or 8000 Series PETs and CBMs
Complete with **Socket-2-ME** 95.00

Skyles Catalogue Page 1

For PET and CBM owners only:

This is just 1 of 20 pages of the newest and biggest Skyles catalog, hot off the press.

We know you'll want this page, in its full 8½ x 10 splendor, and another 19 pages of peripherals, software and books that will make your PET or CBM computer even nicer to live with. So, if we missed sending you your very own copy within the last few weeks, call us at **(800) 227-9998** (unless you live in California, in which case call **(415) 965-1735.**

From Skyles Electric Works, the oldest and largest professional specialists in the business.

Skyles Electric Works
231-E S. Whisman Road
Mountain View, California 94041

point to the new location.

While Wedge is loading itself it uses some screen area for working storage (evidenced by a quick flash on the screen). This can be deadly if you plan to put Wedge below some existing code, as the code will turn to mush. There are several ways around it: use the old DOS-Support program which doesn't jump its boundaries, or load the Wedge first to a lower location, followed by the desired program, or load the Wedge next to the screen and move it.

- Wedge is hard-coded to be available only with disk as device 8. Untested suggestion: it can be used with a differently numbered disk device if you poke the device number in two locations in the Wedge code. Use Supermon's Hunt command to find two 8's. They are in A9 08 (LDA #$08) code sequence. No other 8's exist in the code.

- Wedge's handy curiosity: you can quickly obtain the names of the floppies and bytes free from both drives by saying such things as ›$10, or ›$01, or ›$55 or $›XX or whatever.

- While developing a long BASIC program (or when putting it away for some time) it's a good idea to document it. Otherwise, later it won't seem to resemble any of your thoughts. The meanest thing is the structure of the program – "what does it do and when?" You can't modify a program without knowing that you won't stick a line in a place that breaks up a subroutine or without knowing where it might even be executed.

A cross-reference program is a good thing to use, for instance Cursor (tape-magazine) published Jim Butterfield's fast, machine code Cross-ref routine for disk users (only, I think). It lists variables (in alphabetical order) and lines (in numeric order) indicating places where those variables or numbers are used.

Here is a halfway approach I sometimes use. It's good for tape people. And it can, of course, be used with Cross-reference for a complete picture.

Devoting some space at the end of a program to housekeeping, I can list all subroutine entry points and all GOTO references, like this:

```
2000  ‹S›:ON S GOSUB 450,500,510,320,620,800
2001  ‹G›:ON G GOTO 60,100,150,250,455
```

The syntax in ‹S› and ‹G› guarantees against those lines ever executing, even if I were to lose control of the program, since a SYNTAX ERROR would result. ON X GO X, however, is seen by me, Cross-reference, Toolkit and Power as a perfectly valid list of addresses. They get renumbered correctly and, *if* I keep them clean and up to date, I stand a better chance of knowing how the program is built. It's primitive, but it works. ©

PET Newsletters And Magazines

Richard Mansfield
Assistant Editor

There are several magazines, besides **COMPUTE!**, which feature articles and programs for Commodore computers. Many of them are devoted exclusively to the coverage of PET/CBM machines. Here is a list, in alphabetical order:

1. *Commodore,* Commodore Business Machines, 681 Moore Road, King of Prussia, PA, 19406. Published bi-monthly. $15 per year, $25 Canada and Mexico.

2. *The Midnight Gazette,* (Published by the Central Illinois PET Users' Group), CIPUG, 635 Maple, Mt. Zion, IL 62549. The newsletter is financed by donations and ads from readers. You can send up to four, double-stamped (40 cents) self-addressed, long envelopes to receive the next four issues. Quarterly.

3. *The Paper,* Pearl St., Livingston Manor, NY, 12758. Published six times a year, a single issue is $4. A year's subscription is $20.

4. *TORPET,* 381 Lawrence Avenue West, Toronto, Ontario, Canada M5M 1B9. Newsletter of the Toronto PET Users' Group. $1 per issue. Quarterly.

5. *The Transactor,* Commodore Business Machines Limited, 3370 Pharmacy Ave., Agincourt, Ontario, Canada, M1W 2K4. Official publication of Commodore Canada. All back issues, $35. Current volume (six issues per year), $10. Bi-monthly. ©

SYSRES™

THE ULTIMATE RESIDENT PROGRAM MANIPULATION SYSTEM FOR PET™/CBM™ MICROCOMPUTERS

SO COMPLETE, EVEN THE BEST OF THE COMPETITION DOESN'T COMPARE!

EXTENDED DOS SUPPORT

@ (type "N" keyboard)
< (type "B" keyboard)
! (original keyboard)
> (for 'wedge' users)

These commands may be used interchangably, to perform the following dos support functions.

Command	Function
@	Display disk status / send command
@N	Format (header) a new diskette
@I	Force initialize diskette
@V	Validate diskette (collect)
@D	Duplicate diskette
@C	Copy or concatenate disk file(s)*
@R	Rename file
@S	Scratch file(s)*
@$	List directory*
@U:	Reset disk drive
@L	List disk file or BASIC program*

* Added/enhanced disk command.

EXTENDED EDITOR

Command	Function
/	Quick load from disk
↑	Quick load from disk with auto run
APPEND	Append from disk to end of current program
AUTO	Auto line number (allows header)
BLOAD	Load machine language (binary) file
BRUN	Load and execute machine language program
CHANGE	Change pattern to another pattern
CLOSE	Close one or all files
CMD	Set output to file (does not send "READY.")
DELETE	Delete a range of lines from program
DUMP	Dump all scalar variables to screen or file
EXEC	Execute a file as keyboard commands
FIND	Find occurances of a pattern
GET	Read a sequential file into editor
KEY	Define a key as a special function
KEYS	Turn key functions on
KILL	Disable SYSRES™
KILL*	Disable SYSRES™ and unreserve memory
LIST	Improved BASIC LIST command
LOAD	Defaults to disk drive
MERGE	Merge from disk into current program
MON	Break to current machine language monitor
OLD	Restore program after "NEW"
PUT	Send program to disk as text file
RENUMBER	Renumber all or part of program
RUN	Run current program, ignores screen garbage
SAVE	Defaults to disk drive, allows replace
SETD	Set disk device #, allows multiple drives
SETP	Set printer channel, format mode, paging
TRACE	Select 1 of 3 trace/step modes and speed
VERIFY	Compare current program against disk/tape
WHY	Print position of last error
WHY?	List line of break or error
*	Send output to printer
#	Display current version of SYSRES™

COMPARE SPECIFICATIONS!	SYSRES™	POWER™
Number of ADDED commands	33	13
Number of IMPROVED BASIC commands	7	none
Number of DOS SUPPORT commands	11	none
Approximate added syntax options	1200	60
Instruction manual length	86 pages	75 pages
Instruction manual style	structured	conversational
Re-loactable?	yes	no
Use on more than one (any) PET/CBM™	yes	no
Upgradable	yes	no

COMPARE FEATURES!	SYSRES™	POWER™
Automatic printer output?	yes	no
Selectable ASCII conversion?	yes	no
List programs without loading them?	yes	no
Formatted program listings?	yes	no
Dump SEQuential/RELative files?	yes	no
Edit data files?	yes	no
True program merge?	yes	no
Auto number with **AUTO TEXT?**	yes	no
Load machine language programs?	yes	no
Auto-execute machine language programs?	yes	no
Directory (menu) file commands?	yes	no

COMPARE "EQUIVALENT" FUNCTIONS!

Function: Change occurances of one pattern to another.

Feature	SYSRES™	POWER™
Command word	CHANGE	@
'Wild cards' in search string?	yes	yes
'Wild cards' in replace string?	yes	no
Selectable range?	yes	yes
Match in entire text?	yes	yes
Match in commands only?	yes	no
Match exact variable names?	yes	no

Function: Define special one-key functions.

Feature	SYSRES™	POWER™
Command word	KEY	REM"
Requires BASIC program changes?	no	yes
Destroys variables?	no	yes
Re-define any key?	yes	no
Maximum string length	255	73
Quotes and carriage-return allowed	yes	no
Re-define any token key?	yes	no
Retain user keys from program to program?	yes	no

JUST A FEW OF THE FEATURES OF SYSRES™

* Fast up/down scrolling which works!
* Advanced repeat-key routine!
* Re-define any or all keys as any keyword (full or short form) or as any string up to 255 charactors long!
* Auto line numbering which can feed a string of up to 127 charactors as well!
* Extended DOS support (requires DOS 2A or greater)!
* Never enter another file name! All file commands work from the directory!
* Supports multiple disk drives!
* List BASIC programs, sequential and relative files without loading them into memory!
* TRUE PROGRAM MERGE (overlay). Supports subroutine libraries!
* Load and run machine language programs with parameter passing!
* Supports multiple printers!
* Automatic printer output with paging plus formatted listings with full ASCII code conversion including cursor control and special charactors for non-CBM™ printers!
* Edit text files and assembler source code without leaving BASIC!
* Renumber part of a program or even change the order of lines!
* Over 700 FIND/CHANGE commands including variable names ("A$" will not match "BA$"), pattern matching with "wild-cards", and even commands to remove spaces and REM's!
* Three TRACE modes including trace variables!
* Does not affect BASIC program operation!
* One AUTO-BOOT DISKETTE works for ALL PET™ or CBM™ computers (BASIC 2.0 or greater with at least 16k of RAM.). SYSRES™ requires NO ROM SPACE or extra boards, so you can take it with you if you want to use another computer. It may be put above the screen if you have RAM there. It boots automatically without disturbing any program in RAM!
* If, **for any reason,** you are not satisfied with the SYSRES™ system, you may return it along with any back-up disks (within 30 days) for a **full refund.** Your disks will be erased and returned to you.
* Diskette and Extensive Manual – only $75 (Please specify disk drive model when ordering.)

CALL US FOR THE NAME OF YOUR NEAREST DEALER

We are pleased to announce the aquisition of the author, Don Lekei, and the rights to SYSRES™. Don is now hard at work producing versions of our **STOCKFILE**™ series of integrated **INVENTORY CONTROL, POINT OF SALE, ORDER ENTRY,** and **BILL OF MATERIALS** packages for the PET/CBM™ computers. The best inventory control system for the APPLE][™ will soon be available for the CBM™. See your local dealer for details!

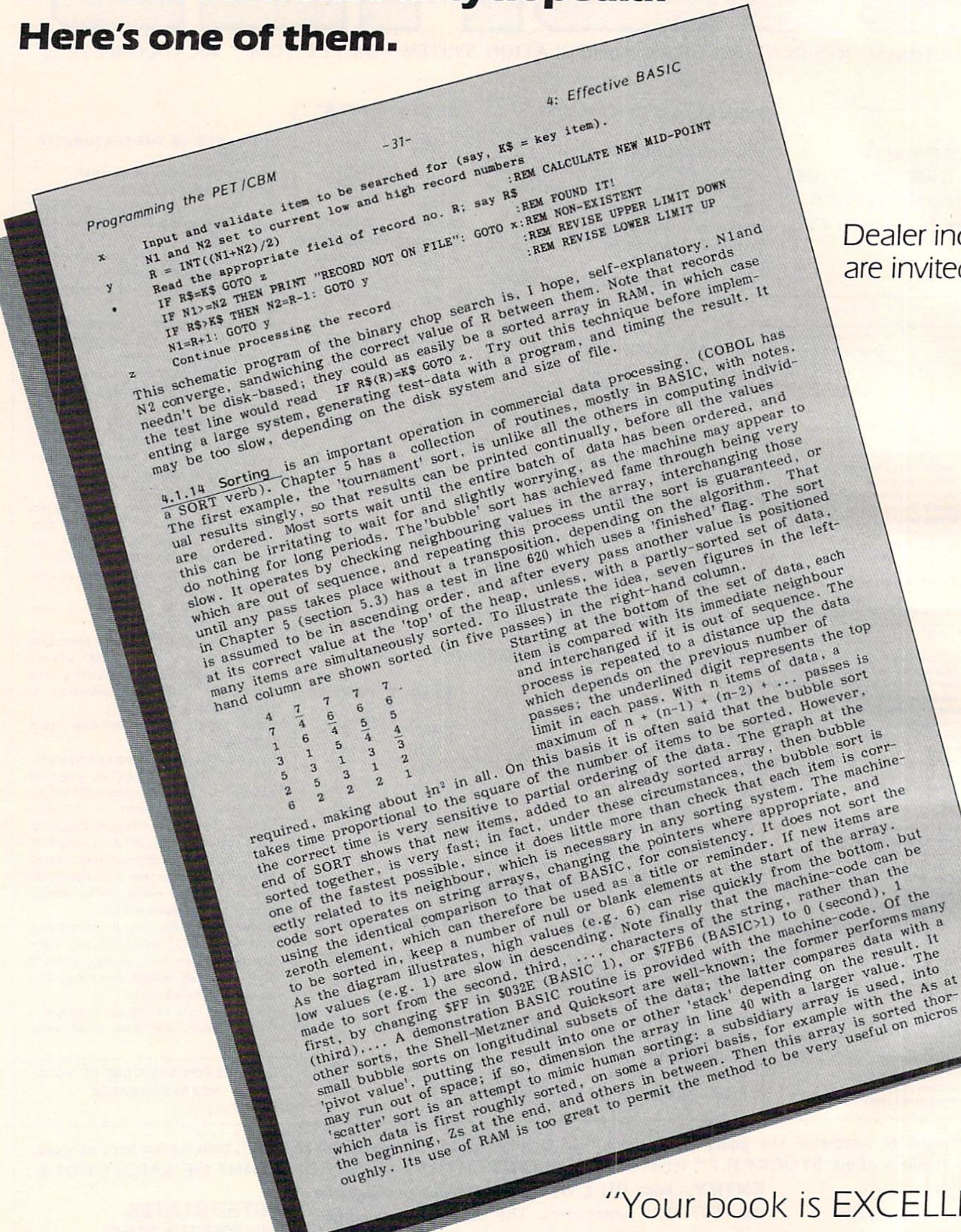
CANADA
Tel: (604) 984-0477
#6, 144 West 15th St.
North Vancouver, B.C.
Canada V7M 1R5

UNITED STATES
Tel: (206) 734-3744
#204, 4202 Guide Meridian
Bellingham, WA
U.S.A. 98226

SOLIDUS INTERNATIONAL CORPORATION

There are over 500 reasons to own this reference encyclopedia.

Here's one of them.



Programming the PET/CBM — -31- — 4: Effective BASIC

```
x   Input and validate item to be searched for (say, K$ = key item).
    N1 and N2 set to current low and high record numbers
y   R = INT((N1+N2)/2)                          :REM CALCULATE NEW MID-POINT
    Read the appropriate field of record no. R; say R$
    IF R$=K$ GOTO z                             :REM FOUND IT!
    IF N1>=N2 THEN PRINT "RECORD NOT ON FILE": GOTO x:REM NON-EXISTENT
•   IF R$>K$ THEN N2=R-1: GOTO y                :REM REVISE UPPER LIMIT DOWN
    N1=R+1: GOTO y                              :REM REVISE LOWER LIMIT UP
z   Continue processing the record
```

This schematic program of the binary chop search is, I hope, self-explanatory. N1and N2 converge, sandwiching the correct value of R between them. Note that records needn't be disk-based; they could as easily be a sorted array in RAM, in which case the test line would read IF R$(R)=K$ GOTO z. Try out this technique before implementing a large system, generating test-data with a program, and timing the result. It may be too slow, depending on the disk system and size of file.

4.1.14 Sorting is an important operation in commercial data processing. (COBOL has a SORT verb). Chapter 5 has a collection of routines, mostly in BASIC, with notes. The first example, the 'tournament' sort, is unlike all the others in computing individual results singly, so that results can be printed continually, before all the values are ordered. Most sorts wait until the entire batch of data has been ordered, and this can be irritating to wait for and slightly worrying, as the machine may appear to do nothing for long periods. The'bubble' sort has achieved fame through being very slow. It operates by checking neighbouring values in the array, interchanging those which are out of sequence, and repeating this process until the sort is guaranteed, or until any pass takes place without a transposition, depending on the algorithm. That in Chapter 5 (section 5.3) has a test in line 620 which uses a 'finished' flag. The sort is assumed to be in ascending order, and after every pass another value is positioned at its correct value at the 'top' of the heap, unless, with a partly-sorted set of data, many items are simultaneously sorted. To illustrate the idea, seven figures in the left-hand column are shown sorted (in five passes) in the right-hand column.

4	7	7	7	7
7	4	6	6	6
1	6	4	5	5
3	1	5	4	4
5	3	1	3	3
2	5	3	1	2
6	2	2	2	1

Starting at the bottom of the set of data, each item is compared with its immediate neighbour and interchanged if it is out of sequence. The process is repeated to a distance up the data which depends on the previous number of passes; the underlined digit represents the top limit in each pass. With n items of data, a maximum of n + (n-1) + (n-2) + ... passes is required, making about $\frac{1}{2}n^2$ in all. On this basis it is often said that the bubble sort takes time proportional to the square of the number of items to be sorted. However, the correct time is very sensitive to partial ordering of the data. The graph at the end of SORT shows that new items, added to an already sorted array, then bubble sorted together, is very fast; in fact, under these circumstances, the bubble sort is one of the fastest possible, since it does little more than check that each item is correctly related to its neighbour, which is necessary in any sorting system. The machine-code sort operates on string arrays, changing the pointers where appropriate, and using the identical comparison to that of BASIC, for consistency. It does not sort the zeroth element, which can therefore be used as a title or reminder. If new items are to be sorted in, keep a number of null or blank elements at the start of the array. As the diagram illustrates, high values (e.g. 6) can rise quickly from the bottom, but low values (e.g. 1) are slow in descending. Note finally that the machine-code can be made to sort from the second, third, ..., characters of the string, rather than the first, by changing $FF in $032E (BASIC 1), or $7FB6 (BASIC>1) to 0 (second), 1 (third),.... A demonstration BASIC routine is provided with the machine-code. Of the other sorts, the Shell-Metzner and Quicksort are well-known; the former performs many small bubble sorts on longitudinal subsets of the data; the latter compares data with a 'pivot value', putting the result into one or other 'stack' depending on the result. It may run out of space; if so, dimension the array in line 40 with a larger value. The 'scatter' sort is an attempt to mimic human sorting: a subsidiary array is used, into which data is first roughly sorted, on some a priori basis, for example with the As at the beginning, Zs at the end, and others in between. Then this array is sorted thoroughly. Its use of RAM is too great to permit the method to be very useful on micros.

Dealer inquiries are invited.

"Your book is EXCELLENT!"

JIM STRASMA,
From a letter to the author.

A **COMPUTE! Books** Publication
In conjunction with Level Limited.

Programs which let motor impaired, nonverbal people communicate with others using a computer, can be of great benefit. Below are versions of a menu-communicator for Apple, PET, and VIC computers.

Micros With The Handicapped

Developing A Communications Program: Part II

Susan Semancik and Marshall Curtis
Delmarva Computer Club

In our last column, we discussed the need for developing a program to help nonverbal, motor-impaired individuals in their communication needs. As the following outline shows, the second part of this series on construction of such a program will identify some of the considerations in the selection of a menu for the program:

I. Introduction (April issue of **COMPUTE!**)
  A. Need for communications program
  B. Outline of series
  C. Reasons for tutorial approach
II. Menu Setup (June issue of **COMPUTE!**)
  A. Type of Communication
    1. Word processing
    2. Computer programming
    3. Daily routine
    4. School classes
  B. Format of Message
    1. Considerations
      a. Output device for message
      b. Multiple input functions
      c. Maximum message length
      d. Screen size of computer display
    2. Calculations
      a. Number of screen lines for message
      b. Number of screen lines for menu
    3. Positioning
      a. Top of screen
      b. Bottom of screen
  C. Format of Menu
    1. Menu entries
      a. Sentences
      b. Words
      c. Characters
    2. Menu arrangement
      a. Row
      b. Column
      c. Block
    3. Menu spacing
      a. Number of blank rows and columns
      b. Number of entries per row or column
      c. Number of rows or columns per screen
  D. Display of Menu
    1. Static
      a. PET computer
      b. VIC computer
      c. Apple II computer
    2. Dynamic

There are many things to consider when developing a communications program. The most fundamental is the purpose for which communication is desired. Rather than develop one program to deal with all types of communication needs, it might be better to have separate programs dedicated to specific objectives – for instance, a program to do word processing functions with printer or modem output; or a program to communicate to a computer in a programming language such as BASIC, PASCAL, or machine language; or a program to communicate with others on a daily routine basis; or a program to use specific vocabularies for different school classes; etc. A list of communication needs compiled on an individual basis among nonverbal, motor-impaired persons would have many similar requirements, regardless of differences in computer equipment available or alternative input devices needed.

We will start with the development of a program to communicate with others on a daily basis, and will develop other types of programs as requests warrant them. Having decided on the type of communication, the next choice to consider is the format for communication: will the user be communicating by selecting from a menu of sentences, words, characters, or a combination of these? Must the physical line length of the computer's display screen be the deciding factor in this choice? Not necessarily. With the use of vertical and/or horizontal scrolling, screen sizes do not have to limit the length of menus or messages.

If enough computer memory or a disk is available, even multiple screens or "pages" of menus could be alternately displayed to increase the number of selections that can be made rapidly. If screen size is limited, a possible alternative might be to have a menu of sentences and/or words permanently displayed on a large poster rather than on the computer's screen. Then the user could access the poster's menu by a row/column or number selection method, with the choice being displayed on a printer, or even verbalized by a

computer-controlled speech synthesizer.

A Versatile And Portable Program

In the series of articles we will be presenting on communications, we will be looking at some of the special features of three different computers and input devices we have available. This will mean that we will want the program we develop to be as versatile and portable as possible, and will try not to use features peculiar to a particular system if a more standard approach can be found. This will help those readers who want to develop a similar communications program for different systems. The three computers we will use have the following screen characteristics: 1) Commodore PET computer with 25 lines of 40 characters each; 2) Commodore VIC computer with 23 lines of 22 characters each; and 3) Apple II computer with 24 lines of 40 characters each.

Since this program will involve communicating with others on a daily basis, each user must make up an individual list of words involving his/her own routines. Start with a list of frequently used names of people, places, and articles, and of activities, actions, or descriptions most often associated with these words. Also, consider word endings if grammatical structure will be important in your messages, as well as punctuation, numbers, and special symbols. The alphabet should also be included so words can be formed that are not accessible from the menu.

Individualizing the menu will speed up the communication process. Not only can a message be formed faster with the computer's help, but the formation will not require the active involvement of the person with whom you want to communicate. The computer would be used in this case as a recorder of the message, so that there will be continuity of expression, and the message can be repeated without having to be reformed. Using the sound capability of most home computers, an audible signal could also be given when a message is ready for viewing.

For this demonstration program, we will choose a menu format that will include words and characters. If you want the entire menu to be visible at all times, then the computer's display screen characteristics will determine the size and structure of the menu. This type of menu will be called a static menu and will be shown first, since it is the easiest to create in a general fashion for any type of home computer. Later, we will investigate several types of dynamic menus, such as multiple menus and scrolling menus.

Next, decide on whether the message you form will be sent to an external device, and/or displayed on the computer's screen. If it will only be sent to an external device, such as a printer, then the entire screen can be used for the menu. If it will also be sent to the screen, then consider the maximum length necessary for your messages. To calculate the number of screen lines needed for the message area in a static menu, divide the maximum message length by the number of characters contained in the computer's screen width. Round up if any remainder exists, and/or add one to the result if you don't want words split at the right edge of the screen.

Also, decide if you will need several input functions and, if so, will they be performed by an alternative input device, or menu-driven from the screen. Input functions will be the topic of a later column, but they can include such things as erasing your last entry from the message, changing the selection speed for input, changing menus, changing output devices, or alternating between areas of the screen. If these functions are selected from the screen, then your communication's menu area will be smaller for a static menu.

We will assume in this demonstration that 160 characters will be the maximum message length to be displayed on the computer's screen. Also, we will reserve two lines of the screen for input functions, and will include an extra line in the message area to take care of words that would otherwise be split at the right edge of the screen. This means that our three computers would require 5, 9, and 5 lines, respectively, of the screen for the message area, and would have 18, 12, and 17 lines, respectively, remaining for the menu area.

Example 1: Commodore VIC Computer

160/22 =	**8**	**lines for message**
	+1	**for no edge-splitting of words**
	9	**lines for message area**
	+2	**for input functions**
23 - 11 =	**12**	**lines for menu**

The menu can be arranged in rows, columns, blocks, etc. Remember that the main consideration in the arrangement is how easy it will be to indicate and retrieve a selection. The row or column arrangements give sufficient structure to satisfy the selection consideration. For this demonstration, we will use a column arrangement, with an attempt made to set the words in columns simulating sentence structure, and with characters appearing in an end column.

Spacing between adjacent columns and rows will be a factor of the distance the user will be from the computer's screen, in conjunction with the character size displayable by the computer. If vision is also a problem for the user, can the computer's characters be displayed larger? Yes, but since that's a concern shared by many others who may not need a communications program, we will address that problem as a separate issue in a later article.

For our demonstration program, we will assume one blank row between column entries, and one space between adjacent columns. The number of entries per column will be the quotient when you divide the number of screen lines for the menu by one more than the number of blank rows between column entries. This means that the PET, VIC, and Apple computers can have 9, 6, and 8 column entries, respectively, with the decisions made so far.

Example 2: Apple II Computer

17/2 = 8 entries per column

Since the width of each column will be determined by the largest word within it, group your list of words into sentence structure arrangement, and determine the number of columns that will form a static menu for your screen width. Sample menus follow for each of the three computers:

Example 3: Menu for the PET Computer:

DOCTOR	I	IS	HAVE	COME	SEE	INGEDS
TEACHER	YOU	ARE	HAS	BATH	EAT	AOTFR
WILL	WE	GO	GOOD	DRINK	AND	.ULHCP

Example 4: Menu for the VIC Computer:

DR.	IS	COLD	INGEDS12
I	AM	WHEN	AOTFR34
YOU	ARE	DRINK	.ULHCP56
MOM	EAT	WANT	?MYWKB78
DAD	NO	TIME	,VJQZX90
HOT	YES	SLEEP	;$%()'+-

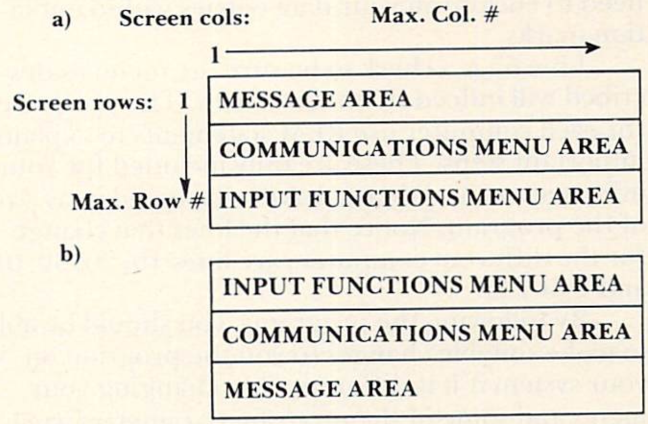
Example 5: Menu for the Apple Computer:

DOCTOR	I	IS	HAVE	COME	SEE	INGEDS
TEACHER	YOU	ARE	HAS	BATH	EAT	AOTFR
WILL	WE	GO	GOOD	DRINK	AND	.ULHCP

Once you've given the menu sufficient thought, then one of the last decisions to be made is the placement of the communications menu and the input functions menu, either at the top, bottom, or middle of the screen. And, if the message is also to be placed on the screen, where will it go? For our demonstration program, we will assume the communications menu will be in the middle of the screen, with the message area being either at the top or bottom of the screen. We are also assuming that the screen columns are numbered from left to right starting from one, and the screen rows are numbered from top to bottom starting from one.

We are finally at a point where we can start to consider the programming aspect of the project. It may seem that it took us a long time to get to this point, but if not enough attention is paid to the choosing and positioning of the elements of the menu, then no amount of programming tricks will make this be a functional tool for the motor impaired, nonverbal user.

Example 6: Display screen placement.



A Simple BASIC Program

We will start with a simple BASIC program that will enable you to view your individual menus on your computer's display screen. Programs 1 through 3 contain programs for our three computers to display the menus of Examples 3-5 by rows. The following is a list of variables that will be used in those programs:

W = the number of characters per screen width

RM = the number of rows in the communications menu

BR = the number of blank rows between column entries

CM = the number of columns in the communications menu

BC = the number of spaces between adjacent columns

RI = the number of rows in the input functions menu

SR = the starting screen row for the communications menu

SC = the starting screen position for the 1st column

S( ) = the starting column positions for each column

L( ) = the width of each column

P = the tabbing position for a column

Note that line 75 of the program is written for a TAB function which starts count at zero. If your computer's TAB function starts count at one, replace line 75 with P=S(C)+TP. In line 95, TP is used to adjust the TAB value for computers which can TAB beyond the screen width. For those that don't, this line can be adjusted or deleted. If your system doesn't allow variable dimensions as in statement 25, then set the DIM for each variable to the value of W. Also, on other computers you may need to enclose all your data entries within quotation marks.

Line 65 is a check to be sure the menu as described will indeed fit on the screen. The programs for each computer use REM statements to explain important steps. These are only included for your information, and do not have to be typed in as part of the program. Notice that the lines that change for the different computers are lines 10, 20, 30, 95, and 140-195.

By following the programs, you should be able to make suitable changes to run the program on your system if it is different. Try changing your menu and some of the program parameters, such as number of columns, number of blank rows between column entries, starting row or column position on the screen. You should get a feel for how flexible this program can be.

In the next issue we'll cover the selection process in the program. Will selected entries be retrieved from DATA lists, subscripted variables, memory blocks, or from the screen itself? We will also need to look ahead at the possibility of a dynamic menu and how multiple menus can be entered without disturbing the flow of the program. Let us know if you have any special ideas you'd like developed in this program, and we'll try to incorporate them as we go along.

Delmarva Computer Club
P.O. Box 36
Wallops Island, VA 23337

Program 1.

```
5 REM EXAMPLE 8A) PET COMPUTER
10 PRINT CHR$(147);:REM CLEAR TEXT SCREEN
20 W=40:RM=9:BR=1:CM=7:BC=1:RI=2:SR=3:SC=1
   :REM SET MENU PARAMETERS
25 DIM S(CM),L(CM):S(1)=SC
30 DATA 7,4,4,4,5,3,6:REM COLUMN WIDTHS
35 IF CM=1 THEN 50
38 REM CALCULATE STARTING POSITION OF EACH
    COLUMN
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 POSITI
   ON 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
   $
75 P=S(C)-1+TP
80 PRINT TAB(P);M$;:NEXT C
90 IF S(CM)+LEN(M$)-1<W THEN PRINT:GOTO 10
   0:REM WRAPAROUND ADVANCES A LINE
95 IF BR=0 THEN TP=TP+W:IF TP>87 THEN TP=0
   :REM 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 BLA
   NK ROWS BETWN COLUMN ENTRIES
120 NEXT R
130 GOTO 130:REM DISPLAY ISN'T DISTURBED UN
   TIL USER BREAKS PROGRAM
139 REM ENTER DATA BY ROWS
140 DATA DOCTOR,I,IS,HAVE,COME,SEE,INGEDS
145 DATA TEACHER,YOU,ARE,HAS,BATH,EAT," AOT
   FR"
150 DATA WILL,WE,GO,GOOD,DRINK,AND,.ULHCP
155 DATA HOW,DO,CAN,BAD,SLEEP,IN,?MYWKB
160 DATA WHO,GET,AM,DID,BED,OUT,",VJQZX"
165 DATA WHAT,MOM,WANT,HOT,RADIO,TV,";$%()'
   "
170 DATA WHERE,DAD,TO,COLD,ROOM,YES,"!*/^=:
   "
175 DATA WHEN,JOHN,TIME,THE,FOOD,NO,-01234
180 DATA WHY,RICK,DAY,CALL,PLAY,AT,+56789
200 PRINT "MENU SIZE ERROR!":END
```

Program 2.

```
5 REM EXAMPLE 8B) VIC COMPUTER
10 PRINT CHR$(147);:REM CLEAR TEXT SCREEN
20 W=22:RM=6:BR=1:CM=4:BC=1:RI=2:SR=3:SC=1
   :REM SET MENU PARAMETERS
25 DIM S(CM),L(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
    COLUMN
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 POSITI
   ON 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
   $
75 P=S(C)-1+TP
80 PRINT TAB(P);M$;:NEXT C
90 IF S(CM)+LEN(M$)-1<W THEN PRINT:GOTO 10
   0:REM WRAPAROUND ADVANCES A LINE
95 IF BR=0 THEN TP=TP+W:IF TP>87 THEN TP=0
   :REM 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 BLA
   NK ROWS BETWN COLUMN ENTRIES
120 NEXT R
130 GOTO 130:REM DISPLAY ISN'T DISTURBED UN
   TIL USER BREAKS PROGRAM
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
```

Program 3.

```
5 REM EXAMPLE 8C) APPLE II COMPUTER
10 TEXT : HOME : REM CLEAR TEXT SCREEN
20 W=40:RM=8:BR=1:CM=7:BC=1:RI=2:SR=3:SC=1
   :REM SET MENU PARAMETERS
25 DIM S(CM),L(CM):S(1)=SC
30 DATA 7,4,4,4,5,3,6:REM COLUMN WIDTHS
35 IF CM=1 THEN 50
38 REM CALCULATE STARTING POSITION OF EACH
    COLUMN
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 POSITI
   ON 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
   $
75 P=S(C)-1+TP
80 PRINT TAB(P);M$;:NEXT C
90 IF S(CM)+LEN(M$)-1<W THEN PRINT:GOTO 10
   0:REM WRAPAROUND ADVANCES A LINE
95 IF BR=0 THEN TP=TP+W:IF TP>87 THEN TP=0
   :REM 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 BLA
    NK ROWS BETWEEN COLUMN ENTRIES
120 NEXT R
130 GOTO 130:REM DISPLAY ISN'T DISTURBED UN
    TIL USER BREAKS PROGRAM
139 REM ENTER DATA BY ROWS
140 DATA DOCTOR, I,IS,HAVE,COME,SEE,INGEDS
145 DATA TEACHER,YOU,ARE,HAS,BATH,EAT," AOT
    FR"
150 DATA WILL,WE,GO,GOOD,DRINK,AND,.ULHCP
155 DATA HOW,DO,CAN,BAD,SLEEP,IN,?MYWKB
160 DATA WHO,GET,AM,DID,BED,OUT,",VJQZX"
165 DATA WHAT,MOM,WANT,HOT,RADIO,TV,";$%()'
    "
170 DATA WHERE,DAD,TO,COLD,ROOM,YES,"!*/^=:
    "
175 DATA WHEN,HELP,TIME,CALL,FOOD,NO,+56789
200 PRINT "MENU SIZE ERROR!":END
```

©

Learning With Computers

Mary M. Humphrey
Teaching Tools: Microcomputer Services

Summer Computing

This fall many teachers will be reading some new answers to an old assignment: "How I Spent My Summer Vacation." They may have already read reports of summer camps with bugs, but this time the bugs are in programs and the correct spelling of bite is b-y-t-e.

A new type of specialty camp, the computer camp, is gaining popularity. Five years ago, the first camp to include computer instruction was started in Connecticut. It was begun with the idea that computing is fun, and children need opportunities to learn about computing as an educational and creative recreation. The idea has spread, and this year there are several computer camps from Cape Cod to San Diego. We can expect their numbers to continue to grow as more children begin using computers in school and at home.

Recently a friend asked his 13 year-old son if he wanted to go to a computer camp this summer. Daniel, who had been working with computers at school and at home, was very excited about the idea. His first question was, "What sports and stuff do they have?" He then wanted to know, "Are there going to be any kids my age?" and, "Who's going to be running it?" It wasn't until the next day that Daniel asked his father, "What computers are they going to have, and what are they going to teach us?" After thinking over the answers for awhile, he asked one last question, "Suppose I don't go to camp this summer, could I have the money to buy my own computer instead?"

The last question was a particularly good one. For kids like Daniel who have had a lot of computing experience, are confident and can get the help they need at school or at home, having their own machine might be a better investment than going to camp. For kids who are just beginning to learn about computers, a computer camp can provide the instruction they need to make good use of their own machine. For kids who are interested in projects beyond their teachers' or parents' skills, a computer camp can be an enjoyable way to get special help. The decision of whether to go to a camp, and to which one, should depend on matching a child's needs and interests to what a camp has to offer.

What To Look For In A Computer Camp

When I began contacting computer camps for information (see Table), I found Daniel's questions quite useful. They cover several important similarities and differences among computer camps:

1) *Camping Activities.* Most computer camps offer a variety of "sports and stuff." Hiking, swimming and field sports are available at all of the camps I contacted, and some also provide tennis, riding, boating, field trips, and indoor sports. Crafts, campfires, and other traditional camping activities are included at all of the overnight camps.

The amount of activities and the time scheduled for them is related to the length of the camping session. Most four week or longer sessions allow a large part of the day for non-computing activities, while one or two week sessions usually provide more intensive computing experiences.

2) *Other Campers.* Some of the camps I contacted accept girls and boys as young as 7 years old, but most camps specify a 10 to 18 year-old age range. All of the camps welcome beginners. The director of a camp starting its third season said that, while as many as half of his campers have no previous computing experience, he also has a number of return students who receive advanced instruction.

In addition to their student instruction, one camp offers a special six week session for teachers on how to teach about computers and programming. During the last three weeks of camp, each teacher works with two of the student campers.

3) *Camp Staff.* Staff can be described as either computing or camping staff, "indoor and outdoor people" as one supervisor termed them. The camping staffs are generally people with experience in leading sports, crafts and group activities. Some camps are managed by professional camping organizations.

The computing staffs vary widely among camps. One camp limits its computer instructors to teachers who have had experience with teaching children about computers. Another camp employs university faculty and students from education and computer science departments. They also bring in special guest speakers from the computer industry. Such staff differences may reflect differing proportions of young beginners and older advanced students at each camp.

4) *Computing Instruction.* All the camps provide instruction in the roles of computers in everyday life and in how computers work. Beginning BASIC programming is also included. Camps with longer sessions usually teach additional topics such as

ARTWORX. IT'S A WHOLE NEW WORLD OF SOFTWARE.

Scene from BETA FIGHTER during creation using the DRAWPIC graphics editor.

HODGE PODGE: by Marsha Meredith (Atari and Apple)
NOW AVAILABLE FOR ATARI!!! This captivating program is a marvelous learning device for children from 18 months to 6 years. HODGE PODGE consists of many cartoons, animation and songs which appear when any key on the computer is depressed. A must for any family containing young children.
PRICE .... $19.95 diskette

□ **PM EDITOR:** by Dennis Zander (Atari, 16K)
Create your own fast action graphics game for the Atari 400 or 800 using its player missile graphics features. By using player data stored as strings, players can be moved or changed (for animation) at machine language speed. All this is done with string variables (PO$(Y)=SHIP4) This program is designed to permit creation of up to 4 players on the screen, store them as string data and then immediately try them out in the demo game included in the program. Instructions for use in your own game are included. PM EDITOR was used to create the animated characters in **ARTWORX** RINGS OF THE EMPIRE and ENCOUNTER AT QUESTAR IV.
PRICE .... **$29.95** cassette **$33.95** diskette

□ **ROCKET RAIDERS** by Richard Petersen (Atari 24K)
Defend your asteroid base against pulsar bombs, rockets, lasers, and the dreaded "stealth saucer" as aliens attempt to penetrate your protective force field. Precise target sighting allows you to fire at the enemy using magnetic impulse missiles to help protect your colony and its vital structures.
PRICE .... **$19.95** cassette **$23.95** diskette

□ **FOREST FIRE TWO:** by Richard Petersen (Atari 24K)
FOREST FIRE has been enhanced and now offers a two player mode for head to head competition to see who can survive, suffer the least damage and put their fire out first. User input now determines landscape, wind and weather conditions, offering limitless game variation. FOREST FIRE's excellent color graphics have been made even better, turning your computer into a super-detailed fire scanner.
PRICE .... $16.95 cassette $20.95 diskette

□ **GIANT SLALOM:** by Dennis Zander (Atari, 16K)
Bring the Winter Olympics to your computer anytime of the year! Use the joystick to guide your skier's path down a giant slalom course consisting of open and closed gates. Choose from three levels of difficulty. Take practice runs or compete against from two to eight additional skiers.
PRICE .... **$15.95** cassette **$19.95** diskette

□ **THE PREDICTOR** by Thomas Barker
(Apple, Atari, TRS-80, North Star and CP/M (M-BASIC)
This is a complete package that covers least squares fitting of parameters for two or more variables. THE PREDICTOR can be used for predicting sales and process behavior, trend analysis, model building and many other uses calling for multilinear regression techniques. Each option in the program is prompted with simple YES/NO commands making it **very** easy to use.
PRICE .... **$29.95** diskette

□ **PILOT:** by Michael Piro (Atari, 16K)
Pilot your small airplane to a successful landing using both joysticks to control throttle and attack angle. PILOT produces a true perspective rendition of the runway, which is constantly changing. Select from two levels of pilot proficiency.
PRICE .... **$16.95** cassette **$20.95** diskette

□ **TEACHER'S PET:** by Arthur Walsh (Atari, Apple, TRS-80, PET, North Star and CP/M (MBASIC) systems).
This is an introduction to computers as well as a learning tool for the young computerist (ages 3-7). The program provides counting practice, letter-word recognition and three levels of math skills.
PRICE .... **$14.95** cassette **$18.95** diskette

□ **MAIL LIST 3.0:** (Atari, Apple and North Star)
The very popular MAIL LIST 2.2 has now been upgraded. Version 3.0 offers enhanced editing capabilities to complement the many other features which have made this program so popular. MAIL LIST is unique in its ability to store a maximum number of addresses on one diskette (typically between 1200 and 2500 names!). Entries can be retrieved by name, keyword(s) or by zip codes. They can be written to a printer or to another file for complete file management. The program produces 1, 2 or 3-up address labels and will sort by zip code (5 or 9 digits) or alphabetically (by last name). Files are easily merged and MAIL LIST will even find and delete duplicate entries! The address files created with MAIL LIST are completely compatible with **ARTWORX** FORM LETTER SYSTEM.
PRICE .... **$49.95** diskette

□ **THE VAULTS OF ZURICH:** by Felix and Greg Herlihy (Atari, 24K, PET)
Zurich is the banking capital of the world. The rich and powerful deposit their wealth in its famed impregnable vaults. But you, as a master thief, have dared to undertake the boldest heist of the century. You will journey down a maze of corridors and vaults, eluding the most sophisticated security system in the world. Your goal is to reach the Chairman's Chamber to steal the most treasured possession of all: THE OPEC OIL DEEDS!
PRICE .... **$21.95** cassette **$25.95** diskette

□ **BRIDGE 2.0** by Arthur Walsh (Atari (24K), Apple TRS-80, PET, North Star and CP/M (MBASIC) systems)
Rated #1 by Creative Computing, BRIDGE 2.0 is the only program that allows you to both bid for the contract and play out the hand (on defense or offense!). Interesting hands may be replayed using the "duplicate" bridge feature. This is certainly an ideal way to finally learn to play bridge or to get into a game when no other (human) players are available.
PRICE .... **$17.95** cassette **$21.95** diskette

□ **ENCOUNTER AT QUESTAR IV:** by Douglas McFarland (Atari, 24K)
As helmsman of Rikar starship, you must defend Questar Sector IV from the dreaded Zentarians. Using your plasma beam, hyperspace engines and wits to avoid Zentarian mines and death phasers, you struggle to stay alive. This BASIC/Assembly level program has super sound, full player missile graphics and real time action.
PRICE .... **$21.95** cassette **$25.95** diskette

NEW PROGRAMS!

HAZARD RUN: by Dennis Zander (Atari, 16K)
The sheriff has spotted you and you must make the treacherous run through Crooked Canyon past Bryan's Pond to the jump at Hazard Creek and safety. You can even put the joystick-controlled GEE LEE car up on two wheels to make it through some tight spots. A lead foot is not always the answer as you dodge trees, rocks and chickens in this nerve-racking game. HAZARD RUN employs full use of player/missile graphics, re-defined characters and fine scrolling techniques to provide loads of fast action and visual excitement.
PRICE .... **$27.95** cassette **$31.95** diskette

BETA FIGHTER: by Douglas McFarland (Atari, 16K)
See who will be the ace gunner in this action game set on a spectacular Martian landscape. BETA FIGHTER can be played with one or two players and uses player/missile graphics and delightful sound effects.
PRICE .... **$16.95** cassette **$20.95** diskette

DRAWPIC: by Dennis Zander (Atari 16K)
DRAWPIC provides the user with an unbelievably easy way to create screens in graphics modes 3-7. Just sit back with your joystick and use POINT PLOT, DRAW LINE, RUBBER BAND fill and COLOR SET to create beautiful images on your Atari. Full or partial screen images are saved as string data in the program and can be instantly recalled and combined into new images using machine language subroutines. These graphic images can be easily incorporated into your own programs. The images of HODGE PODGE and the landscape of BETA FIGHTER were made using DRAWPIC.
PRICE .... **$29.95** cassette **$33.95** diskette

T: A TEXT DISPLAY DEVICE: by Joseph Wrobel (Atari 16K)
T: is an auto-loading, co-resident assembly language routine which greatly expands the display capabilities of the Atari. It allows you to freely intermix both text and graphics without the use of modified display lists, PEEKS or POKES. This is done by defining a new device ("T:"); printing to that device puts text onto the screen. The size of the text is determined by the graphics mode used.
PRICE .... **$17.95** cassette **$17.95** diskette

ORDERING INFORMATION

Call ARTWORX toll-free number to order direct:
800-828-6573
In New York, Alaska, Hawaii call:
(716) 425-2833

All orders are processed and shipped within 48 hours.

Shipping and handling charges:
Within North America: Add $2.00
Outside North America: Add 10% (Air Mail)
New York State residents add 7% sales tax

Quantity Discounts:
Deduct 10% when ordering 3 or more programs

Ask for ARTWORX at your local computer store.

Write for FREE Catalogue listing more information about these and other quality ARTWORX programs.

Artworx

150 North Main Street Fairport, NY 14450

COMPUTER CAMPS

Camp Organization	Camp Location(s)	Sessions / Prices	Campers	Computer Instruction	Computer Resources
Atari Computer Camp 40 East 34th St. New York, NY 10016 (800)847-4180	U.C. San Diego Asheville, NC Sheboygan, WI E. Stroudsburg, PA	4 weeks/ $1590 8 weeks/ $2790 ($100 deposit)	10-18 years old	2 hours instruction and all day free time "Designing our own computer curriculum"	Atari 400 & 800 2 kids: 1 computer
Computer Camp, Inc. Suite G 1235 Coast Valley Santa Barbara, CA 93108 (800)235-6965	Santa Barbara, CA Tahoe Pines, CA Cape Cod, MA	2 weeks/ $795	7-16 (80 per session)	3 hours instruction 3 hour's free time "BASIC, PASCAL, FORTRAN, assembler, electronics lab, AI, robotics"	Apple, Atari Commodore, Texas Instruments 2:1
Computers for Kids, Inc. 8 Benton Ct. #4 Tiburon, CA 94920 (415)435-1310	Tiburon, CA	1 & 2 weeks approx. $4-500. per week	10-18 (30-35 per session)	2-3 hours instruction "computer literacy. BASIC, special projects"	Apple, IBM personal computer 25 computers
Computer Tutors at Stanford School of Ed. Stanford University Stanford, CA 94306 (415)497-2119	Stanford University	5 weeks (days) $1000 (extra fee for resident camp)	12-15 (66 per session)	3 hours instruction afternoon free time "BASIC, assembler, PASCAL, graphics, video disks, word processing, robotics"	IBM personal computers 2:1
Data Base Computer Camp 6454 Valley View Rd. Oakland, CA 94611 (415)339-2961	Placerville, CA	10 days/ $425 ($150 deposit)	7-16 (50 per session)	3 hours instruction and free time "computer literacy, BASIC, special projects"	Apple, Commodore 2:1
National Computer Camp Box 624 Orange, CT 06477 (203)795-3049	Orange, CT Atlanta, GA	1,2,& 4 weeks $345 per week	10-18 (120 per session)	2-3 hours instruction all day free time "BASIC, machine & assembler, novice to advanced instruction"	TRS-80, Apple, Commodore, Wang 50 computers
Timberline Tech Computer Camp 1287 Lawrence Sta. Rd. Sunnyvale, CA 94086 (408)745-1110	Sunnyvale, CA	2 weeks/ approx. $895	10-17 (60 per session)	2-3 hours instruction and free time "beginning to advanced instruction"	Apple 2:1
Computer Camps International 310 Hartford Turnpike Vernon, CT 06066 (203)871-9227	East Haddam, CT Denton, TX Whitewater, WI ("many East coast sites")	2 weeks/ $290 (1/2 day sessions) /$795	5-17(day) 9-17(res) (100 per session)	4 hours instruction and free time "any level-BASIC, LOGO, other languages, robotics, special projects – computer morality class"	Apple, Texas Instruments 100 computers 1:1
CompuCamp 7101 York Ave. South Edina, MN 55435 (612)835-0064	St. Paul, MN Beaver Falls, PA	1 & 2 weeks $175/wk-day camp $390/week residence camp	8-17 (50-75 per session)	5 hours instruction and free time "BASIC, LOGO, PASCAL, PLATO* Tutor, beginning to advanced instruction	Apple, Atari, Texas Instruments, TRS-80, PLATO Tutor System* 2:1
Lake Forest Computer Camp Sheridan & College Roads Lake Forest, IL 60045 (312)234-3100	Lake Forest College, IL	1 week/ $350	11-17 (70 per session)	6 hours instruction 3 hours free time "beginning and intermediate BASIC"	Apple 2:1

*PLATO: Control Data Corporation microcomputer specially developed for PLATO software.

other computer languages, use of peripheral devices like printers and video discs, graphics, word processing, robotics and special projects. Some camps also provide an electronics lab as one type of "crafts" class.

Many camps have more than one type of microcomputer system available. Campers usually work two to a machine – the buddy system appears to be a good idea for computing as well as other camp activities. Generally, there is one instructor for every three to five campers. Classes last two to three hours each day. All of the camps provide free time for extra computer practice. Some camps do restrict game playing to the lowest priority for computer use, or to using only games made by the campers themselves.

The Table gives a sampling of the computer camps available this summer. It includes addresses and phone numbers to contact for more information.

What If You're Not Going To Camp?

The summer months mean kids are out of school, families take vacations, and there is generally more time for parents and kids to do things together. Home computing is one activity you can enjoy as a family recreation.

The most obvious choices for recreational home computing are game programs. These should be chosen with some consideration to who will be playing them, as well as what the game does. Games that require quick reflexes and good visual-motor coordination provide exciting competition for players of about the same skill level. However, when there is a great difference in ability, some parents may be disappointed to learn that they are no competition for their children. Competitive games that stress strategies or knowledge of particular facts are also best used by players of about the same level. While parents can use this type of game to teach their children, it may not be very interesting for the parent, and more like homework than a game for the child.

Adventure games are one type of computing activity that works well with parents and children. The game presents a challenging task, such as finding a treasure or getting to a secret place before a certain time limit. Players are given messages and hints to help find their way, avoid dangers and enemies, and gather extra treasures or points. The adventures are designed to allow players many options and to respond differently to each choice. Since players work together to accomplish the task, rather than working against each other, different ability levels don't lead to lopsided scores. The details of each game change as players make different choices of what to do and where to go. This kind of variety within a set of rules means players get better with practice, but not bored. Adventure games encourage a lot of discussion and decisions among players and are a good choice for family computing.

Creating your own household software can be an effective and enjoyable way to learn programming skills and coordinate computing with other family activities. Parents and kids can work together on writing small programs to compute the gas mileage of the family car, keep track of vacation expenses, figure batting averages for a whole little league team, print out price labels for a garage sale, etc... Maybe you can use VisiCalc or other commercial programs to do all of these tasks, but then you wouldn't have the fun and learning that come from working together on a very personalized project.

Programs for composing and playing music and for creating graphic art displays share features with the activities recommended above: they are fun and easy to use, allow people with different skill levels to work together, encourage interaction between the users, and have enough variety to be used many times. With a little imagination, parents and kids can come up with many more ideas for fun home computing.

Next month's column will cover another topic of interest to kids, parents, and teachers who learn with computers – LOGO, a computer language designed for children and computer novices. Three versions of the LOGO language, Apple LOGO by LOGO Computer Systems, Inc., the Terrapin LOGO Language by Terrapin, Inc., and M.I.T. LOGO by Krell Software Corp. will be reviewed. ©

Review:

VIC-20 Cartridge Games

Harvey B. Herman
Associate Editor

The cartridge games described here have some common features. They all plug in to the back of the VIC and start automatically on power up. Each has impressive graphics, color, and sound as part of the program. They begin with a demonstration of the program features. Provision is made to center the program on the screen if your TV is misaligned. Many of the games can be played with either the keyboard or a joystick. When a score is shown, the previous high is also given, so subsequent players have something to shoot for.

One cautionary note: do not insert or remove the cartridge with power on. The instructions make this point and I heartily endorse it. Perhaps it would be wise to assign the task to a careful adult or older child. Now on to the games.

Jupiter Lander

Armchair astronauts can now have a real time simulation of a space ship landing on Jupiter. Three landing sites are available – two of them are quite difficult at first. Your object is to score the most points before your fuel runs out. When you land softly, bonus points are awarded and your fuel supply is boosted. More difficult sites, of course, have a higher bonus. The rate of descent (or ascent) is continuously monitored by a gauge on the right side of the screen. The landing is A-OK if the gauge needle is centered on touchdown.

My kids rated the game, initially, 8 out of 10. However, they seem to play it more than the others so I suspect the real rating is higher. An old fogey, like me, enjoyed it, but found it almost impossible to land on the more difficult sites. The kids found it challenging but learned how to do it almost every time. C'est la vie.

Super Alien

You are in an irregular grid being chased by unfriendly aliens trying desperately to stay alive. When an alien catches you, he eats you – not a pretty sight. The object is to score points by trapping them in air bubbles which you have laid down in strategic spots. When an alien is trapped by a fully inflated air bubble, you have a short time to reach it (using the keyboard or joystick) and deflate it off the board. Otherwise, the alien escapes, destroys the bubble and continues the chase. Points are scored when an alien is removed from the board. The number of points is determined by removal time, i.e., faster players will receive a higher score. Extra aliens are added when all the aliens are removed and the game begins anew. Your turn is over when you have been eaten three times. The aliens are normally relatively slow moving and not terribly smart. However, if the game goes on too long, they suddenly become fast and aggressive and it is impossible to escape their greedy mouths.

I enjoyed playing this game more than my kids; they rated it 6 out of 10. Perhaps they were influenced by the fact that our joystick was not working properly (the connector was loose) and they were forced to use the keyboard. Also the grid was oversized for our TV and you could not see the score and the whole grid at the same time.

VIC Avenger

This program is modeled after the popular arcade game, Space Invaders. You are in control of a base (space ship) at the bottom of the screen which can move left or right while firing at enemy aliens and dodging their bombs. The aliens are arranged in rows which move back and forth relentlessly closing in with time. As with many of the other games, you are given a choice of keyboard or joystick control. Points are scored when you destroy an alien. At unpredictable times a mystery ship moves across the screen. Each class of alien is assigned a different point value. The score for the mystery ship is random. Your base can protect itself by hiding behind solid objects but these are continuously being annihilated by enemy bombs. If the aliens drop too low you cannot avoid being destroyed.

At the start you are given three bases (turns). When you reach 1500 points an extra base is awarded. The game is over when you have lost all your bases. My kids rated it 9 out of 10 and it has proven to be the second most popular game. They have become quite proficient at it and put my feeble attempts to shame.

Draw Poker

VIC deals you a poker hand and your payoff is determined by the odds. Less than a pair of jacks is worthless but a big hand like a royal flush pays 500 to 1. You are allowed to bet up to 9 coins. As you bet, odds are displayed so you can see exactly how much each poker hand is worth. The cards have their backs to you (Commodore Japan logo) and are exposed dramatically one at a time. After the deal and before the draw you select the cards you want to keep (0 to 5). If you wind up with a winning hand the bank will offer to cut cards with you, double or nothing. Resist the temptation on big winning hands.

My kids rated it 6 out of 10 as they don't seem to care much for poker simulations. As for me, I prefer losing to friends.

Super Slot

This program is a realistic computer simulation of a slot machine. I believe you lose as fast as you do on the real ones in Las Vegas or Atlantic City. You have the option of playing with the keyboard or a joystick. First you place a bet of up to 5 coins from your initial stake of 80. Betting extra coins allows more winning combinations. If you bet the maximum, you can win five different ways. The possible winning combinations are shown initially and can be recalled at will. When you're ready to play, a smiling man is shown pulling a lever. He keeps smiling even when you lose. The wheels spin – cherries, lemons, plums and other assorted goodies flash in front of you. The three wheels stop one at a time with lots of dramatic music. If a win shows, your money supply goes up. In any case it's time to bet again. As in real life, the game is over when your money is gone.

The kids and I were impressed with the remarkable graphic symbols. They rated it 9 out of 10. I am not sure how long it can keep an adult who isn't an inveterate gambler amused. Nevertheless, the program should be seen just to appreciate the capabilities of the VIC.

Commodore Business Machines, Inc.
950 Rittenhouse Rd.
Norristown, PA 19403
$29.95 each

©

We'll have a review of two implementations of FORTH for PET and one for Apple and other Atari FORTHs in upcoming issues. For further information about this increasingly popular language, see The Forth Page elsewhere in this issue.

Review:

QS FORTH For Atari

Charles Brannon
Editorial Assistant

FORTH is now available for almost any computer, thanks to the distribution of public domain source code for the language by the FORTH Interest Group (FIG). Fig-FORTH is the group's implementation standard of FORTH for microcomputers.

QS FORTH was the first Atari implementation of fig-FORTH. It comes on a floppy disk with a padded vinyl manual. The manual is extensive and well-written, but does assume a basic familiarity with FORTH. An excellent resource for learning to program in FORTH is *Starting FORTH* by Leo Brodie (available from the FORTH Interest Group, P.O. Box 1105, San Carlos, CA 94070, $16.00).

The QS manual is divided into chapters devoted to various classifications of FORTH words (such as INPUT/OUTPUT, number-handling, compiler words, etc.). In the back of the book are some useful sample programs and an extensive FORTH glossary (vital to the FORTH programmer), with a solid explanation of every FORTH word in the vocabulary.

The Editor

An important consideration of a FORTH system is the Editor, since you'll be using it to prepare your program for saving onto disk (although you can also compile FORTH words "interactively"). The QS FORTH Editor is very well done. It is a page-oriented text editor which permits you to type your program in as if you were using a word processor. You can edit programs as large as the disk can hold, by paging forward and backward through your screens at the touch of a single key. One disadvantage is that QS FORTH screens are only half as large as standard fig-FORTH screens, (sixteen 32-character lines rather than 64 character lines).

This ease of editing lasts only while in the Edit mode, however. When in direct mode, you cannot move the cursor around the screen and edit, as you can in BASIC or with the Assembler Cartridge. You cannot clear the screen without invoking an error message.

Features

QS FORTH supports the full fig-FORTH vocabulary, so let's look at the "extras" available. QS FORTH offers a utility package that adds several useful programming tools, such as a simple CASE structure, memory DUMP, Stack Print (shows contents of a stack non-destructively), 2DUP (double DUP), and LOCATE, a handy word that will list the source screen of a word, if it was compiled from disk. Also included is an I/O package that supports printer, cassette, screen, and keyboard input/output, using an approach similar to BASIC with OPEN, PUT/GET, and CLOSE. There is also a printer toggle that can switch output to the printer or screen. Included in the I/O package is a set of words to access Atari graphics. The similarity to BASIC continues here as well with words like GRAPHICS, SETCOLOR, PLOT, DRAWTO, etc. Use of the Atari's four-voice sound is also supported, with SOUND and XSND. XSND is a useful word that shuts off all sound, a task usually accomplished with END in BASIC.

The Assembler

QS FORTH includes a powerful assembler. Like most FORTH assemblers, the code for the assembler is remarkably brief, yet it supports source code with labels, structured control statements such as IF/ENDIF and BEGIN/UNTIL, and multiple statements per line. This assembler lacks many 6502 FORTH assembler standards such as N, IP, W, and only supports one macro return – PUSHOA. Nevertheless, it is quite useable.

Disk I/O

QS FORTH, again like most FORTHs, does not use the standard Atari DOS. It simply formats the disk into screens, and accesses this data directly, rather than indirectly, via DOS. This makes more disk memory available, and simplifies I/O, but it creates a compatibility problem. FORTH cannot read files produced by BASIC or even machine language. Despite this, QS FORTH expects new disks to be formatted by DOS II. There are also words to COPY screens or duplicate the entire disk using a sector-by-sector copier.

FORTH is coming into its own as a language. In the latest APX catalog, for example, two very impressive programs were in FORTH.

QS FORTH
Quality Software
6660 Reseda Boulevard
Suite 105
Reseda, CA 91335
$79.95

©

Review:

Hardbox For PET/CBM

Richard Mansfield
Assistant Editor

The old distinctions between "big" and "small," between mini- and microcomputers, keep breaking down. What would you call a PET with 80 megabytes (80 million bytes) of memory? It is possible to attach as many as four Corvus hard disk units with up to 20 megabytes each to a PET by using Small Systems Engineering's new Hardbox.

A typical setup would involve attaching the cable (which normally goes between your computer and a floppy disk drive) to the Hardbox instead. The unit is about as big as a medium-sized portable radio and connects directly to a Corvus hard disk drive. It acts as an intelligent controller and can handle up to four hard drives – hence, 80 megabytes.

It takes very little getting used to: the software is compatible with both PET/CBM DOS versions one and two so it will work with floppy files and existing programs. The manufacturer states that it "is designed to appear to the PET as a fast, high-capacity floppy disk unit" and that's what it does. Even the "wedge" is allowed. To scratch a file, for example, you could use either sC"name" or PRINT#1,"S0:filename" – the two, familiar DOS grammars. The Hardbox comes with a collection of utility programs for diagnostics, formatting, backup, and so on. The only unsupported DOS commands are memory-read, -write, and execute.

Data transfer is fast. Drive access time for ten or 20 megabyte drives is 40ms average and a five megabyte drive is 125ms average. The maximum record size is 255 bytes in relative files, the maximum number of records is 65535, and the maximum file size is 16 megabytes, either sequential or relative. On a five megabyte drive, more than 2000 files can be created with 13 open per Hardbox at any one time.

There is an issue to be resolved when using megabyte hard disks. How do you back them up? A 20 megabyte Corvus unit costs $5995. It may be impractical to buy a second unit to serve merely as backup. The solution? "Corvus Mirror" is supported by the Hardbox. An ordinary videocassette recorder can be attached as a backup device and it will copy the hard disk at a rate of 7.5K per second. A ten megabyte drive can be backed up in 20 minutes. Alternatively, a PET floppy drive or even a cassette unit could be used for backing up medium-size files.

Hardbox also supports multi-user environments. With an eight-way multiplexer, eight PETs can address the same hard disk at the same time, with individual and shared areas, passwords, and protected zones. Each Hardbox may be separated by up to 20 yards from the multiplexer and, with two levels of multiplexing, a maximum of 64 users can connect.

Hardbox
Small Systems Engineering
222B View St.
Mountain View, CA 94041
415-964-8201
$695

©

Book Review:

Beyond Games:

Systems Software For Your 6502 Personal Computer

Jim Butterfield
Toronto

This book covers 6502 machine language essentials for four machines: Apple, Atari, Ohio Scientific and PET. It does this by building a monitor, piece by piece; differences between machines are accomodated by a "system data block" which covers such things as the length of a screen line, input/output routines, etc.

The Approach

The author spends four chapters outlining the characteristics of the 6502 in a fairly easy and pictorial manner; it's not a rigorous treatment, but users will find it non-intimidating. In chapter five he begins writing some "useful" subroutines, making allowances for the differences between the various machines.

Starting at chapter six, we have an objective: let's write a "visible monitor" which will do many of the things a monitor system does: displaying registers and memory and allowing changes. The monitor builds on the subroutines previously established.

The initial monitor looks rather like that of the KIM; it allows only one memory location to be displayed or changed, plus various registers. Subsequent chapters add features to the monitor, giving such facilities as hexadecimal dumps, disassemblies, other utilities and a simple text editor.

Appendices give the various program modules in Assembler Code, Hexadecimal dump format, and BASIC DATA statements (for POKEing). Oddly, the "system data block" is not supplied in hexadecimal.

The Monitor

Readers should accept the "visible monitor" as an exercise, not as a powerful working tool that they will depend upon in the future. Some essential features are missing, especially interfaces to external code: breakpoints and GO. The monitor takes up a lot of room – 4K of memory – in view of what you get. Most machines will have good monitors already in place.

But the monitor itself isn't the object of the game. Getting there is most of the fun, or at least the education, and users should seek to do more than just type in coding lines: they should try to understand the objectives and methodology.

Some routines are written, but not used immediately. At the end of chapter 7, there are two exercises to try to overcome this problem; it would have been useful to have many more of these to aid checkout of the various modules. If the user is going to write a routine, he's better off if he can put it to use right away.

Perhaps the single most useful part of the book is the way that it compares the various 6502 machines.

There are some operational features of the visible monitor that could be streamlined. The prompt which asks the user to supply an address should flip him into the address mode to guard against errors, for example. Some of the missing features might be considered "student exercises"; the reader should be able to improve on things he doesn't like.

The monitor doesn't deal with input/output other than the keyboard and screen. The logic makes room for a printer, but the coding to make the connection isn't in place. Activities are confined to the screen.

Perhaps the book's title is misleading in at least this sense. A prospective buyer would be likely to think: "Beyond Games? Must be business applications." If so, he might be disappointed to find a book that is concerned with inner space – memory manipulation and hexadecimal notation rather than business-oriented techniques.

The Material

The book's approach to learning coding is very matter-of-fact. Theory is not stressed and the material is presented pleasantly in a conversational manner.

There are a few errors; generally, they are minor in nature. In a discussion of indexing, page 21, it is stated that "...the 6502 will operate on some address higher (...or equal to the base address...) in memory." That's not always true for zero page indexing.

The author has quite a bit of trouble with the Compare instruction and the associated flags. Page

23 incorrectly shows the condition of the N flag after a compare (it might be easier to mark this flag "not relevant"), and a footnote on the same page states: If you wish to test the status of the Carry flag after a Compare, you must set it before the compare. Wrong. The sample coding carries through the error in part: page 55 gives the sequence CMP #$0A; BMI ... which is bad logic (it should be BCC) but does work in this case since the range of numbers in A is limited. Elsewhere, this type of sequence is given correctly – address 13DA has satisfactory code – did the author learn as he wrote the book?

The output of the disassembler is not satisfying. Indirect indexed addressing should show the addresses in zero page mode. The use of the dollar sign to signify hexadecimal is not consistent within the disassembly. Unidentified op codes are translated as BAD, which I tend to read as Binary Add; and one byte is almost inexplicably translated to TEX.

The hex dumps are hard to read. I had repeated problems distinguishing the digit 8 from 9 or B. The dump labeled D3 has an incorrect byte at $13D2; at least, it's incorrect until you plug in the exercise given much later in D11.

The book is nicely written and the program listings well commented. The visible monitor itself is probably not a useful end product, but the learning process along the way can be very valuable.

Perhaps the single most useful part of the book is the way that it compares the various 6502 machines. Several appendices compare the character sets, screen mapping, and simple input/output characteristics of the various machines. Even if a reader knows machine language, he may find value here. The broader characteristics of the machine (other I/O such as disk and printer) is not dealt with in any depth.

I hate the title: it's not relevant. "Beyond BASIC" might have made more sense, since the objective is to teach machine language techniques. After all, games – very good games – can be written in machine language; and monitors can be written in BASIC.

In many ways, it's a pleasing book. It's one of the few that attempts to teach you 6502 machine language with reference to the actual machine in which the language will reside.

Beyond Games: Systems Software for Your 6502 Personal Computer
by Ken Skier
BYTE/McGraw Hill
70 Main Street, Peterborough, NH 03458
432 pages. $14.95 ©

Book Review: 6502 Assembly Language Routines

Jim Butterfield
Toronto

Here's a book containing over 40 subroutines to do some of the most common jobs in 6502 Assembly language.

The subroutines correspond to some of a programmer's most commonly needed tasks. Input, output, arrays, arithemetic and string functions are dealt with. The end result is a handy recipe book of frequently needed operations.

The emphasis is on utility functions. Complete packaged programs are not offered here; the reader is supplied with the coding chunks he needs for the bread and butter activities. Want to input a decimal number in ASCII and convert it to binary? You'll find the information here. The same is true if you want to output, to use hexadecimal numbers, or convert to the EBCDIC character set.

Organization

The first 160 pages or so might be termed, "A 6502 review manual." The characteristics of the 6502 are reviewed in the first chapter, entitled General Programming Methods. The second chapter seems directed to users from other processor environments: it shows how to implement features they might view as "missing" from the 6502's instruction set. For example, how do you negate the value in the A register? Exclusive OR with hexadecimal FF and then add 1, that's how. Users without experience from other environments will still find this a fascinating collection of simple techniques.

Chapter 3 is a beauty. Entitled "Common Programming Errors," it itemizes in detail some of the most frequent mistakes that 6502 programmers make. It's valuable reading. Indeed, perhaps the authors should have read it themselves one last time: they might have spotted their minor coding error on page 182, where they use BPL after a comparison. As page 136 sternly points out, "In comparing unsigned numbers, the Carry flag indicates which number is larger."

The program section starts on page 157. At this point, there are details on how the subroutines are organized. There's also a list of the subroutines themselves, which might have more properly been given in or near the Table of Contents at the start of the book.

The programs are broken into categories: Code conversion, Array Manipulation, Arithmetic, Bit Manipulation, String Manipulation, Array Operations, Input/Output and Interrupts. The book concludes with three appendixes containing general reference material, a Glossary, and an Index.

Program Style

It should be emphasized that the programs are supplied in assembly language, not machine language. The reader will not see the final machine code; he's expected to take the "source coding" and put it through his own assembly process. Similarly, the user will choose the location of the program and its variables when he performs the assembly. This gives the book's programs a rather abstract feel even though they can be assembled into working code very quickly.

The assembly language is non-standard for the 6502 environment. Where most coders would write hexadecimal FF as $FF, the book writes 0FFH. Where the standard assembler code to set aside two locations should be *=*+2, the book uses .BLOCK 2. This seems to be an offshoot of Adam Osborne's determination to hew to a universal assembly language standard. Unfortunately, there doesn't seem to be a 6502 assembler using such a standard; as a result, the book becomes a little more difficult to read and use.

Each subroutine comes with excellent formalized documentation. Memory space used, execution time and many other relevant factors are carefully noted. There is a variety of styles, particularly in the area of passing values to and from the subroutine: sometimes it's a common area, sometimes the registers, sometimes the stack. The knowledgeable reader should have no trouble adapting this to the particular style wanted. It's all well spelled out.

The coding is straightforward and non-clever. You won't find any elegant and obscure algorithms in this collection; and that's probably good. If you decide to try your hand at adapting a program to your own special requirements, it will be unlikely to jump up and bite you. The code takes the obvious approaches; and this is particularly good for learning.

Sometimes the authors err on the side of caution. In Multiple-Precision Binary Addition and Subtraction, for example, they specify that the routines add and subtract unsigned binary numbers. In fact, the same routines work on signed numbers.

Coverage

The emphasis is on integer arithmetic. Reasonable enough considering that the vast majority of machine language applications are integer-oriented. Some readers might wish for information on floating-point or even fixed-point fractions; they won't find it here.

Arithmetic is limited to the four basic functions. This is to be expected where numbers are fixed point. Don't look for square root routines or more advanced material here ... it's all fundamentals.

Many of the arithmetic routines assume that all values have the same size (same number of bytes). This may need to be adjusted by the user: for example, it is often useful in division to have a small divisor (or quotient).

Strings are "conventional," limited to a maximum of 255 bytes. Users may find it necessary to shift the coding around to suit their own particular organization. The book assumes that string length is supplied as a value. Other systems – signalling the end of a string with a zero byte or a carriage return – would need some minor adjustments.

The only sort given is a bubble sort. That has the advantage of simplicity, but perhaps readers should be warned that it's rather inefficient and they should keep their eyes open for other, more efficient, techniques.

The RAM Test program is naive. The most common RAM failure is not caused by storing information and not being able to read it back. It's a result of interference between addresses: store something at one address and it will also appear in another location. You'll never catch this kind of thing by storing and checking a single byte at a time.

I missed random numbers. There are several easy techniques, and at least one might have been usefully supplied.

It's a highly useful reference book. The introduction is a good summary of the 6502's working characteristics; the programs are useful ones that can be put to work.

The book is especially well adapted to a user who is moving to the 6502 from other processors. Complete beginners might find that the book is too fast for them, but the information is all there.

6502 Assembly Language Subroutines
by Lance A. Leventhal and Winthrop Saville
Osborne/McGraw Hill, 630 Bancroft Way
Berkeley, CA 94710
550 pages. $16.99

©

COMPUTE! *will be covering FORTH regularly. FORTH is sometimes mentioned as an alternative to machine language: it is easier to program than machine language and it runs faster than BASIC. However, machine language itself can be easily woven into a FORTH program when greater speeds are necessary.*

The FORTH Page

Machine Language And FORTH

Richard Mansfield
Assistant Editor

What language you decide to use is frequently a question of speed: how fast you can program with it or how fast the program itself will later execute. One of the advantages of FORTH is that most people find they can program in it faster than in machine language. And FORTH executes at roughly 20 times the speed of a comparable BASIC program.

Machine language runs at maximum machine speed. There is no "interpreting" going on during execution, events tumble past one another. Nothing has to be translated; machine language *is* the machine's native language. Things do not so much *mean* something in machine language – they *are* that something. The drawback is that the programmer must spend extra time (sometimes a great deal of extra time) writing the program itself.

FORTH is a curious language, however, from several points of view. Some programmers take to it immediately and fiercely, never returning to BASIC. They find FORTH the most natural, most efficient way to communicate with their computer and they can program faster in FORTH than in BASIC.

Others find FORTH difficult, bizarre even. FOR I = 1 TO 10 becomes 11 1 DO in FORTH. There are three to four times as many statements and commands (called *words*) in FORTH as there in the BASIC vocabulary. So choices are multiplied: there can be dozens of ways to get there from here.

Many versions of FORTH include an assembler. Like programming in FORTH itself, FORTH assembling can have a remarkable clarity and simplicity – once you're accustomed to the oddities.

If a FORTH program is executing too slowly, you can look for the loop where the most frequently repeated action is going on. Replacing key parts of this loop with machine language can greatly speed up the program run.

... FORTH assembling can have a remarkable clarity and simplicity – once you're accustomed to the oddities.

An Example Of FORTH Code

Take a look at Screen 100 (Program 1). This is an example of a machine language subroutine within a FORTH routine. The small program called TEST is expected to search through any screen, looking at each byte for whatever character has previously been stored in the variable 1STCHAR. It prints the addresses of any matches.

To execute TEST, you would type 10 BLOCK TEST to find all matches, on screen ten for example. Above TEST is the machine language word ?CHAR which is used by TEST to make each comparison. This might seem a roundabout way of doing things, but it does result in roughly triple the speed of a comparable subroutine for ?CHAR written in straight FORTH.

Where an ordinary FORTH word begins with a colon (:), machine language is invoked with the word *CODE* which changes the number base to hexadecimal and sets up the assembler. Program 2 is a disassembly of ?CHAR. It is useful, especially when first working with FORTH assembly, to be able to easily disassemble and study the code you create. Notice that 1 # LDA, is the traditional LDA #$01. FORTH, as always, expects reverse notation; in assembling, the format is operand-addressing mode-mnemonic.

SETUP JSR, sends the machine to a subroutine which eases communication between FORTH and machine language. SETUP transfers bytes from the FORTH stack to a temporary, eight-byte holding area, called *N*, which starts at address $10 (on the PET). It will move items in two-byte chunks since FORTH operates on two bytes at a time. It knows how much to move by looking at the accumulator. We put a one into the accumulator, so it moves a

THE MILLION DOLLAR WINNER WORDCRAFT ULTRA

The word processing package that's EA$Y to use

Used on thousands of COMMODORE$, the micro computer that's EA$Y to operate

It's EA$Y to $ell and only sold through Dealers

The reasons it's 7 ways EA$Y

1. Screen layout matches printed document.
2. Designed for stenographers and secretaries to use.
3. Full choice over horizontal and vertical spacing.
4. True inter-character proportional spacing.
5. Multiple-user:— up to 8 screens can be connected.
6. Full text communication between Wordcraft users with automatic error recovery.
7. Comes complete with comprehensive manual & self-pace training guide.

FOR INFORMATION, SO EA$Y TO WRITE OR CALL:

WE$T COA$T: CIMARRON
666, Baker Street,
Suite 319
Costa Mesa, CA. 92626
Telephone: (714) 641 1156

EA$T COA$T: COMPUTER MARKETING $ERVICE$
300, W. Marlton Pike,
Cherry Hill, N.J. 08002
Telephone: (609) 795 9480

two-byte unit off the FORTH stack (this is the I, the address within a screen to be checked, generated by the loop in TEST). If we'd used 02 # LDA, four bytes would have been transferred from the stack to N.

Following SETUP, the address of the target byte is at N (properly in low-high 6502 addressing format), so we can load the accumulator with indirect Y to fetch the byte from memory. Two points: when you descend to machine language from FORTH, the Y register is set to zero – you need not LDY #$00. Also, the X register is used by FORTH as the pointer to its stack. If your machine language subroutine will in any way affect X, you should XSAVE STX, before anything else and XSAVE LDX, before returning to FORTH. XSAVE, like N, is a temporary holding space which is a safe place to keep the X register.

Line three illustrates another way that machine language easily communicates with FORTH. The variable 1STCHAR might contain a decimal 76 (ASCII code for the letter *L*.) This would allow TEST to find all the L's on screen 100 if we later typed 100 BLOCK TEST. To get this 76 which is in 1STCHAR into the machine language code, we need only use the word itself, 1STCHAR, which (as in FORTH) will leave behind the address where the variable's value is in memory.

Provisions For Branching

In any event, the value in the accumulator is then compared with the value located in variable 1STCHAR and line four puts a one on the FORTH stack if the compare is true or line six puts a zero on the stack if the bytes are not equal. When control is returned to TEST, it will use the zero or one to decide whether or not to print an address on the screen.

The FORTH assembly process can provide for forward branching. The IF, THEN, structure uses the 0= to test the result of the CMP,. As Program 2 illustrates, a failure of comparison (BNE) is assembled in place of the 0= IF, to skip over the true flag creation of line 4. It uses the address of THEN, to know where to go. In both cases, however, we exit via PUSHOA JMP,. This sends control to a subroutine that returns to FORTH after first pushing onto FORTH's stack a two-byte version of the number found in the accumulator.

There are other ways to return to FORTH. If no stack manipulations are required, NEXT JMP, is the common exit. Some adaptations of FORTH require a word which signals the end of assembly. In the version illustrated here there is no such requirement, but it is helpful to return to decimal mode at the end of a machine language word.

One final note, because FORTH assembly allows multiple mnemonics per line, subroutines can be formatted in a structured way. A comparison of the readability of ?CHAR in Program 1 with its disassembly in Program 2 demonstrates the clarity it is possible to achieve. In addition, comments can be included anywhere by enclosing them in parentheses with a space following the first parenthesis: (comment).

Program 1.

```
SCR # 100
 0 76 VARIABLE 1STCHAR
 1    CODE ?CHAR
 2             1 # LDA, SETUP JSR, N )Y LDA,
 3                   1STCHAR      CMP,
 4          0= IF, 1 # LDA, PUSH0A JMP,
 5      THEN,
 6                 0 # LDA, PUSH0A JMP,
 7                          DECIMAL FORTH
 8
 9
10 : TEST DUP 1024 + SWAP      DO
11                           I ?CHAR
12                   IF    CR I .   ENDIF
13                            LOOP ;
14
15
```

Program 2.

```
3A34 A9 01    LDA #$01
3A36 20 6A 06 JSR $066A
3A39 B1 10    LDA ($10),Y
3A3B CD 3A 39 CMP $393A
3A3E D0 05    BNE $3A45
3A40 A9 01    LDA #$01
3A42 4C 70 09 JMP $0970
3A45 A9 00    LDA #$00
3A47 4C 70 09 JMP $0970
```

©

Odds & Ends: Atari

Add A Text Window To GRAPHICS 0

Charles Brannon
Editorial Assistant

The text window can be a useful feature in the graphics modes, enabling a simultaneous text and graphics display. The text window is very similar to a miniature GRAPHICS 0 text screen: all the editor functions are supported, and scrolling and screen clearing are confined to the small four-line window.

This same capability would be useful for a GRAPHICS 0 display. For example, a menu, a list of choices, could be presented in the top twenty or so lines of the screen, and the user's input taken in the lower four lines of the text window. Any errors, such as the user typing editor keys in an INPUT statement, would not interfere with the rest of the screen. Conveniently, any scrolling when caused by a line like this one:

```
150 PRINT "NAME";:INPUT N$:IF LEN(N$)>8
    THEN PRINT "* TOO LONG *":GOTO 150
```

would not cause the menu above it to scroll as well.

How is all this done? With a single POKE statement. Location 703 normally contains the number 24. If you POKE a four in its place, the cursor is zapped to the bottom of the screen and the text window is in place.

Note that you can't print to the upper part of the screen with PRINT statements; you have to use PRINT#6 as you do with GRAPHICS modes 1 and 2. The POSITION statements also only affect the upper part of the display, you must use POKEs to position text window output.

Here is an example program to demonstrate the use of the window. It is a simple disk menu program. Notice that you don't need to use PRINT#6 to print to the upper part of the screen until after the POKE 703,4 takes place.

```
100 REM DEMONSTRATES "TEXT WINDOW"
110 REM SIMPLE MENU PROGRAM
120 REM FOR DISK DRIVE
130 TRAP 150
140 OPEN #1,6,0,"D:*.*":GOTO 160
150 ? "Can't read directory":END
160 GRAPHICS 0:COL=0:POKE 752,1:REM DISA
BLES CURSOR
170 DIM A$(20),F$(14):TRAP 230
180 INPUT #1;A$
190 POSITION COL,LINE:? A$(1,14)
200 LINE=LINE+1
210 IF LINE>20 THEN COL=COL+13:LINE=0
220 GOTO 180
230 POKE 703,4:REM CREATES TEXT WINDOW
240 FOR I=1 TO 100:? I,:NEXT I:REM ONLY
FOR DEMONSTRATION
250 ? "{CLEAR}Run which program";:INPUT
A$:REM CLEAR ONLY CLEARS WINDOW
260 TRAP 290
270 F$="D:":F$(3)=A$
280 RUN F$
290 ? "Can't RUN ";F$;"."
300 END
```

©

Here's an alternative to Apple Pascal's Transfer function when you need hardcopy listings.

Apple Pascal Lister

Scott Barrus
Ramsey, NJ

The lister program below is written in modules. The first is self-explanatory. It puts a header on the top of the screen. The second module goes with the first. It is used to delay long enough for a person to read the header. If a shorter or longer time is desired, the number 3000 can be changed.

The module Input puts the prompt "File name" on the left-hand side of the screen, 15 spaces down. It also names a string "namefile" to be used with the names module. If a name is entered without a ".text" on the end, this module attaches it. A possible addition would be to prompt and ask if ".text" were wanted.

The module Names builds and writes the Filename and page # on the top of each sheet. It also then puts 79 "=" 's underneath the filename line. Since Pascal does not have a tab function, either a tab type module would have to be written or, just use spaces to separate the word "page" and file names. If a further right position were wanted, then more blanks could be added or a tab module made up.

Getfile does the hard work. It reads a line from the input file and then writes each line to the printer. It does this 55 times and then sends a form feed to the printer. It will keep doing this until it comes across an end of file (EOF) marker at which time this module ends.

Openfile opens the avenue for the program to work. It opens the text file to be read and it activates the printer as a file named outfile.

Procedure Epson is a bit of customizing because the author uses an MX-80 printer. Since the program can print more than one file at a sitting, the change mode option was added to give the user flexibility. If the printer being used does not have all the capabilities of the Epson, either eliminate the lines that cannot be used or eliminate module Epson. If different codes are used on any printer, those codes can be substituted for the Epson codes.

The final section is the body of the program. The program starts by assigning the value of one to "pagenumbers." A header comes next. The file must be opened so that when we state what Epson code is to be used the printer will be able to get the code. All that is left is to let the program get (and print) out the files. When done, the user is prompted for another file. If none is wanted, the screen is cleared and the user gets a cheerful message. Finally the file opened is closed and the program ends.

To run the program, type X from the main prompt line in Pascal. When asked what file to execute, type in the name that was assigned to the compiled code. To run the program as on the example disk, type in List:Lister.code. The program will be off and running.

```
Program Lister;

  const

     Pagelength = 55;
     Linelength = 79;

  var
    files,
    Pagenumbers,
    time               : Integer;

    Filename,
    Namefile           : String;

    Line,
    Nametitle,
    Separator          : String [ 255 ];

    Infile,
    Outfile            : Text;

    Ans,
    answer,
    choice             : Char;

Procedure Header;
   begin
     page(output);
```

```
    Writeln('     ******************************');
    Writeln('     *                            *');
    Writeln('     *     Apple Pascal Lister    *');
    Writeln('     *       by Scott Barrus      *');
    Writeln('     *                            *');
    Writeln('     ******************************');
  end;

Procedure Wait;
 begin
   for time := 1 to 3000 do;
 end;

Procedure Input;
 begin
   Page(output);
    Gotoxy(1,15);
    Write('File name: ');
    Readln(filename);
     Namefile := filename;
       if pos('.',filename)=0 then
         filename := concat ( filename,'.text');
 end

Procedure Names;
 begin
 Write(outfile,chr(14),namefile,chr(20));
 Writeln(outfile,'                        Page ',pagenumbers);
   Separator := '=';
  for files := 1 to linelength do
   Separator := concat(Separator,'=');
 Writeln(outfile,Separator);
 Writeln(outfile);
 end;

Procedure Getfile;
 begin
  Repeat
    Names;
    For files := 1 to Pagelength do
      begin
        Readln(infile,line);
        Writeln(outfile,line);
      end;
    Page(outfile);
    pagenumbers := pagenumbers + 1 ;
  Until EOF(infile) = True;
 end;

Procedure OpenFile;
 begin
   Rewrite(outfile,'printer:');
   Reset(infile,filename);
 end;

Procedure Epson;
 begin
   repeat
   Page(output);
     Writeln('Choose:');
     Writeln;
```

```
    Writeln('(A).......Emphasize');
    Writeln('(B).......Double Strike');
    Writeln('(C).......Both');
    Writeln('(D).......Cnx Either');
    Writeln('(E).......Continue as is');
    Writeln;
    Write('choice: ');
    Readln(choice);
  until choice in [ 'A','a','B','b','C','c','D','d','E','e'];
  case choice of
    'A','a'   : Writeln(outfile,chr(27),'E');
    'B','b'   : Writeln(outfile,chr(27),'B');
    'C','c'   : Writeln(outfile,chr(27),'E',chr(27),'B');
    'D','d'   : begin
                Page(output);
                 repeat
                  Writeln('Cancel which?');
                  Writeln('(A)...Emphase');
                  Writeln('(B)...Double Strike');
                  Writeln('(C)...Both');
                  Writeln('(D)...Continue as is');
                  Write('Choice: ');
                  Readln(choice);
                 until choice in ['A','a','B','b','C','c','D','d'];
                 case choice of
                  'A','a'  : Writeln(outfile,chr(27),'F');
                  'B','b'  : Writeln(outfile,chr(27),'H');
                  'C','c'  : Writeln(outfile,chr(27),'F',
                                         chr(27),'H');
                  'D','d'  : Exit(Epson);
                 end; [ case ]
              end;
      'E','e'   : Exit (Epson);
   end; [ case ]
end; [ epson ]

 Begin [ Main Program ]
    pagenumbers:= 1;
    Header;
     Wait;
    Repeat
       Input;
       Openfile;
         Epson;
       Getfile;
         Write('Another file? (Y/N) ');
         Readln(answer);
   Until answer in ['N','n'];
   Page(output);
   Gotoxy(1,15);
   Writeln('Have a Nice Day!');
   Close (infile);
end.
```

©

Hooking Up DOS Wedge To PETs With Power

Elizabeth Deal
Malvern, PA

Here is a cookbook procedure that allows the use of Commodore's DOS support and/or the Universal Wedge with PETs equipped with Professional Software's Power. It is only for Upgrade PETs. BASIC 4.0 systems don't really need the Wedge.

Follow These 6 Steps Once:

1. Reset the PET and turn POWER on with the usual SYS9*4096. 2. Load the Wedge program. Then get into the monitor with SYS4. 3. Make the listed changes, being careful to make only the changes appropriate for your version of the Wedge program: Old DOS-SUPPORT needs a change in the contents of $0749/A, $074E/F and $0798/9 from 00 76 to FE 97 (jump to Power instead of the CHRGOT routine). It also needs a replacement of the four bytes at $072C to $072F from the existing C9 40 F0 0D to EA EA EA EA (to disable the '@' key). Universal Wedge needs changes in the contents of $0549/A, $054E/F and $0598/9 from 00 76 to FE 97, and replacement of the four bytes at $052C to $052F from C9 40 F0 0D to EA EA EA EA. Remember to push RETURN after each change so your PET can hear you. 4. Save the program under a different name, "PDOS" perhaps. 5. Now RUN the Wedge. If the PET crashed, go back to step 1. 6. Get into the monitor once again and ask for bytes $0070 to 00723 (.M 0070 0072). Write down the two bytes that follow 4C and don't lose the note. This is the wedge's address in 6502 format, low byte first.

In The Future:

Turn Power on, load your newly created wedge, RUN it in the usual fashion, and always remember to write down the contents of $0071/72. (You may need this later, as we shall see.)

What's Happening?

After Power has been set up, we take over and tell the PET to check our Wedge routine before it goes to Power. That's done by the change in $0070/71. In three cases (no recognized wedge command, error channel read and utility commands), the Wedge routine formerly exited to the CHRGOT subroutine ($0076). We force it to go to Power with JMP $97FE (six bytes beyond the start of Power), which is the very change you made in three places to the Wedge code. Printing the directory, load and load/run sections remain intact.

Finally, we eliminate the '@' key from Wedge's repertoire in order to leave it for a more important task in Power's search routine. We did it by replacing four consecutive bytes with $EA (NOP) to skip that code. The '>' (greater than key) and all others remain unchanged.

Assorted Notes

This is just one of many ways the hookup can be achieved. It loosely follows the instructions on page 62 in the Power manual. I think a better way is that described on page 63. However, the method presented here is the simplest to understand and explain. The method on page 63 requires some fiddling with the Wedge code and would require more typing on your part.

You have to remember the address of the Wedge, which is in the two bytes you jotted down. You can use instant key REM macros for remembering them. The reason is that if you use Power's FIX command, Power will stick the old values (its own address) into the CHRGET routine ($0070) and you will have to change them back to point to the Wedge.

If you use the OFF command, Power will replace your address with the regular CHRGET code, also disabling the Wedge. It's best to leave the Wedge disabled until Power is turned on again. Otherwise, if you enable the Wedge by changing the values in the CHRGET routine you will ultimately end up in Power, since that's where we told the Wedge to go when its done. For the same reason you should not use this version of the Wedge if you don't plan to use Power.

Extra Note On The Wedge

I have written recently that the Universal Wedge overshoots its correct loading boundaries when it is RUN. I also incorrectly stated that the old DOS-support does not. Sorry! Both programs do it. The reason is that, as coded, two pages (512 bytes) of code are moved to high memory but the code reserves only enough room for the Wedge code (359 bytes). You can write your own mover as a fix. If not, and if you are loading the Wedge to a place other than near the top of memory you may want to lower the top of memory pointer to accomodate the entire program, otherwise you'll wipe out whatever code you have already placed above the Wedge. This can be done by entering FF in place of 67 in byte $08B2 in the DOS support, or in byte $066B in the Universal Wedge, if you have the room. ©

A Self-Modifying P/M Graphics Utility

Kenneth Grace, Jr.
Colts Neck, NJ

The utility in Program 1 sets up a skeleton program for player/missile graphics. It presents a series of questions about the P/M situation you want to create and then modifies itself according to your responses. I hope you will find the program useful. But I also hope it will stimulate your thinking on other ways to use the self-modification ability, introduced by Bruce Frumker in **COMPUTE!** (August, 1981 #15). I also want to give some further publicity to a method for controlling P/M motion using string manipulations, as introduced by George Blank in the April 1981 issue of *Creative Computing*.

There are several steps involved in setting up P/M graphics, and they have been covered in **COMPUTE!** and elsewhere. The steps are easy, but there are several choices available along the way (resolution, number of players, colors, positions, etc.). That's where this utility comes in. It contains all the basic steps, and where there are choices to be made, they are presented to you. The program then uses Frumker's technique to add lines to the program. At the end it uses the same technique to delete lines that are not needed, including the lines which ask the questions.

When the utility has finished, you are left with the skeleton of a P/M graphics program. You can RUN it at this point to check things out. But to make it a real program, you will have to draw the playfield and add the main loop for controlling motion, checking collisions, etc. In other words, the utility does just the P/M setup. However, note that there are some subroutines included which you can use for player motion.

Since I make extensive use of Frumker's technique, I have split it into two subroutines, at 9900 and 9910-9920. Between the two subroutine calls I put PRINT statements for the lines to be added to, or deleted from, the program.

Aside from these two subroutines, the heart of the program is in lines 3-66. Lines 3-55 present the series of questions through which you define your particular P/M arrangement. The self-modifying feature is used after every question or two to add the appropriate statements to the section beginning at line 9000. At a few places the program STOPs while you enter lines containing SETCOLOR statements for the playfield colors or DATA statements containing the bytes defining the shapes of the players and missiles.

At the end of this sequence lines 56-66 and 9930 delete lines 3-66, the unneeded P/M motion routines in 100-185 (when you have less than four players and four missiles), and other unneeded lines in 9000-9700. I couldn't figure out how to get rid of all the unnecessary lines, so later you may wish to delete lines 9900-9960.

When the utility has finished, you are left with lines 1-2, 99, appropriate subroutines from 100-185, a trivial loop at 200, and the P/M setup steps starting at 9000. Note that lines 20-24 and 9030 take account of Fred Pinho's rules (**COMPUTE!** September '81, #16) for placing P/M memory so that it doesn't overlap the memory for the BASIC GRAPHICS mode. Starting from this skeleton, I suggest that you use lines 3-98 for REMarks, opening titles, instructions, etc., and begin your main program at line 200.

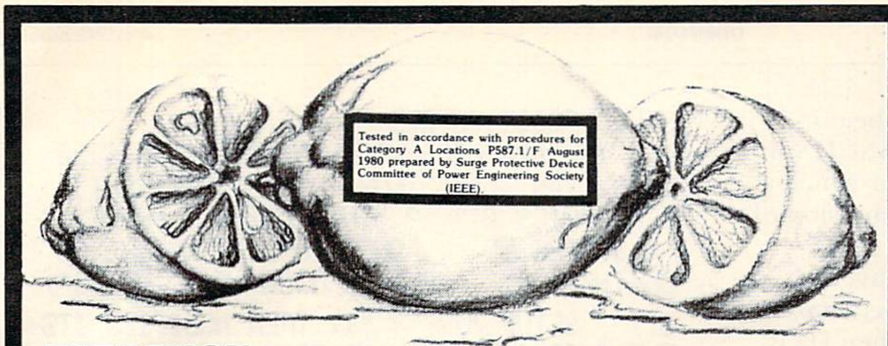
Motion Using Strings

The string manipulation method for player motion is based on Mr. Blank's column mentioned above. I refer you to that article for a detailed explanation. The basic idea is that you trick your Atari into treating the Player/Missile memory as the string array storage area for P0$,P1$,...,M$. Lines 1-2 and 9500-9580 do this. Atari's fast string handling routines can then be used for vertical motion of the players or for animation.

In order for this to work, P0$,P1$,...,M$ must be the first variables mentioned in the program. I suggest that you turn off power momentarily, then key in line 1 and the rest of the program. In line 2, VTAB is a pointer to the start of the variable table, which contains eight bytes for each variable. ATAB points to the start of the string array table, which is where the actual values are stored. Each pass through lines 9530-9570 modifies the eight bytes for P0$ (or P1$, etc.) in the variable table, including the offset from ATAB where the actual values for P0$ are stored (the P/M graphics memory).

The bytes defining the players are stored in strings D0$,D1$,... at lines 9090, 9140, 9190, etc. Each character in a string is stored in memory as a byte containing the corresponding ATASCII value. In this case, we want our data BYTE treated as though it were already an ATASCII value, so we use D0$(I,I) = CHR$(BYTE). Note that this is a different way of using strings for P/M from Alan Watson's method (**COMPUTE!** September 1981, #16). The demo mirrors Watson's example.

The descriptions D0$,D1$,... are initially read into P/M memory (i.e., into P0$,P1$,...) at lines



THE LEMON Electronic Protection Devices

THE LEMON is a solid state shunting device that can be easily plugged into any standard duplex outlet. It eliminates disruptive and possibly damaging transient disturbances (spikes) which shorten equipment life, reduce efficiency, cause "hiccups" and many other malfunctions. THE LEMON has 6 outlets designed for 3 wire grounded plugs (also accepts 2 wire outlets) at a maximum load limit of 15 Amps.
Cat. No. 3516 **$54.95**

400/800 TECHNICAL USER NOTES Atari

Another fine reference manual for the Atari 400 or 800, the 400/800 TECHNICAL USER NOTES details the hardware as well as the Operating System of the Atari computer system.
Cat. No. 3141 **$29.95**

SAVE SAVE SAVE

Buy both the DE RE ATARI and the 400/800 TECHNICAL USER NOTES and save 10%!!! BOTH BOOKS FOR ONLY $44.95!!

DE RE ATARI ATARI APX

DE RE ATARI is the definative guide for Atari owners who wish to fully exploit the programming powers of the Atari 400 or 800 computer. Written by the programming "wizards" of Atari, this book is sure to become a valuable addition to any Atari owner's library.
Cat. No. 3789 **$19.95**

WORDMANIA Syncro, Inc.

Another fine piece of software from SYNCRO, WORDMANIA is based on the use of anagrams. The game is played in real time, with self adjusting levels of play. There are over a thousand words and the combinations are selected at random so the same word will appear in a different combination each time. A challenging device is used for the player to beat, score is based on time accumulated. The program has been developed so that a child and adult can play at the same time and the levels will adjust accordingly. The program is in color, large letters, and utilizes sound effects.
Cat. No. 3907 Atari, 16K, cass **$14.95**
Cat. No. 3908 Atari, 24K, disk **$19.95**

FLIP 'N' FILE Innovative Concepts, Inc.

The ideal solution for keeping your Atari 400/800 ROM cartridges safe from dirt or loss. Each FLIP 'N' FILE is molded from attractive, high-impact smoked plastic and holds up to 10 cartridges.
Cat. No. 3790 **$24.95**

VIDEO COMMAND CONTROLLER FOR ATARI

Zircon Int'l.

The "all-in-one" controller for both Atari computer systems and the Atari VCS. Incorporating a quality 8 digital switch mechanism, VIDEO COMMAND allows left and right control, forward and back movements, right and left rotation, diagonal movement, as well as up and down action of the fire mechanism. All this at a real low price-how can you refuse?!?
Cat. No. 3902 **$14.95**

HOW TO ORDER

Write or phone. Pay by check, M/C, VISA, or COD (add $1.50 for COD). (800) 423-5387 (213) 886-9200
Offer expires June 30, 1982.

Mention this ad and we pay shipping (UPS ground only).
HW Electronics 19511 Business Center Dr. Dept. G6
Northridge, CA 91324

WHEN IN SOUTHERN CALIFORNIA, VISIT OUR RETAIL STORES

HW ELECTRONICS

19511 Business Center Dr.
Northridge, CA 91324

2301 Artesia Blvd.
Redondo Beach, CA 90277

444 S. Indian Ave.
Palm Springs, CA 92262

SOFTWARE
FOR YOUR 16K TRS-80 COLOR, MODEL I, III ATARI 400/800, APPLE II

NOW YOUR COMPUTER CAN ACT LIKE A COMPUTER SHOULD!!! WITH OUR MAGNETIC MAGAZINES, YOUR MICRO WILL ENTERTAIN, EDUCATE, AND CHALLENGE YOU.

appletree™ MAGATARI™ TRC™

On cassette or diskette*, you will receive 6-10 programs every month ranging from games, home entertainment, and personal finance, to more of our unique "teaching" programs.
Our next issue will include:

1. **HOME INVENTORY**
2. **Reversi Board Game with Graphics**
3. **Generating Music and Sound Effects**
4. **Machine Language Programming (I)**
5. **Utility of the Month (II)**

You've already missed Stock Market Analysis, Concentration, Checkers, Golf, Algebra and Geometry Tests and Football. ☆ Later issues will include Backgammon, Chess, all Tax Schedules, Baseball, and many more. So don't let another issue pass you by! Subscription price for cassette is $50 per year, $30 per ½ year and $10 for a trial issue; for diskette $75 per year, $45 per ½ year, and $15 for a trial issue.
☆(Back issues available on request)

THE COLOR ACCOUNTANT

With our complete home accounting system, keeping track of your finances can be as easy as PI.

1. **Chart of Accounts Maintenance**
2. **Complete Checkbook Maintenance**
3. **Tax Deduction Flags**
4. **Net Worth Statement**
5. **Income Less Expense Statement**
6. **Complete Graph Package (graphs any files)**
7. **Payments and Appointments Calendar**
8. **Home Budget Analysis**
9. **Mailing List**
10. **Schedule 1040 (Long Form)**
11. **Decision Maker**
12. **Schedule G (Income Averaging)**

All programs are menu-driven and allow add/change/delete. Also, all files and statements can be listed to screen or printer, and saved to cassette or diskette. The cost of THE COLOR ACCOUNTANT is $75 for cassette and $80 for diskette. This package requires 16K except for the ATARI 400/800 which requires 24K for cassette and 32K for diskette. THE COLOR ACCOUNTANT also comes with 40 pages of documentation.

the programmer's program

Do you know all the innovative ways of using BASIC commands? The staff at the Programmer's Institute have designed a set of programs explaining everything the manuals omit.
The Programmer's Program leads you step by step through the fundamentals of programming your computer. Then you explore the finer points...**multiple loops**, **queues**, **stacks**, **game programming**, **sorts**, **trees**, **boolean logic**, and **memory savers.** All these methods make your programming tasks easier and more efficient. Difficult (and very useful) programs are created before your eyes, and designed so you learn at your own speed! **The Programmer's Program is a must for every owner of a microcomputer. ($50 cassette, $55 diskette).**

Order both The Programmer's Program and 1 year's subscription, and the total cost is only $75 cassette, $100 diskette. If you order now, you will receive a Football Program and Computerized Telephone Directory ABSOLUTELY FREE with any $75 order. So don't wait . . . order today!

THE PROGRAMMER'S INSTITUTE

A Futurehouse Company
P.O. Box 3191 Dept. C
Chapel Hill, N.C. 27514
(919) 489-2198 MC & Visa Welcome

Name ____________
Address ____________
City/St. ____________
☐ Programmer's Program ☐ Year Subscription
☐ Color Accountant ☐ ½ Year Subscription
Type of Computer: ☐ Trial Issue
____________ ☐ Cassette ☐ Diskette

ATARI, TRS-80, and APPLE are trademarks of the Atari, Tandy, and Apple Corp. The Programmer's Program, TRC, Appletree, Magatari, and The Color Accountant are trademarks of the Programmer's Institute.

*All software available on cassette for ATARI 400/800, APPLE II+, MODEL I, III, COLOR. On diskette for ATARI 400/800, APPLE II+, MODEL III.

9600-9680. The string variable B$ will be our "blanking" string, so we fill it with ATASCII values of zero at line 9690. If you have a player which is no longer than 30 bytes, adjust the DIMension of B$ accordingly.

To produce vertical motion of a player, the subroutines at 100-133 write over the active part of the existing P$ with blanks from B$. Then D$ is written into P$ at the new position, higher or lower than before.

Vertical motion of a missile is slightly more difficult. The problem is that all four missiles are stored in the same memory. Each missile occupies a two-bit slice of the eight-bit bytes in this memory. Thus, we cannot simply write whole new bytes or blanks into this memory.

Instead, using a machine language routine, we AND the existing memory with a binary mask, such as 11111100, leaving zeroes in the appropriate two-bit slice. To this we *add* the new image from DMx$ or B$. The images being added have zeroes outside the two-bit slice, so they won't affect the images of the other missiles.

Lines 9700-9730 read the machine language routine into the string MOVE$. The missile motion subroutines at 150-185 make two USR calls to this routine, once to write the new missile image and once to blank out the rest of the old image. The last variable in the USR call is the decimal equivalent of the binary mask.

Program 2 presents a demonstration of the use of the utility and of the motion routines. The demo attempts to duplicate Watson's animation program. The top part of the listing shows the answers you should give to the questions presented by the utility. The bottom part shows the lines to be added to the skeleton. Lines 300-530 match Watson's line numbers as closely as possible. A comparison of this demo with Watson's program shows that the motion here is somewhat faster – listen to the rate of the "footsteps."

There you have it – a useful utility, a thought-provoker on self-modifying programs, and a neat way to move those spaceships!

Mr. Grace has offered to save you the work of keying in this lengthy program. Just send a stamped, self-addressed mailer, $3 and a blank cassette to 33 Dana Lane, Colts Neck, NJ 07722.

Program 1.

```
1 DIM P0$(1),P1$(1),P2$(1),P3$(1),M$(1)
2 VTAB=PEEK(134)+256*PEEK(135):ATAB=PEEK
(140)+256*PEEK(141)
3 GRAPHICS 17:POSITION 2,3:? #6;"A SELF-
MODIFYING":POSITION 3,6:? #6;"PLAYER-MIS
SILE":POSITION 2,9
4 ? #6;"GRAPHICS UTILITY":POSITION 6,16:
? #6;"ken grace":FOR T=1 TO 3000:NEXT T
5 GRAPHICS 0:? :? "THIS UTILITY PRESENTS
 A SERIES OF     QUESTIONS BY WHICH YOU
DEFINE A        PLAYER-MISSILE GRAPHICS";
6 ? " SETUP.":? :? "IT THEN MODIFIES ITS
ELF INTO A PROGRAMSKELETON.":? :? "SUBRO
UTINES FOR PLAYER AND MISSILE "
7 ? "MOTION ARE INCLUDED.":? :? "YOU ADD
 THE REST OF THE PROGRAM.":? :? "ANIMATI
ON IS POSSIBLE BY COPYING NEW "
8 ? "SHAPE STRINGS INTO THE STRINGS":? "
DEFINING THE PLAYERS.":? :? :? "PRESS |S
TART| TO BEGIN."
9 X=PEEK(53279):IF X<>6 THEN 9
10 ? "{CLEAR}ENTER THE |BASIC| GRAPHICS
MODE FOR PLAYFIELD":? "GR. ";:INPUT X:GO
SUB 90:? "9000 GRAPHICS ";X:GOSUB 91
11 ? "DO YOU WANT STANDARD COLORS FOR ":
? "GRAPHICS ";X;:DIM X$(3):INPUT X$:IF X
$(1,1)="Y" THEN 13
12 ? :? "USE LINES 9800-9850 TO ENTER |S
ETCOLOR| STATEMENTS.  TYPE |CONT| WHEN Y
OU ARE   READY TO CONTINUE.":STOP
13 ? CHR$(125):? :? "RESOLUTION DESIRED
FOR PLAYERS:":? "0 = DOUBLE-LINE":? "1 =
 SINGLE-LINE (FINER)":INPUT R
14 Y=INT(X/16):X=X-16*Y:IF X<=4 THEN S=8
*(1+R)
15 IF X=5 THEN S=12+4*R
16 IF X=6 THEN S=16+8*R
17 IF X=7 THEN S=24+8*R
18 IF X=8 THEN S=36+4*R
19 GOSUB 90:? "9010 RES=";R;":S=";S:GOSU
B 91
20 ? "NUMBER OF PLAYERS TO BE DEFINED";:
INPUT NP
21 FOR I=0 TO NP-1:? CHR$(125):? :? "COL
OR (0 - 15) AND INTENSITY (0 - 15) FOR P
LAYER ";I;:INPUT X,Y
22 GOSUB 90:? 9050+I;" POKE ";704+I;",";
16*X+Y:GOSUB 91
23 ? "WIDTH OF PLAYER ";I;":":? "0 = NOR
MAL":? "1 = TWICE NORMAL":? "3 = FOUR TI
MES NORMAL":INPUT X
24 GOSUB 90:? 9060+I;" POKE ";53256+I;",
";X:GOSUB 91
30 ? "INITIAL HORIZONTAL POSITION (0 - 2
55) FOR LEFT EDGE OF PLAYER ";I;" (40 TO
 2150N SCREEN)";:INPUT X
31 GOSUB 90:? 9070+I;" POKE ";53248+I;",
";X;":XP";I;"=";X;":REM XP";I;" IS HORIZ
 POS OF PLAYER ",I:GOSUB 91
32 ? "VERTICAL LENGTH (BYTES) OF PLAYER
";I;:INPUT X:? CHR$(125):?
```

```
33 ? "INITIAL VERTICAL POSITION OF TOP O
F   PLAYER  (1 TO ";128*(1+R)-X+1;")";:I
NPUT Y:GOSUB 90
34 ? 9080+50*I;"YP";I;"=";Y;":LP";I;"=";
X;":REM VERTICAL POSITION AND LENGTH OF
PLAYER ";I:GOSUB 91
35 ? "USE LINE ";9100+50*I;" (TO ";9129+
50*I;") TO ENTER DATA STATEMENTS WITH TH
E ";X;" BYTES DEFINING  PLAYER ";I
36 ? :? "TYPE |CONT| TO WHEN FINISHED.":
STOP
40 NEXT I:? CHR$(125):S=0
41 ? :? "HOW MANY MISSILES TO BE DEFINED
 (0 TO 4)";:INPUT NM:IF NM=0 THEN ? CHR$
(125):GOTO 52
42 FOR I=0 TO NM-1:? CHR$(125):? :? "WID
TH OF MISSILE ";I:? "0 = NORMAL":? "1 =
TWICE NORMAL"
43 ? "3 = FOUR TIMES NORMAL":INPUT X:S=I
NT(4^I+0.1)*X+S:GOSUB 90:? "9064 POKE 53
260,";S:GOSUB 91
44 ? "INITIAL HORIZONTAL POSITION OF MIS
SILE";I;:INPUT X:GOSUB 90:? 9074+I;" POK
E ";53252+I;",";X;":XM";I;"=";X
45 GOSUB 91:? "VERTICAL LENGTH (BYTES) O
F MISSILE ";I:INPUT X:? CHR$(125)
46 ? :? "INITIAL VERTICAL POSITION OF TO
P OF   MISSILE (1 TO ";128*(1+R)-X+1;")"
:INPUT Y
47 GOSUB 90:? 9280+50*I;" YM";I;"=";Y;":
LM";I;"=";X;":REM VERTICAL POSITION AND
LENGTH OF MISSILE";I:GOSUB 91
48 ? "USE LINE ";9300+50*I;" (TO ";9329+
50*I;") TO ENTER DATA STATEMENTS WITH TH
E ";X;" 'BYTES' DEFINING"
49 ? "MISSILE ";I:X=INT(4^I+0.1):? :? "A
LLOWED VALUES ARE 0, ";X;", ";2*X;", OR
";3*X:?
50 ? "ENTER |CONT| WHEN FINISHED.":STOP

51 NEXT I:? CHR$(125):?
52 ? "PRIORITY SCHEDULE :":? :? "1 - PLA
YERS 0-3,PLAYFLDS 0-3,BACKGND":? :? "2 -
 PLAYERS 0-1,PLAYFLDS 0-3,PLAYERS"
53 ? "    2-3,BACKGND":? :? "4 - PLAYFLDS
 0-3,PLAYERS 0-3,BACKGND":? :? "8 - PLAY
FLDS 0-1,PLAYERS 0-3,PLAYFLDS"
54 ? "    2-3,BACKGND":? :? "ALSO, THE NU
MERICAL SUMS OF THE ABOVE CHOICES ARE AL
LOWED, GIVING BLACK FOR OVERLAPS."
55 ? :? "ABOVE +32 GIVES COLOR IN OVERLA
PS":? :? "CHOICE";:INPUT X:GOSUB 90:? "9
045 POKE 623,";X:GOSUB 91
56 ? :? "PLEASE WAIT FOR PROCESSING.":?
```

```
57 FOR I=3 TO 9:? I:NEXT I:GOSUB 91:GOSU
B 90:FOR I=10 TO 24:? I:NEXT I:GOSUB 91
58 GOSUB 90:FOR I=30 TO 45:? I:NEXT I:GO
SUB 91
59 GOSUB 90:FOR I=46 TO 50:? I:NEXT I:GO
SUB 91
60 IF NM<4 THEN FOR I=NM TO 3:GOSUB 90:F
OR J=0 TO 5:? 150+J+10*I:NEXT J:GOSUB 91
:NEXT I
61 IF NM<4 THEN GOSUB 90:FOR I=NM TO 3:?
 9285+50*I:? 9290+50*I:NEXT I:GOSUB 91
62 IF NM<4 THEN GOSUB 90:FOR I=NM TO 3:?
 9650+10*I:NEXT I:GOSUB 91
63 IF NM=0 THEN GOSUB 90:FOR I=0 TO 7:?
9700+10*I:NEXT I:GOSUB 91
64 IF NP<4 THEN FOR I=NP TO 3:GOSUB 90:F
OR J=0 TO 6:? 100+10*I+J:NEXT J:GOSUB 91
:NEXT I
65 IF NP<4 THEN GOSUB 90:FOR I=NP TO 3:?
 9085+50*I:? 9090+50*I:? 9600+10*I:NEXT
I:GOSUB 91
67 GOSUB 90:FOR I=51 TO 65:? I:NEXT I:GO
SUB 91
68 ? :? :? "67":? "68":? "90":? "91":? "
92":? "POKE 842,12:? CHR$(125)":POSITION
 0,0:POKE 842,13:STOP
90 SETCOLOR 1,9,4:? CHR$(125):? :RETURN
91 ? :? :? "CONT":POSITION 0,0:POKE 842,
13:STOP
92 POKE 842,12:? CHR$(125):? :SETCOLOR 1
,9,10:RETURN
99 GOTO 9000
100 REM MOTION OF PLAYER 0. XP0 AND YP0
ARE X,Y POSITIONS. DX0 AND DY0 ARE CHANG
ES.
101 TRAP 106:IF DY0<>0 THEN 103
102 POKE 53248,XP0+DX0:GOTO 105
103 IF DY0<0 THEN P0$(YP0+DY0,YP0+LP0+DY
0-1)=D0$:P0$(YP0+LP0+DY0,YP0+LP0-1)=B$(1
,-DY0):GOTO 105
104 P0$(YP0,YP0+DY0-1)=B$(1,DY0):P0$(YP0
+DY0,YP0+LP0+DY0-1)=D0$
105 XP0=XP0+DX0:YP0=YP0+DY0:DX0=0:DY0=0:
RETURN
106 DX0=0:DY0=0:POKE 53248,XP0:P0$(YP0,Y
P0+LP0-1)=D0$:RETURN
110 REM MOTION OF PLAYER 1.
111 TRAP 116:IF DY1<>0 THEN 113
112 POKE 53249,XP1+DX1:GOTO 115
113 IF DY1<0 THEN P1$(YP1+DY1,YP1+LP1+DY
1-1)=D1$:P1$(YP1+LP1+DY1,YP1+LP1-1)=B$(1
,-DY1):GOTO 115
114 P1$(YP1,YP1+DY1-1)=B$(1,DY1):P1$(YP1
+DY1,YP1+LP1+DY1-1)=D1$
115 XP1=XP1+DX1:YP1=YP1+DY1:DX1=0:DY1=0:
RETURN
```

```
:? "WHEN YOU SEE !READY! YOU MAY LIST OR
 RUN":FOR X=1 TO 900:NEXT X:GOSUB 90
116 DX1=0:DY1=0:POKE 53249,XP1:P1$(YP1,Y
P1+LP1-1)=D1$:RETURN
120 REM MOTION OF PLAYER 2.
121 TRAP 126:IF DY2<>0 THEN 123
122 POKE 52350,XP2+DX2:GOTO 125
123 IF DY2<0 THEN P2$(YP2+DY2,YP2+LP2+DY
2-1)=D2$:P2$(YP2+LP2+DY2,YP2+LP2-1)=B$(1
,-DY2):GOTO 125
124 P2$(YP2,YP2+DY2-1)=B$(1,DY2):P2$(YP2
+DY2,YP2+LP2+DY2-1)=D2$
125 XP2=XP2+DX2:YP2=YP2+DY2:DX2=0:DY2=0:
RETURN
126 DX2=0:DY2=0:POKE 53250,XP2:P2$(YP2,Y
P2+LP2-1)=D2$:RETURN
130 REM MOTION OF PLAYER 3.
131 TRAP 136:IF DY3<>0 THEN 133
132 POKE 53251,XP3+DX3:GOTO 135
133 IF DY3<0 THEN P3$(YP3+DY3,YP3+LP3+DY
3-1)=D3$:P3$(YP3+LP3+DY3,YP3+LP3-1)=B$(1
3-1)=D3$:P3$(YP3+LP3+DY3,YP3+LP3-1)=B$(1
,-DY3):GOTO 135
134 P3$(YP3,YP3+DY3-1)=B$(1,DY3):P3$(YP3
+DY3,YP3+LP3+DY3-1)=D3$
135 XP3=XP3+DX3:YP3=YP3+DY3:DX3=0:DY3=0:
RETURN
136 DX3=0:DY3=0:POKE 53251,XP3:P3$(YP3,Y
P3+LP3-1)=D3$:RETURN
150 REM MOTION OF MISSILE 0. XM0,YM0  AR
E X,Y COORDS. DXM0,DYM0 ARE CHANGES
151 TRAP 155:IF YM0+DYM0<1 OR YM0+DYM0+L
M0>128*(1+RES) OR DYM0=0 THEN DYM0=0:GOT
O 154
152 IF DYM0>0 THEN Z=USR(MOVE,M+YM0+DYM0
,DM0,LM0,252):Z=USR(MOVE,M+YM0,B,DYM0,25
2):GOTO 154
153 Z=USR(MOVE,M+YM0+DYM0,DM0,LM0,252):Z
=USR(MOVE,M+YM0+LM0+DYM0,B,-DYM0,252)
154 YM0=YM0+DYM0:POKE 53252,XM0+DXM0:XM0
=XM0+DXM0:DXM0=0:DYM0=0:RETURN
155 POKE 53252,XM0:DXM0=0:DYM0=0:RETURN
160 REM MOTION OF MISSILE 1
161 TRAP 165:IF YM1+DYM1<1 OR YM1+DYM1+L
M1>128*(1+RES) OR DYM1=0 THEN DYM1=0
162 IF DYM1>0 THEN Z=USR(MOVE,M+YM1+DYM1
,DM1,LM1,243):Z=USR(MOVE,M+YM1,B,DYM1,24
3):GOTO 164
163 Z=USR(MOVE,M+YM1+DYM1,DM1,LM1,243):Z
=USR(MOVE,M+YM1+LM1+DYM1,B,-DYM1,243)
164 YM1=YM1+DYM1:POKE 53253,XM1+DXM1:XM1
=XM1+DXM1:DXM1=0:DYM1=0:RETURN
165 POKE 53253,XM1:DXM1=0:DYM1=0:RETURN
170 REM MOTION OF MISSILE 2
171 TRAP 174:IF YM2+DYM2<1 OR YM2+DYM2+L
M2>128*(1+RES) OR DYM2=0 THEN DYM2=0:GOT
O 174
172 IF DYM2>0 THEN Z=USR(MOVE,M+YM2+DYM2
,DM2,LM2,207):Z=USR(MOVE,M+YM2,B,DYM2,20
7):GOTO 174
173 Z=USR(MOVE,M+YM2+DYM2,DM2,LM2,207):Z
=USR(MOVE,M+YM2+LM2+DYM2,B,-DYM2,207)
174 YM2=YM2+DYM2:POKE 53254,XM2+DXM2:XM2
=XM2+DXM2:DXM2=0:DYM2=0:RETURN
175 POKE 53254,XM2:DXM2=0:DYM2=0:RETURN
180 REM MOTION OF MISSILE 3
181 TRAP 184:IF YM3+DYM3<1 OR YM3+DYM3+L
M3>128*(1+RES) OR DYM3=0 THEN DYM3=0:GOT
O 184
182 IF DYM3>0 THEN Z=USR(MOVE,M+YM3+DYM3
,DM3,LM3,63):Z=USR(MOVE,M+YM3,B,DYM3,63)
:GOTO 184
183 Z=USR(MOVE,M+YM3+DYM3,DM3,LM3,63):Z=
USR(MOVE,M+YM3+LM3+DYM3,B,-DYM3,63)
184 YM3=YM3+DYM3:POKE 53255,XM3+DXM3:XM3
=XM3+DXM3:DXM3=0:DYM3=0:RETURN
185 POKE 53255,XM3:DXM3=0:DYM3=0:RETURN
200 GOTO 200
9015 POKE 559,46+16*RES
9020 PMBASE=PEEK(106)-S:POKE 54279,PMBAS
E:PMBASE=PMBASE*256
9030 POKE 53277,3
9085 DIM D0$(LP0)
9090 RESTORE 9100:FOR I=1 TO LP0:READ BY
TE:D0$(I,I)=CHR$(BYTE):NEXT I
9135 DIM D1$(LP1)
9140 RESTORE 9150:FOR I=1 TO LP1:READ BY
TE:D1$(I,I)=CHR$(BYTE):NEXT I
9185 DIM D2$(LP2)
9190 RESTORE 9200:FOR I=1 TO LP2:READ BY
TE:D2$(I,I)=CHR$(BYTE):NEXT I
9235 DIM D3$(LP3)
9240 RESTORE 9250:FOR I=1 TO LP3:READ BY
TE:D3$(I,I)=CHR$(BYTE):NEXT I
9285 DIM DM0$(LM0)
9290 RESTORE 9300:FOR I=1 TO LM0:READ BY
TE:DM0$(I,I)=CHR$(BYTE):NEXT I:DM0=ADR(D
M0$)
9335 DIM DM1$(LM1)
9340 RESTORE 9350:FOR I=1 TO LM1:READ BY
TE:DM1$(I,I)=CHR$(BYTE):NEXT I:DM1=ADR(D
M1$)
9385 DIM DM2$(LM2)
9390 RESTORE 9400:FOR I=1 TO LM2:READ BY
TE:DM2$(I,I)=CHR$(BYTE):NEXT I:DM2=ADR(D
M2$)
9435 DIM DM3$(LM3)
9440 RESTORE 9450:FOR I=1 TO LM3:READ BY
TE:DM3$(I,I)=CHR$(BYTE):NEXT I:DM3=ADR(D
M3$)
```

```
9490 FOR I=PMBASE+384*(1+RES) TO PMBASE+
1024*(1+RES):POKE I,0:NEXT I
9500 OFFSET=PMBASE+512*(1+RES)-ATAB
9510 FOR I=0 TO 4
9520 V3=INT(OFFSET/256):V2=OFFSET-256*V3

9530 POKE VTAB+2,V2:POKE VTAB+3,V3
9540 POKE VTAB+4,128*(1-RES):POKE VTAB+5
,RES
9550 POKE VTAB+6,128*(1-RES):POKE VTAB+7
,RES
9560 VTAB=VTAB+8:OFFSET=OFFSET+128*(1+RE
S)
9570 IF I=3 THEN OFFSET=PMBASE+384*(1+RE
S)-ATAB
9580 NEXT I
9600 P0$(YP0,YP0+LP0-1)=D0$
9610 P1$(YP1,YP1+LP1-1)=D1$
9620 P2$(YP2,YP2+LP2-1)=D2$
9630 P3$(YP3,YP3+LP3-1)=D3$
9650 M$(YM0,YM0+LM0-1)=DM0$
9660 FOR I=1 TO LM1:J=YM1+I-1:M$(J,J)=CH
R$(ASC(M$(J,J))+ASC(DM1$(I,I))):NEXT I
9670 FOR I=1 TO LM2:J=YM2+I-1:M$(J,J)=CH
R$(ASC(M$(J,J))+ASC(DM2$(I,I))):NEXT I
9680 FOR I=1 TO LM3:J=YM3+I-1:M$(J,J)=CH
R$(ASC(M$(J,J))+ASC(DM3$(I,I))):NEXT I
9690 DIM B$(30):FOR I=1 TO 30:B$(I,I)=CH
R$(0):NEXT I:B=ADR(B$)
9700 DIM MOVE$(38):MOVE=ADR(MOVE$):M=ADR
(M$)-1
9705 RESTORE 9720
9710 FOR I=1 TO 37:READ BYTE:MOVE$(I,I)=
CHR$(BYTE):NEXT I
9720 DATA 104,104,133,204,104,133,203,10
4,133,206,104,133,205,104,104,133,207,10
4,104,133,208
9730 DATA 160,0,177,203,37,208,113,205,1
45,203,200,196,207,208,243,96
9999 GOTO 200
```

Program 2. Animation Demo

RUN the utility in Listing 1 and give the following answers:

Graphics Mode: 18
Default Colors: NO

9010 SETCOLOR 4,7,2
CONT

Resolution: 0
Number of Players: 1
Color, Intensity: 1,6
Width: 0
Horizontal Position: 127
Length: 9
Vertical Position: 63

9100 DATA 126,90,66,60,219,189,102,102,231

CONT
Number of Missiles: 0
Priority: 0

Then add the following lines to the skeleton program:

```
200 DIM D01$(9),D02$(9),D03$(9)
210 D01$=D0$
220 RESTORE 520
230 FOR I=1 TO 9:READ BYTE:D02$(I,I)=CHR
$(BYTE):NEXT I
240 FOR I=1 TO 9:READ BYTE:D03$(I,I)=CHR
$(BYTE):NEXT I
300 REM *** VIEW POINTER & STRING ***
310 C=C+1
320 IF C>4 THEN C=1
330 ON C GOTO 340,350,340,360
340 D0$=D01$:GOTO 370
350 D0$=D02$:GOTO 370
360 D0$=D03$
370 P0$(YP0,YP0+LP0-1)=D0$
380 FOR I=1 TO 9
385 IF C=2 OR C=4 THEN SOUND 0,28*I,6,9-
I
390 NEXT I
400 REM *** MOTION ROUTINE ***
410 A=STICK(0)
420 IF A=11 THEN XP0=XP0-1:POKE 53248,XP
0
430 IF A=11 THEN XP0=XP0-1:POKE 53248,XP
0
440 IF A=7 THEN XP0=XP0+1:POKE 53248,XP0

450 IF A=13 THEN DY0=1:GOSUB 100
460 IF A=14 THEN DY0=-1:GOSUB 100
470 GOTO 310
520 DATA 126,90,66,60,219,189,102,230,7
530 DATA 126,90,66,60,219,189,102,103,22
4
```

©

VIC/PET BASIC Program Transfers

Jim Butterfield
Toronto

BASIC programs flow easily from the PET to the VIC. VIC has a relocating feature that repacks the incoming program into whatever space that VIC has available. That's a necessary trick: BASIC may start in any of three different locations within the VIC depending on what extra memory has been plugged in.

It's a little more difficult to transfer programs from the VIC to the PET. The stable PET/CBM system expects a BASIC program to be set up for one specific start location. The PET isn't equipped to handle this flighty VIC format which seems to hop around from place to place.

So for VIC-to-PET transitions we need to cope with a "memory mapping" problem. There are several ways to approach this; some of my favorites are itemized below.

Keep in mind that we're discussing BASIC programs only. Machine Language may need different and special handling. And don't forget: PEEKs, POKEs, WAITs, USRs, and SYS's will probably need coding changes to work successfully in the new environment.

A Few Methods:

1. LOAD the program into the VIC with the 3K expansion module in place. SAVE the program. The program will load correctly into the PET with no further modifications needed.

Note that the VIC must have the 3K expansion and no other. The 8K expansion job won't work.

2. Before loading the VIC program into the PET, type NEW; then FOR J = 0 TO 2:POKE 4096 + J,0:POKE 4608 + J,0:NEXT J.

Load the program into the PET. Type LIST. If the program isn't there, type POKE 41,16. Type LIST. If the program still isn't there, type POKE 41,18.

Don't forget to reset your PET when you've finished playing with the VIC program.

Each of the above combinations corresponds to a VIC configuration at the time the program was saved. If 3K expansion was in place, the first LIST will work without any POKE (see method 1). If no memory expansion was there, POKE 41,16 will do the trick. If 8K or more was there, POKE 41,18 is the magic combination.

3. There's another technique available called a Merge. Space is insufficient to outline the whole story here. Suffice it to say that you can use a complex piece of magic to perform a cassette tape Merge; you can use special software for a disk Merge (for example, Power has this feature) or you can use a machine language program specially written to do the trick.

The capability of merging BASIC lines together is quite important: its usefulness extends far beyond the simple objective of transferring programs between machines. [*See "BASIC Program Merges" in this issue.*] ©

Part I of this article appeared in the March, 1982, issue and Part II appeared last month. This completes the routines which comprise the Supermonitor for the Ohio Scientific Superboard Computer.

Supermonitor: Part III

Frank Cohen
Pacific Palisades, CA

Here is the conclusion to a long and complex program which adds functions to a Superboard II which you would normally find only in an advanced operating system. These functions make it easy to display, move, and modify machine language programs and data.

The programs listed so far have made up the framework of the Supermonitor. The first program presented was called Hexdump and did nothing more than dump an address and eight bytes of data onto the screen. Hexdump was listed first because subsequent programs use some of its subroutines. The second article included two programs. Indata prints a line of eight bytes of data and allows you to modify the contents. After you have modified a byte, Indata allows you to move forward, backward, or skip over subsequent memory locations. Bmove is a simple block move program. Bmove moves a whole block of memory to another location in memory. With just these three programs, entering and editing maching language data is much more efficient and easy than using the ROM Monitor program OSI supplies.

Without a disk system, loading Supermonitor in its entirety takes about five minutes with the Superboard's 300 baud cassette interface. With the assembly listings of Supermonitor you can use only the programs you find interesting. By doing this, you can limit the size of Supermonitor. The listing of the main menu program shows all the equates for all the programs.

All of the programs of Supermonitor use a program called Supercursor V1.3 (**COMPUTE!**, December, 1981, #19, p. 124) to handle its video output. Supermonitor is installed directly below Supercursor at the top of an 8K byte Superboard II. If you don't want to use Supercursor, you can write your own video output routines. To use Supercursor V1.3 a program puts the ASCII character in the CPU's accumulator and executes a JSR to its start address, located at $1E80. Supercursor also has routines to "Home" the cursor and clear the screen. To use the Home functions, jump to the subroutine at $1E80 using a JSR. Use the same instruction to clear the screen at $1EC2.

A Brief Review

Let's go over some terms. An *assembler* is nothing more than a program which takes programs called *source code* and converts them into machine instructions (called *object code*) which can be directly executed by your computer. Assembly language is made up of three-letter codes which abbreviate what the CPU [*Central Processing Unit*] executes. For example, one commonly used instruction is the "load the accumulator" instruction. In machine language, the code is an A9 followed by the byte to be loaded into the accumulator. In assembly language, the instruction looks something like this: LDA. This stands for LoaD Accumulator. But load it with what?

The 6502 microprocessor has twelve different addressing modes. So, following the LDA instruction, the assembler looks for the type of addressing to use. One of the most common is the *immediate mode*. To load the accumulator with the value 00 (hex) the assembler instruction looks like this: LDA #$00. The pounds sign (#) stands for immediate addressing and the dollar sign ($) tells the assembler that this is a hexadecimal number. If you left out the pounds sign, the assembler would think that you want to load the accumulator with a byte residing at location $00 in the zero page of memory. Executing an instruction like LDA $1000 tells the assembler to load the accumulator with the byte at location $1000 in memory. Labels may be used instead of the actual numbers.

These labels are called *equates*. Before entering the program into the assembler labels can be defined. By defining the labels, specific numbers are assigned to alphanumeric names. In the listing of the main menu program, the major equates are shown. For example, the equate named *cursor* is assigned the value $1E40. So, when we tell the assembler to jump to a subroutine called *cursor* (JSR CURSOR) the assembler will execute the subroutine starting at $1E40. Using equates, assembly language becomes easier to read.

Main Menu

This is by far the simplest of the programs. By entering at $1A7B (called SPMON) the program first clears the screen, then homes the cursor and reads the keyboard. When a key is pressed, it checks to see if it is a valid character. If it is, we jump to the correct program. If not, the screen is cleared and we return to the beginning of the program. The valid characters are listed below:

G – EXECUT, transfers control to a machine language program

I – INDATA, displays and modifies memory

QUALITY SOFTWARE FOR TRS-80 COLOR AND OSI

ADVENTURES!!!

For TRS-80 COLOR and OSI. These Adventures are written in BASIC, are full featured, fast action, full plotted adventures that take 30-50 hours to play. (Adventures are inter-active fantasies. It's like reading a book except that you are the main character as you give the computer commands like "Look in the Coffin" and "Light the torch.")

Adventures require 16K on COLOR-80 and TRS-80. They sell for $14.95 each.

ESCAPE FROM MARS
(by Rodger Olsen)

This ADVENTURE takes place on the RED PLANET. You'll have to explore a Martian city and deal with possibly hostile aliens to survive this one. A good first adventure.

PYRAMID (by Rodger Olsen)

This is our most challenging ADVENTURE. It is a treasure hunt in a pyramid full of problems. Exciting and tough!

TREK ADVENTURE (by Bob Retelle)

This one takes place aboard a familiar starship. The crew has left for good reasons – but they forgot to take you, and now you are in deep trouble.

NEW!!

CIRCLE WORLD – We got Kzinti and puppeteers and problems. Our newest and biggest adventure. Requires 12k on OSI and 16K on TRS-80 Color.

NUCLEAR SUB – You are trapped in a nuclear sub at the bottom of the ocean. Escape and even survival is in doubt. Plotted by three of the most sadistic – I mean "creative" minds in adventure programming.

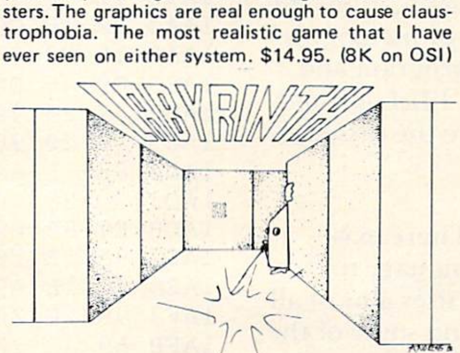
VENTURER!—A fast action all machine code Arcade game that feels like an adventure. Go berserk as you sneak past the DREADED HALL MONSTERS to gather treasure in room after room, killing the NASTIES as you go. Great color, high res graphics, sound and Joystick game for the TRS-80 Color or OSI machines. (black and white and silent on OSI.) Tape only. $19.95.

BASIC THAT ZOOOMMS!!
AT LAST AN AFFORDABLE COMPILER FOR OSI AND TRS-80 COLOR MACHINES!!! The compiler allows you to write your programs in easy BASIC and then automatically generates a machine code equivalent that runs 50 to 150 times faster.

It does have some limitations. It takes at least 8K of RAM to run the compiler and it does only support a subset of BASIC—about 20 commands including FOR, NEXT, END, GOSUB, GOTO, RETURN, END, PRINT, STOP, USR(X), PEEK, POKE, *, / , + , - , X , X , = , VARIABLE NAMES A-Z, A SUBSCRIPTED VARIABLE, and INTEGER NUMBERS FROM 0 - 64K.

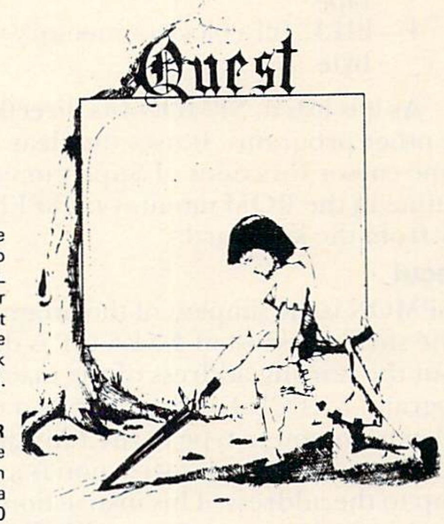
TINY COMPILER is written in BASIC. It generates native, relocatable 6502 or 6809 code. It comes with a 20 page manual and can be modified or augmented by the user. $24.95 on tape or disk for OSI or TRS-80 Color.

LABYRINTH – 16K EXTENDED COLOR BASIC – With amazing 3D graphics, you fight your way through a maze facing real time monsters. The graphics are real enough to cause claustrophobia. The most realistic game that I have ever seen on either system. $14.95. (8K on OSI)



PROGRAMMERS!

SEE YOUR PROGRAM IN THIS SPACE!! Aardvark traditionally pays the highest commissions in the industry and gives programs the widest possible coverage. Quality is the keyword. If your program is good and you want it presented by the best, send it to Aardvark.



QUEST – A NEW IDEA IN ADVENTURE GAMES! Different from all the others. Quest is played on a computer generated map of Alesia. Your job is to gather men and supplies by combat, bargaining, exploration of ruins and temples and outright banditry. When your force is strong enough, you attack the Citadel of Moorlock in a life or death battle to the finish. Playable in 2 to 5 hours, this one is different every time. 16K COLOR-80 OR TRS-80. ONLY $14.95.

SPACE ZAPPER – Protect your central Star Base from ships that attack from all four sides. Fast reflexes are required as the action speeds up. Great for kids or Dads. This game has high speed high resolution graphics and looks as if it just stepped out of the arcades. – 16K extended or 32K disk. BASIC TRS-80. Color only. $14.95.

Please specify system on all orders

This is only a partial listing of what we have to offer. We have arcade and thinking games, utilities and business programs for the OS1 and TRS-80 Color. We add new programs every week. Send $1.00 for our complete catalog.

TRS 80 COLOR **AARDVARK - 80**
2352 S. Commerce, Walled Lake, MI 48088
(313) 669-3110 **OSI**

C – CLEAR, clears the screen
D – HXDMP, dumps memory to the screen
M – BMOVE, moves a block of data
S – TAPOUT, saves a block of data to cassette tape
F – FILL, fill a block of memory with a specified byte

As it is listed, SPMON fits directly under all the other programs. It uses the clear screen and home cursor functions of Supercursor and a subroutine in the ROM monitor (at $FFBA) to get a key from the keyboard.

Execut

If SPMON is the simplest of the programs, EXECUT is the smallest. Most of EXECUT is devoted to input the starting address of the machine language program. EXECUT prints "G=" on the screen and expects you to type in the four digit address. An infrequently used instruction is applied to jump to the address. This instruction is called the "jump indirect" instruction. EXECUT uses the INADR subroutine in HXDUMP to input the address to locations $00E7 and $00E8. We then use the jump indirect instruction to use these addresses.

Fill

This program is similar to BMOVE. FILL loads a block of memory with some value you input. It starts by asking the beginning address of memory by printing "S=" on the screen. Type in the four digit hexadecimal address. FILL then asks for the ending address by printing "E=". Again, input the address. Then it asks what the block of memory is to be filled with. FILL is a very fast program and will fill all 64K in about two seconds. FILL is listed to fit after the main menu and before the cassette tape program.

Tapout

This is the most valuable program. There exist programs that save from machine language to tape, but the problem is that BASIC uses almost all of the zero page memory locations and some of the main memory making it difficult to work around. Since the Superboard's ROM monitor already has a tape input routine, this program only stores data onto cassette.

TAPOUT makes use of BASIC's cassette output subroutine stored in ROM. By setting location $0205 to FF (hex) a jump to subroutine instruction outputs the contents of the accumulator to the cassette interface at 300 baud. After TAPOUT is finished, it resets location $0205 to 00. If you want to use TAPOUT from a machine language program, put the starting address at location $00E9 and $00EA and also the ending address at location $00E7 and $00E8. Then execute the program.

After you install the three programs in this issue, it is necessary to make some slight modifications so that all the programs will return control to the main menu program. To do this you will need to enter the following modifications:

```
1C36   4C   7E   1A   ;For BMOVE
1CB0   4C   7E   1A   ;For INDATA
1D1D   4C   7E   1A   ;For HEXDUMP
1D38   F0   E3
```

```
;EQUATES
CURSOR      =$1E40
CLS         =$1EC2
HOME        =$1E80
INADR       =$1D93
CR          =$1E95
LF          =$1EAB
KEYIN       =$FFBA
EXECUT      =$1ABA
FILL        =$1ACA
TAPOUT      =$1B3F
BMOVE       =$1BC6
INDATA      =$1C56
HXDMP       =$1D20
ADR         =$E7
EBAD        =$E9
SBAD        =$EB
TMP         =$ED
CVAHX       =$1DF3
CVHA        =$1D72
OFLAG       =$0205
AOUT        =$FFEE
```

```
1A7B 20 C2 1E 20 80 1E A9 24
1A83 20 40 1E 20 BA FF C9 47
1A8B D0 03 4C BA 1A C9 49 D0
1A93 03 4C 56 1C C9 4C D0 03
1A9B 4C 7B 1A C9 44 D0 03 4C
1AA3 20 1D C9 4D D0 03 4C C6
1AAB 1B C9 53 D0 03 4C 3F 1B
1AB3 C9 46 D0 C4 4C CA 1A A9
1ABB 47 20 40 1E A9 3D 20 40
1AC3 1E 20 96 1D 6C E7 00 20
1ACB 80 1E A9 53 20 40 1E A9
1AD3 3D 20 40 1E 20 96 1D A5
1ADB E7 85 E9 A5 E8 85 EA 20
1AE3 95 1E 20 AB 1E A9 45 20
1AEB 40 1E A9 3D 20 40 1E 20
1AF3 96 1D 20 95 1E 20 AB 1E
1AFB A9 42 20 40 1E A9 3D 20
1B03 40 1E 20 BA FF 20 40 1E
1B0B 20 F3 1D 0A 0A 0A 0A 85
1B13 ED 20 BA FF 20 40 1E 20
1B1B F3 1D 18 65 ED 85 ED A5
1B23 ED A0 00 91 E9 E6 E9 A5
1B2B E9 D0 02 E6 EA A5 E9 C5
1B33 E7 D0 EC A5 EA C5 E8 D0
1B3B E6 4C 7E 1A 20 80 1E A9
1B43 53 20 40 1E A9 3D 20 40
1B4B 1E 20 96 1D A5 E7 85 E9
1B53 A5 E8 85 EA A9 45 20 40
1B5B 1E A9 3D 20 40 1E 20 96
1B63 1D A9 FF 8D 05 02 A9 2E
1B6B 20 EE FF A5 EA 20 A5 1B
1B73 A5 E9 20 A5 1B A9 2F 20
```

©

Machine Language: First Steps, Part II

Jim Butterfield
Toronto

In the last episode, our hero, F. R. Vescent has started to convert the following bar graph program from BASIC to machine language:

```
200 DATA 15,10,30,35,28,28,15,0
210 READ V:IF V=0 GOTO 300
220 J=48:FOR K=1 TO J
230 J=J+1:IF J=58 THEN J=48
240 PRINT CHR$(J);
250 NEXT K
260 PRINT
270 GOTO210
300 END
```

He has coded lines 220 to 260 inclusive to read:

```
        LDX   #$30
        LDY   #$01
YLOOP   INX
        CPX   #$3A
        BNE   SKIP
        LDX   #$30
SKIP    TXA
        JSR   $FFD2
        INY
        CPY   $0300
        BCC   YLOOP
        LDA   #$0D
        JSR   $FFD2
        RTS
```

He puzzles for a moment over the BCC YLOOP; he knows that this is equivalent to Branch-Less-Than; and this will cause the loop to be executed one time too few. After a few moments thought, he changes the LDY #$01 to LDY #$00; that should do the job. Now he's ready to translate the above Assembler code into machine code, which is what the computer really needs.

The coding will be placed in the PET's first cassette buffer, which starts at hexadecimal address 027A; this is noted at the left of the first line. Now, F. R. looks up LDX in his table of Op Codes, and finds that there are about five different ways the instruction can work. The one he wants is Immediate Mode – flagged by the "#" sign in the operand. That translates to hex A2, so he adds to the first line to make:

```
027A A2 30      LDX #$30
```

Counting off the address bytes, he decides that the next address must start at hex 027C. He writes this value at the left of the second line. Continuing the translation, he gets:

```
027A  A2  30              LDX  #$30
027C  A0  00              LDY  #$00
027E  E8           YLOOP  INX
027F  E0  3A              CPX  #$3A
0281  D0  02              BNE  SKIP
0283  A2  30              LDX  #$30
0285  8A           SKIP   TXA
0286  20  D2  FF          JSR  $FFD2
0289  C8                  INY
028A  CC  00  03          CPY  $0300
028D  90  EF              BCC  YLOOP
028F  A9  0D              LDA  #$0D
0291  20  D2  FF          JSR  $FFD2
0294  60                  RTS
```

There are a few coding tricks that F. R. has to keep in mind to do this translation. First, two-byte addresses are coded "backwards" – that is, hex 0300 is coded as 00 03 and hex FFD2 is coded as D2 FF. Secondly, Branches are coded as a relative offset, so that D0 02 may be read as "if not equal, hop over the next two bytes." A relative branch value of hex EF (see BCC YLOOP) causes the program to branch back 17 locations from the start of the following instruction, if the branch condition is satisfied.

The above "hand assembly" can be greatly helped by the use of assemblers or "tiny assemblers" which work out the Op Code values and calculate the branch values.

Now F. R. is ready to put the machine language program into memory. He calls the machine language Monitor with SYS4, and then asks for a memory display from 027a to 0294 with:

```
.M 027A 0294
```

He might get anything. He will go back and type over the memory contents to enter his program. The changed memory map will look like this:

```
.:  027A  A2  30  A0  00  E8  E0  3A  D0
.:  0282  02  A2  30  8A  20  D2  FF  C8
.:  028A  CC  00  03  90  EF  A9  0D  20
.:  0292  D2  FF  60  ..  ..  ..  ..  ..
```

Now that the machine language program is in place, the BASIC program can be written to hook in with it. F. R. returns to BASIC (with .X) and writes:

```
200 DATA 15,10,30,35,28,28,15,0
210 READ V:IF V=0 GOTO 300
220 POKE 768,V
230 SYS 634
270 GOTO210
300 END
```

Line 220 POKEs the desired value into memory where the machine language program will pick it up. Line 230 calls the machine language program. After the ML program is executed, the BASIC program will resume at line 270.

Try entering the program – both machine language and BASIC – and run it. You'll see the extra speed that machine language gives you.

The program is complete. But it isn't in a form that's very suitable for saving, especially if we want to save to tape. We'll conclude the fearless exploits of F. R. Vescent in the next episode. ©

Did you ever wonder just how a computer can "pick up" a phone, call another computer, and then hang up? How you can "call" the operator without dialing zero? When you hear the ring, is it really the bell at the other phone making the sound? This month, Mike clears up these and other telephone mysteries.

Telecommunications

Using The Telephone

Michael E. Day
Chief Engineer
Edge Technology
West Linn, OR

An important part of the data communications link is the phone line itself. Since we don't have any direct control over the telephone system, it can present real problems when trying to use it with a computer.

There are three major operations that a computer must be able to perform in order to place a call. The first is the actual placement of the call. The second is the data transfer, the use of the line. Finally, the third operation is the hang-up procedure.

At first glance, these actions might seem rather simple things, but, for a computer, they can be a nightmare. Until recently, the phone company would not even allow you to attach your computer to the phone line. Several court cases, and a recent change in the attitude of the FCC has changed the situation. Now you are allowed to attach your computer (or any other device) to the phone line. The only restrictions are that the attaching device be certified by the FCC and that the phone company be notified.

How One Computer Dials Another

By looking at how a call is normally placed, we can see what a computer must do to place a call.

When nothing is going on, the phone line is in an *idle* condition. In this state, there is approximately 48 volts DC between the red and green wires of the line. (They are referred to as the *tip* and the *ring* of the phone line. Long ago, the red wire was attached to the tip and the green wire was attached to the first ring of the old switchboard plugs.) The 48 volts is provided by the telephone exchange.

To place a call, the first thing you do is remove the handset from the telephone and place it to your ear to wait for the dial tone. The tone indicates that the exchange has recognized that you want to place a call and that you may now begin dialing. From a computer's standpoint, this is a bit of a problem. Essentially, it must rely on a hardware device to do the actual dial tone detection for it: a *dial tone detector*.

... three major operations that a computer must be able to perform in order to place a call.

There is a little trick we can pull, however. If you have ever observed people placing a call, you will notice that many do not place the handset to their ear and wait for the dial tone before they begin dialing. This is generally because they are unaware of the requirement or have simply developed a routine which bypasses the need to detect the dial tone. This is possible because over 90% of all call requests are answered within two seconds, and 99.9% within five seconds, on most phone exchanges.

So, by simply performing a routine that causes the dialing to begin sometime after two seconds have passed, you can be assured that most of your calls will get through without ever listening for the dial tone. This is easy to do with the computer; a computer is quite good at waiting. We just wait five seconds after simulating picking up the handset and assume that we have the dial tone. This is called *blind dialing*.

Once we have a dial tone (or think we do) the dialing process can begin. On a touch tone style phone, this is done by sending audio tones to the exchange that are coded to represent the numbers we wish to dial. Eight separate tones are used in pairs to represent the various numbers and codes available with the touch tone system. For a computer to dial this way it must have another piece of hardware, a touch tone dialer. However, there is another little trick that can be used to get around this problem, if you are unwilling to pay for the touch tone dialer or you do not have the touch tone system available to you. Use the rotory dial method.

The actual timing of the dialing is critical on a rotary dial style phone. The dialing is accomplished by momentarily "releasing" the phone line and then "retaking" it again (as if you were to quickly hang the phone up and pick it back up again).

For each number to be dialed, there is a corresponding number of pulses of this sort to be generated. If a one is to be dialed, only a single release sequence is performed. For a nine, there would be nine release sequences. A zero is represented by ten release sequences. This is why you can sometimes reach the operator by simply jiggling the handset button on the telephone since you are simulating dialing the zero for operator. With few exceptions, all touch tone systems can handle rotary dial operations.

This means that it is possible to forego the touch tone completely and use a relay that we also use to simulate raising the handset for the dial tone.

Timing is critical with rotary dialing: the time of the pulses, and the time between the pulses, must be carefully watched. The release time must be 61 milliseconds, plus or minus three milliseconds, and the time between pulses must be 39 milliseconds, plus or minus three milliseconds. Additionally, the time between digits dialed must be greater than 600 milliseconds, but less than ten seconds. That is, after you sent the pulses to dial a five, you must wait at least 600 milliseconds before starting the pulse sequence for a two. This waiting time is done with the phone "off hook."

Once the dialing has been completed, we then wait for the *ring-back signal.* This is a signal the exchange sends back to you that indicates that it is ringing the phone that you have called. This signal is not the actual ringing of the other phone. It is generated at the exchange for you to hear at your end. The two "rings" are not synchronized and you should not assume that when you hear a ring signal it is occurring at the same time at the other end.

If, after waiting, the phone is not answered, or you get a busy signal, you hang up your own phone which terminates the call. If the ringing sound does not occur, you hang up and try placing the call again – assuming that you did something wrong.

For a computer, all this could be a bit tricky. It must be able to recognize the ringing signal, the various busy signals, dialing errors, or wrong numbers. However, if we concentrate on the ends, not the means, it all comes down to a single, simple action.

A Greatly Simplified Method

If the phone at the other end is busy, doesn't answer, was incorrectly dialed, or the wrong party is reached, we can simply hang up the phone and try again. So all we really need to know is if the computer at the other end has answered. This is one of the functions of a modem. [*The box that attaches to a computer and handles telephone calls for it.*] If the called computer's modem is in the auto-answer mode, then that modem will answer the phone and send a signal back to your modem saying that it has answered. All we have to do after we have dialed the number is sit and wait for the answer signal. If we don't receive an answer after a specified period of time, *for any reason,* we hang up the phone.

To make things even easier, the answer signal that is sent is the same kind of signal that the modem uses to transmit data. All we have to do is wait for our modem to tell us that it is ready to communicate with the distant computer. This bypasses many kinds of error indicators. We don't care *why* we were unable to complete the call, only that we weren't. Once a signal has been received, the computers can communicate until they are ready to hang up.

Normally, when you are finished with a call you tell the party you are talking with that you are done and hang up. Alternatively, they may tell you that they are going to hang up. Sometimes, for various reasons, either of you may hang up without telling the other. And, of course the phone company may hang up for either or both of you due to a malfunction. This, too, could get to be rather complicated for a computer unless we recognize that the end result of any such hang-ups is that the communications link has simply been broken. This means that we are no longer receiving the carrier from the other modem. Therefore, we just observe the carrier detection signal from the modem and, if it goes out, we know we've lost the connection and that we should hang up the phone.

A Computer Picks Up The Phone

Normally, when someone calls we hear the bell. This signal is created by the exchange by rapidly reversing the 48 volt signal that is present on the phone line when it is in the idle state. This reversal occurs about 20 times a second for several seconds, pauses, and then repeats.

The rate of the reversal can vary from exchange to exchange and may be anywhere between 16 times a second to 60 times a second. A modem can detect this signal, (assuming you have an *auto-answer* modem) and can generate its own signal called the *ring indicator.* Depending on the modem, the ring indicator signal may either echo the line reversals of the phone line, or just be on for the duration of the ringing signal.

Once there is a ring indicator signal, it is up to your computer to recognize that the signal is there and to send a *DTR signal* to your modem to tell it to go ahead and answer the call. (The DTR signal could also be left always on, indicating to the modem

HUNTINGTON COMPUTING

ADVANCED OPERATING SYSTEMS

#9483	Apple-Aids	$44.89
#9482	Mostly Basic/Household	$26.89
#9481	Mostly Basic/Interface-Scientific	$26.89
#9480	Voyage of the Valkyrie	$35.89

ARTSCI

#6061	Basic Mailer	$59.49
#6060	Magic Window	$84.99
#6062	Magic Words	$62.89

AURORA

#480;	Executive Secretary	$212.49
#4803	Gradebook	$51.00
#4804	Hebrew II	$50.99
#4811	Hebrew II	$79.19
#4805	Quicktrace	$43.99

AVALON HILL

#2058	Major League Baseball	$25.49
#2052	North Atlantic Convoy (Cassette)	$12.69
#2060	Empire of the Overmind (Cassette)	$25.49
#2059	Empire of the Overmind	$29.69
#2061	Tanktics	$24.59

BEAGLE BROS.

#3257	Alpha Plot	$34.69
#3250	Dos Boss 3.3	$20.39
#3255	Dos Boss 3.2	$20.39
#3251	Game Pack #1	$20.39
#3252	Game Pack #2	$20.39
#3253	Game Pack #3	$20.39
#3254	Game Pack #4	$20.39
#3258	Tip Disk	$17.99
#3256	Utility City	$25.89

BLACKBURG

#9160	Active Filter Design	$35.89
#9162	Electronics I	$35.89
#9163	Electronics II	$35.89
#9164	Electronics III	$35.89
#9165	Video Display Graphics	$35.89

BRODERBUND

#1958	Alien Rain	$21.19
#1960	Alien Typhoon	$21.19
#1967	Arcade Machine	$39.49
#1966	David's Midnight Magic	$30.69
#1952	Galactic Empire	$22.99
#1951	Galaxy Wars	$22.99
#2000	Genetic Drift	$25.39
#1957	Golden Mountain	$16.89
#1961	Payroll	$335.69
#1949	Red Alert	$22.99
#1965	Space Quarks	$25.39
#1964	Space Warrior	$21.19
#1956	Tank Command	$12.99
#1955	Tawala's Last Redoubt	$25.39
#1959	Track Attack	$26.89

HARDWARE

#8035	Apple Clock (Mountain)	$246.39
#8200	Applesoft Renumber and Merge in ROM (Soft Control)	$33.89
#8206	Applesoft Utility ROM (Soft Control)	$33.89
#747	B S R Control (Novation)	$16.69
#2107	Basic Compiler (Microsoft)	$335.69
#8205	Basic in ROM (Soft Control)	$33.89
#8039	Blitz Bug (Omni)	$21.89
#8025	C P S Pascal (Mountain)	$16.99
#86	Cable Adaptor (Centronics)	$35.19
#24	Character Set Plus (Lazer)	$22.39
#6534	Cassette Control Device (Hartley)	$67.95
#9043	Data Link (Link)	$87.99
#217	DISASM/65 (Lazer)	$26.29
#2362	Dithertizer II with Camera (Computer Station)	$639.99
#8203	Dual DOS ROMS (Soft Control)	$46.69
#7951	E Z Port (Versa)	$21.19
#748	Expansion Mod (Novation)	$33.15
#1476	Game Paddles (TG)	$33.89
#1070	Joyport (Sirius)	$67.39
#12	Joystick (Mimco)	$59.95
#322	Large Character Eprom (Videx)	$34.29
#27	Lower Case Adaptor Rev. 7 (Paymar)	$29.99
#9021	Memory Expansion Module (Prometheus)	$148.69
#8060	Micro-Verter (ATV)	$31.49
#749	Modem Handset (Novation)	$24.65
#6240	Paddle-Adapple (Southern California)	$26.89
#39	560G Paper Tiger (IDS)	$1,225.00
#8020	MX100 Ft. Printer (Epson)	$749.00
#725	Pro Paddle (Rainbow)	$44.89
#35	Ram Card (Microsoft)	$169.00
#16	16K Ram Expansion Board (Andromeda)	$169.00
#325	Soft Switch (Videx)	$25.49
#36	Z-80 Softcard (Microsoft)	$299.00
#860	Super Fan II (R. H. Electronics)	$64.99
#31	Switchplate (Videx)	$17.99
#59	Vision 80 Card (Vista)	$297.49

MICRO LAB

#2253	Crown of Arthain	$29.69
#2250	Dogfight	$25.99
#2254	Mad Venture	$21.19
#2259	Palace in Thunderland	$21.19
#2263	Roach Hotel	$30.69

Softlights

By Fred Huntington

THREE-YEAR OLD ATARI COMPUTER USERS BIRTHDAY SALE

Last year, on my daughter Melody's second birthday, we ran a special on Hodge Podge, the great Apple game for preschoolers. This year, we're very happy because Artworks has come out with a version of Hodge Podge for the Atari. My daughter, Melody, has gone ape over it.

Melody doesn't hover around the Apple at home now, just the Atari, because she wants to play Hodge Podge. Whenever she presses a letter, something happens. For example, if she presses "B" a banjo appears and plays music. There is some action for every key on the keyboard.

The graphics are tremendous and the music is great. There is one major flaw, however. Everytime I get near the Atari, I can't play my games. Melody wants to play Hodge Podge.

So, if you have a child between two and six, get them Hodge Podge and be a hero. And, to help celebrate Melody's third birthday, if you say "Happy Birthday, Melody," you can have it for **$15.94** instead of the $19.95 list price. Even at full list price, I believe Hodge Podge to be one of the best bargains available today. (We'll even give you the Apple version for $15.99.)

IBM SOFTWARE

We're now concentrating all our efforts on carrying software for the Apple, Atari, and IBM computers. New for the IBM PC are The Programmer and Las Vegas Black Jack and Word Star. We'll be having more and more IBM PC software. If it's available, we'll try to carry it.

GRAPHICS PRINTER - UNDER $350

We're carrying a great new line of printers – Axiom. Their smallest printer is a knockout. It uses regular computer paper, does fine graphics, comes complete with card and cable and is being sold by us for only $349 plus shipping.

The only disadvantage we can honestly find is that it is only about half as fast as an Epson. But at that price, who needs speed?

MISCELLANEOUS

#2182	Adventure in Time (Phoenix)	$25.39
#2301	Apple World (USA)	$52.99
#7890	Applecillin (XPS)	$33.89
#3700	Asteroid Field (Cavalier)	$22.49
#2700	Astroscope (AGS)	$26.49
#2180	Birth of the Phoenix (Phoenix)	$35.89
#2407	Bishop's Square (Datasoft)	$25.39
#1601	Bloody Murder (Stoneware)	$17.99
#3000	Bookkeeper (Delta)	$76.39
#3702	Bug Attack (Cavalier)	$26.29
#9700	Castles of Darkness (Logical)	$30.69
#3001	Checkwriter (Delta)	$33.99
#9860	Chinese Lessons (CTI)	$26.09
#7851	Cosmo-Mission (Cosmos)	$21.19
#9241	The Count (Pear)	$22.39
#2752	Creative Financing (Howardsoft)	$127.49
#2205	Creature Venture (Highland)	$21.19
#4401	Cyborg (Sentient)	$28.99
#1607	D B Master Version 3 (Stoneware)	$194.59
#4250	Falcons (Piccadilly)	$25.39
#7878	Financial Partner (Denver)	$220.49
#6080	Firebird (Gebelli)	$25.39
#3954	Form Letter (S. Sorcery)	$26.99
#7852	Hungry Boy (Cosmos)	$21.19
#3451	Interlude (Syntonic)	$16.99
#9840	Jabbertalky (Mind Toys)	$26.29
#3150	The Landlord (Min Micro)	$649.99
#9242	The Liberator (Pear)	$26.89
#2406	Lisp Interpreter (Datasoft)	$106.19
#3953	Mailing List (S. Sorcery)	$44.89
#9240	The Manipulator (Pear)	$31.39
#4552	Market Analyzer (RTR Software)	$224.99
#4550	Market Charter (RTR Software)	$110.39
#9100	Memory Page Editor (Compu-tron)	$44.89
#9580	The Menu (C & H Video)	$26.29
#9380	Menu Generator (Crane)	$35.09
#1804	Monty Plays Monopoly (Visicorp)	$31.99
#2211	Mummy's Curse (Highland)	$26.39
#2200	Oldorf's Revenge (Highland)	$16.89
#4400	OO-Topos (Sentient)	$27.99
#7870	Pascal Programmer (Denver)	$105.99
#7871	Pascal Tutor (Denver)	$105.99
#7650	Pornopoly (CCI)	$25.39
#3484	Pot of Gold I (Rainbow)	$33.89
#3485	Pot of Gold II (Rainbow)	$33.89
#2751	Real Estate Analyzer (Howardsoft)	$127.49
#6600	Rubik's Cube (Software Alternative)	$16.89
#6081	Russki Duck (Gebelli)	$31.39
#1150	Sargon II (Hayden)	$29.79
#2701	Sexoscope (AGS)	$26.49
#8901	Snack Attack (Datamost)	$25.39
#8520	Solitaire (Computek)	$25.39
#2302	Space Raiders (USA)	$25.39
#2306	Star Dance (USA)	$21.89
#3701	Star Thief (Cavalier)	$25.39
#7163	Star Venture (Compu-Thing)	$26.89
#9180	Starship Commander (Voyager)	$35.89
#4251	Suicide (Piccadilly)	$26.29
#3955	Text Editor (S. Sorcery)	$58.39
#8901	Thief (Datamost)	$25.39
#7853	Thunderbirds - GX (Cosmos)	$21.19
#7102	Trickshot (IDSI)	$35.09
#1606	Utility Pak #1 (Stoneware)	$21.19
#4100	Visicaid (Data S. C.)	$29.69
#1820	VisiCalc 3.3 (Visicorp)	$199.00
#1805	VisiFile (Visicorp)	$212.49
#9740	Vocanoes (Earthware)	$43.49
#2181	Zoom Grafix (Phoenix)	$35.89
#7880	Zork I (Infocom)	$33.89
#7881	Zork II (Infocom)	$33.99

SIRIUS

#1058	Autobahn	$25.39
#1068	Beer Run	$19.99
#1073	Borg	$26.89
#1074	Computer Foosball	$25.39
#1065	Copts & Robbers	$29.69
#1051	Cyber Strike	$33.89
#1067	Dark Forest	$19.99
#1060	Gamma Goblins	$25.39
#1061	Gorgon	$33.89
#1069	Hadron	$29.69
#1057	Pulsar II	$25.39
#1071	Snake Byte	$25.39
#1064	Sneakers	$25.39
#1056	Space Eggs	$19.99
#1050	Star Cruiser	$20.99
#1072	Twerps	$26.89

SYNERGISTIC

#1220	Adventure to Atlantis	$35.99
#1216	Data Reporter	$186.99
#1202	Dungeon Campaign	$14.79
#1210	Escape from Arcturus	$25.39
#1214	The Linguist	$33.99
#1200	Odyssey	$25.39
#1207	Program Line Editor	$33.99
#1201	Wilderness Campaign	$18.49

Call Toll-Free **800-344-5106** (outside California)

HUNTINGTON COMPUTING
Post Office Box 1297
Corcoran, California 93212

Foreign Orders 209-992-4481
In California 800-692-4146

Outside Calif. 800-344-5106

We take **MasterCard, American Express or VISA** (Include card # and expiration date). California residents add 6% tax. Include $2.00 for postage. Foreign and hardware extra. Foreign (excluding Canada): remit U.S. currency, checks on U.S. banks, use listed charge cards, or make direct wire transfers through Security Pacific Bank, Corcoran, for a $6.00 charge. All overseas orders shipped by air. Send for free catalog. Prices subject to change without notice.

that the computer will answer any call immediately.)

The modem answers a call by turning on the "off-hook relay" which simulates taking the handset off of the phone hook or cradle. This causes a low impedence load to be placed on the phone line (approximately 600 ohms) and it remains as long as the phone is off-hook. This tells the exchange that you have accepted the call and are ready to answer it. The exchange then begins a supervisory process to stabilize the link and begin any necessary billing.

During all this, the answering modem cannot yet transmit any data; it may disrupt the supervisory process. Most of the new exchanges simply prevent any signals from being transferred during this time, but some of the older exchanges don't have this protection and signals could get disrupted. This could cause anything from the exchange hanging up on you to its billing you for a call you didn't make. This time period of disallowed transmission lasts a maximum of two seconds. Once this time period has passed, the computers can communicate normally. If you later lose the carrier signal, you should assume that the other end has hung up their phone and hang up yours. ©

If you have a SuperPET, this article will show you how to use all 96K from Commodore BASIC.

Run 96K Programs On The SuperPet

Paul Donato
Sudbury, Ontario

The SuperPet has an additional 64K of memory: 16 4K banks addressed at memory location 9000 hex. POKEing a decimal number from 0 to 15 into decimal location 61436 will cause the appropriate bank to be switched in. Full use of this memory is possible when the SuperPet is used in the Commodore BASIC mode. By having a main program in the regular 32K of PET memory one can access up to 16 4K modules or BASIC routines which are preloaded into the additional 4K memory banks. To do this the main program must do basically three things:

1. it must POKE into location 61436 the number of the desired bank.
2. it should then POKE 41,144. This causes the start-of-BASIC pointer to point to 9000 hex.
3. finally, the program should execute a GOTO, directing program execution to a line number within the desired module.

Ideally, the preceding steps should all be executed within the same program line. All program variables still reside in the regular 32K of memory and can therefore be shared by the main program and all of the modules. All string variables should be initialized by the main program. This will insure that no string pointers will point beyond $8000 hex.

To return to the main program from a module, simply POKE 41,4 and GOTO to a line in the main program. Again, these commands should be executed on the same program line in the module program.

Loading Programs In The 4K Banks

The BASIC program modules are written, saved, and loaded in the regular 32K PET memory normally. To transfer the module into the 4K bank, a small machine code program residing at $7FB0 hex can be used. This program must first be loaded into memory using the PET monitor before the modules are loaded in. Once this is done the appropriate module is called into memory using a load command, the appropriate bank number is then POKEd into location 61436 and the transfer is executed with the command SYS32688. Once all of the required modules are loaded in this way, the system is reset with the command SYS64790 and the main program is then LOADed in and RUN.

Note: Trying to see which bank is switched in by PEEKing location 61436 is not possible because the banks are latched in during the POKE command and checking this location after this gives a meaningless number.

This is a listing of the machine code program which transfers BASIC program module from main memory in the PET to location 9000 hex and makes it executable.

```
7FB0 A9 00       LDA #$00     ;SET UP LOW ADDRESS
                              POINTERS
7FB2 85 00       STA $00      ;HEX 0400
7FB4 A9 04       LDA #$04
7FB6 85 01       STA $01
7FB8 A9 00       LDA #$00     ;SET UP POINTERS TO HIGH
7FBA 85 02       STA $02      ;ADDRESS HEX 9000
7FBC A9 90       LDA #$90
7FBE 85 03       STA $03
7FC0 A2 0F       LDX #$0F     ;NO. OF 256 BYTE BLOCKS TO
7FC2 A0 00       LDY #$00     ;TRANSFER-1
7FC4 B1 00       LDA ($00),Y  ;DO THE ACTUAL TRANSFER
7FC6 91 02       STA ($02),Y
7FC8 88          DEY
7FC9 D0 F9       BNE $7FC4    ;FINISH THE BLOCK (256
                              BYTES)
7FCB E6 01       INC $01      ;MOVE TO NEXT BLOCK
7FCD E6 03       INC $03
7FCF CA          DEX
7FD0 30 03       BMI $7FD5    ;BRANCH IF LAST BLOCK
                              DONE
7FD2 4C C2 7F    JMP $7FC2    ;IF NOT CONTINUE TRANSFER
7FD5 A9 90       LDA #$90     ;CHANGE START OF BASIC
                              POINTER
7FD7 85 29       STA $29      ;TO 9000 HEX
7FD9 20 B6 B4    JSR $B4B6    ;RELINK THE BASIC TEXT AT
                              9000 HEX
7FDC A9 04       LDA #$04     ;RESTORE START OF BASIC
                              TO 0400
7FDE 85 29       STA $29
7FF0 60          RTS          ;RETURN
```

©

Atari Drives And Disk Operating Systems

Richard Kushner
High Bridge, NJ

It's now the tenth time that you've waited five minutes for your program to load from tape on your Atari and you have begun to wonder if there isn't a better way. Even though the Atari tape recorder seems to load flawlessly, the waiting time becomes a nuisance, not to mention the fact that you must be careful to set the recorder to just the right spot on the tape in order to give the computer just the right length of silence before the data comes streaming in.

And how many times have you lost a program during development because you didn't want to "waste" the time required to save it on tape? The answer, of course, is to get a floppy disk drive, in particular the Atari 810. There are not yet any non-Atari disk drives available for your computer. It isn't an easy decision to make when the disk drive costs more than half the cost of the computer. Let's try to cover the things that you need to know in order to make an intelligent purchase and also include an analysis of the two disk operating systems (DOS) that Atari has released: the original DOS I and the updated and improved DOS II.

Care And Feeding Of The 810

The 810 disk drive comes in a case measuring 4.5 H x 9.5W x 12D, attractively designed to match your computer. It comes with a separate power supply, just like the one that runs the computer. (A mild complaint: If each of the peripherals comes with one of these power supplies, where do we put them all?) With the 810's two input/output connectors, it is a simple matter to interconnect whatever items you have feeding into or out of your computer. There are two slide switches in the back which you set to tell the computer which drive it is addressing (#1-#14). With only one drive (and that covers nearly all of us) you set it for #1.

You must always power up the drive before the computer or else the computer will not be aware that there is a disk drive on the line. You also want to be sure that you do not have a diskette inserted when you power up, as this is an easy way to destroy any information on it. Keep in mind that the drive unit must be placed a minimum of 12 inches from the TV or telephone or anything else that generates a magnetic field. This is another easy way to wipe out a diskette.

As for diskettes, you can use soft or hard sectored ones because the Atari doesn't use the timing holes at all, but rather writes the timing information on the disk during formatting. You will quickly learn the do's and don'ts of physically handling diskettes. Be sure to use only felt-tipped pens when writing on labels attached to the diskettes. Failure to do this can ruin a diskette. Each side of a diskette will store about 92K [*92 kilobytes, or 92 x 1024 bytes*] which is roughly equivalent to four C-10 cassette tapes.

The power of the disk drive comes into play in the rate of reading and writing data and programs from and to the disk (6000 bits per second) and the identification of each file on a diskette by name. To run a program you simply type RUN "D1:TESTPROG.EXT", where D1 is the drive number, TESTPROG is the name of the program (up to eight characters, beginning with an alphanumeric, containing no spaces and using any letters or numbers) and .EXT is an extension that can be used to identify programs by type (i.e., .BAS for BASIC, .DAT for DATA, etc.). The DOS (much more on that later) searches the diskette for the program, loads it, and then begins to execute it. All the other Atari I/O commands are also available (SAVE, LIST, ENTER, PUT, GET, LOAD, PRINT, INPUT, etc.)

The time required to SAVE a program to diskette is longer than the time required to LOAD the program from the diskette because, during SAVE, the computer checks to be sure that the data has been correctly transferred. You can override this "write with verify" feature if you wish. For us former tape recorder users, the transfer rate is still so fast that there is no longer any time to go and get a snack while a program LOADs or SAVEs.

As the Atari software field matures, more and more programs are appearing only on diskettes, either because of length or because they need to make frequent access to disk stored data during execution. I have seen advertisements for Adventure-type programs that use up to six diskettes and database programs that are only practical on disk. There is even a magazine for the Atari that appears only on diskette. It has been reported that 90% of all Apple owners have disk drives. As a final argument, for those who are still holding onto their money, the purchase of a disk drive gives you the opportunity to turn your Cadillac of personal computers into a Ferrari. Enough said?

Atari DOS I Versus DOS II, Or Vas Ist DOS?

Just as the computer requires an operating system in order to function, so, too, does the disk drive. The disk operating system (DOS) comes recorded on a diskette and is the first thing that the computer looks for when it is powered up. I will be concentrating on DOS II in this article, referring to DOS I when it seems appropriate to point out the improvements that DOS II brought about.

DOS II consists of three files; DOS.SYS, DUP.SYS, and AUTORUN.SYS. DOS.SYS boots when the computer is turned on. You can then direct all BASIC commands to the disk drive. DUP.SYS contains the utilities to run the DOS menu and is called from BASIC by typing DOS. AUTORUN.SYS is intended to boot automatically after DOS.SYS and can be modified to perform any function that the user feels should automatically come on line. It currently looks for the 850 Interface Module handler and, if it finds it, the computer loads it. This corrects one of the flaws in DOS I. Any file may be named AUTORUN.SYS, thus booting automatically on initialization.

Speaking of flaws, DOS I has several, all of which are corrected in DOS II. Those with DOS I may be unaware of some of these flaws (and therefore blame themselves or their computer for some odd happenings), so here is a list of the more significant problems:

1. The DUPLICATE DISC command occasionally has trouble with the first eight sectors and, if DOS is in these sectors, you will get a diskette that will not boot.

2. AUTO.SYS loads files, but won't execute them.

3. COPY FILE and DUPLICATE FILE commands allow illegal use of wildcards and this causes problems.

4. If FORMATting finds bad sectors, the user is not informed.

5. In some instances, full length filenames are not accepted.

6. Sectors on diskettes are sometimes taken out of user availability when they shouldn't be.

DOS II boots in seven seconds. You are then in BASIC and can use all the I/O commands to communicate with the disk drive. In order to use the DOS menu commands, you must type DOS. After about nine seconds you get a menu [*a list of choices*]. The reason for this method is that then it is unnecessary to use active computer memory to hold the menu functions unless they are specifically needed.

DOS I has the menu automatically boot and this consumes a considerable amount of memory (about 6K) which is then unavailable for program storage unless you dump DOS. DOS II could potentially get you in trouble when you called for the menu, since it requires a portion of memory that you might also be using and, hence, you would lose your program. This has been neatly taken care of by allowing the user to create a MEM.SAV file on his diskette. MEM.SAV is a place for the computer to put that part of memory that DUP.SYS needs for the menu. Thus, the computer automatically saves your program before going to the menu and restores your program after you release the menu. The penalty here is time, since MEM.SAV takes about 30 seconds to operate. The alternative is to SAVE any resident programs before calling the menu and not use MEM.SAV at all. There is a tradeoff here since MEM.SAV provides a nice degree of protection. Anyone who has lost a program after hours of typing will appreciate this feature.

The rest of the menu commands relate to operations on the diskette-stored files, and their names are pretty much self-explanatory. Their implementation is well described in the instruction manual. An area of major concern is the compatibility of DOS I and DOS II formatted diskettes. All diskettes must be formatted before they can be used and the DOS I and DOS II formatting is not the same. Using DOS I on DOS II formatted diskettes will destroy files, whereas using DOS II on DOS I formatted diskettes will either work or else give an error message. Therefore, the rule of thumb is to always use DOS II, especially if you are not sure of the diskette formatting.

There is a bug in DOS II for those who have an 850 Interface Module with something attached to a serial port on it. The COPY command, which allows you to copy a file disk-to-disk, disk-to printer, or disk-to-whatever, does not allow you to copy to the serial port. This is because the Interface Module handler, which boots at the same time as DOS, occupies the same space in memory as DOS and, thus, gets garbled. The result of a COPY to the serial port is that the computer goes off to never-never-land and you have to turn it off and reboot to regain control (losing any programs in memory). As I write this, Atari is working on the problem of relocating the Interface Module handler out of harm's way.

The Dark Side Of The Force

The 810 Disk Drive and/or DOS II have some "sometimes" bugs. There are occasions when disks from my drive will not run on other drives and vice versa. This was traced to speed variations in the drives. In fact, Atari redesigned their disk drive to incorporate an external data separator. This has effectively cleared up this problem. For those with older disk drives, Atari will be offering to upgrade those drives.

An alternative is a do-it-yourself installation that has been promoted by Atari Users' Groups and which is easy to do. Also available from either of these two sources is a fast formatting chip modification that speeds up the read-write operations. On the horizon are double density and dual disk drives from other suppliers. The latest Atari computers being sold also have a modified operating system board in them. The most significant change when using these will be the elimination of the annoying pauses that occur during communication with the disk drive.

[*For a third alternative to disk speed adjustment, see "Getting Your Atari Disk Drive Up To Speed,"* **COMPUTE!**, *May, 1982, #24*.]

Atari Disk Menu With DOS II

The "Atari Disk Menu" program in **COMPUTE!**, January, 1981, #8, is a very useful program which enables the user to access programs on disk without having to remember their names or their exact spelling. This program will not, however, run on DOS II. However, it is easy to fix the program *and* add the feature of printing out the free sectors available on the disk.

With DOS II there is a difference in format for the "free sectors file" between DOS I and DOS II. In DOS I this file is simply a number indicating the sectors that are unused, while in DOS II this file has the number of sectors available plus the phrase "free sectors." Therefore, when line 130 tests for the length of the file (looking for <5 if the file is the DOS I "free sectors" file) it fails in DOS II (which has a longer file length) and gives an "end of file" error and stops.

A consideration of the file format makes a fix very easy. All program name files consist of two spaces plus a file name of up to eight characters plus an optional decimal point and three character extender. For example, a file name may be TEST-RUN4.BAS where the length of the file name may be less than this. The filename that holds the free sector information, however, is made up of the number of free sectors plus the phrase "free sectors." The key is the absence of the leading spaces. Therefore, if line 130 becomes

```
130 IF FILE$(2,2) <> " " THEN PRINT FILE$:
    GOTO 500
```

When the program encounters the free sectors file it will print it out, thus giving the number of sectors, and go on to ask which program you wish to run. FILE$(2,2) is used rather than (1,1) because if the file is locked, FILE$(1,1) will have a "*" in it. This small change makes a useful program even more useful. ©

A Simple 2716 EPROM Programmer For The PET

Phil Gentile
Belton, MO

PET owners who are involved in machine language programming usually collect a number of subroutines that aid in the operation of the PET. These subroutines may consist of repeat key functions, screen print programs, or other helpful machine language routines which may be written by the user or found in magazines.

These subroutines must all be loaded into RAM before they can be used. The second cassette buffer, or a portion of high RAM, is usually the choice.

Now, for just a few dollars and a little time, you may put all of these programs in EPROM [*Erasable, Programmable ROM. These chips are a compromise between RAM and ROM – like ROM they will hold information with the power off. Like RAM, you can reprogram them (by erasing them using special techniques).*] EPROM is a better location for these programs for two reasons. First, if the programs are in EPROM, they don't have to be loaded; the programs are there when you power up. Second, in EPROM, the programs don't take up any of your precious RAM space.

There are a number of EPROM programmers on the market, ranging in price from many hundreds down to fifty or sixty dollars. They all do basically the same thing, the expensive ones are just fancier and more versatile in the types of EPROMs which they can program.

This EPROM programmer, as it is presented here, can only program a 2716, or 2716 compatible EPROM. The 2716 is a 2K byte EPROM. I have found 2K to be more than enough space to hold all of the machine language programs that I need, including the DOS Wedge. If you find that you need more space, the programmer and the accompanying software may be easily modified to program larger EPROMs. You may do this by connecting the unused address lines on IC-2, and modifying the M/L "program pulse" subroutine to satisfy the requirements of the EPROM you select.

The EPROM programmer is very simple. It consists of three 74164, eight bit shift registers, one 24 volt regulator, a couple of switches and three 9 volt batteries. One shift register holds the data shifted out by the PET. The remaining two shift registers hold the EPROM addresses as they are shifted out. The 24 volt regulator reduces the voltage from the series wired batteries down to the 24 volts necessary to program the 2716 EPROM. The PET, using the accompanying software and the PET's user port, supplies all of the necessary data, addresses, and signals to program the EPROM. The five volt VCC supply can also be taken from the PET if you use low power (ls) type IC's. This circuit, using "ls" IC's, will draw approximately 100 Ma. This small amount of current may safely be drawn from pin B of the second cassette edge connector (J3).

You may use any type of construction you wish to build the circuit. Wire wound, PC board, or plug-in type breadboard are all OK. If you wish to save the considerable expense of a zero force socket for the EPROM (about $9.00), then I would recommend the plug-in type breadboard construction. The EPROM can be easily inserted and removed from this type of board, reducing the chance of damage to the EPROM pins.

The 74164 shift registers are common IC's and can be purchased in most electronics supply stores. The 24 volt regulator is also a common component. Be sure to get the positive rather than the negative type regulator. The switches are miniature SPDT. The batteries used are standard 9 volt transistor type. I used the heavy-duty versions.

There are only a few things to watch out for. Be sure that the regulator is wired properly. The allowable voltage range on pin 21 of the 2716 EPROM is 24 to 26 volts. Be careful when handling the EPROM. Static electricity on the pins may destroy it. Last, but most important, *always* follow the proper sequence when applying the voltages to the EPROM. The 5 volt VCC supply *must* be turned on first, followed by the 24 volt VPP supply. When turning the voltages off you must reverse this procedure. If you follow the instructions as they are given by the program, you will have no problem. If you don't, you may destroy your EPROM.

Software Control

The software to control the EPROM programmer consists of a small BASIC program and three small machine language subroutine modules that reside in RAM at 4100 through 4276.

The BASIC program is used primarily to interface with the user. It issues instructions for the operation of the EPROM programmer and accepts user information on where the machine language code (to be programmed into the EPROM) is located in the PET's memory. It also asks the user for the EPROM starting address. The BASIC program then gets the selected code from the PET's memory and passes it to the machine language modules. They, in turn, send the code to the EPROM programmer.

The first M/L module, located at 4104, sends the data from the PET's memory serially out over the PA-0 line. As it sends this data out over PA-0, it also sends a shift (clock) pulse out on the PA-1 line which is connected to the data register, IC-1. This clock pulse loads the data on PA-0 into the data register.

Next, the BASIC program jumps to the M/L routine, located at 4184, which shifts out the EPROM address. This routine sends the address out on the PA-0 line also, but it sends the shift signal out over the PA-2 line which is only connected to the address register (IC-2 & IC-3). Since the data register is not connected to the PA-2 line, the information stored in this register is not affected when the EPROM address information is shifted into the address register.

Finally, after all of the data and EPROM address information is safely stored in the registers, the machine language module located at 4107 sends a 50 milisecond "program" pulse to the 2716 EPROM telling it to store the byte of data in the selected EPROM address. This "program" signal is sent out over the PA-3 line. The BASIC program then begins the cycle all over again for the data contained in the next core location. This continues until all of the PET memory addresses, requested by the user, have been programmed into the EPROM. The BASIC and machine language programs occupy RAM locations 1025 through 4276. You may place your EPROM code anywhere else in RAM that you wish.

To use this EPROM programmer you must first, of course, have some machine language code that you want to put in EPROM. The example I will use here is a procedure for putting Commodore's DOS Wedge in EPROM. The machine language code for the DOS Wedge is completely relocatable. After the Wedge has been loaded from disk and executed, the Wedge M/L code resides in the PET's top 359 bytes of RAM. To find out exactly where this code has been stored, go to the machine language monitor and display locations $0070 thru $0072. The program that loads the Wedge, stuffs a jump to the address of the Wedge into these locations. If the Wedge has been loaded, you should see a "4C" in location $0070, followed by the low-byte, hi-byte address of the Wedge. On my 16K PET, this code reads, "4C 99 3E". This translates into the decimal location, 16025.

Now we know where the code for the DOS Wedge is in core, so let's put it into EPROM. First hook the EPROM programmer up to the PET's user port. Do not put the EPROM in its socket yet. Now hook up both the 5 volt and the 24 volt power supplies. Make sure that both of the switches on the programmer are turned off. Put the EPROM into it's socket on the programmer and load the EPROM programming software. The program will tell you to first turn on the 5 volt, and then the 24 volt power supplies. Do this in the sequence described.

The program will next ask you for the starting RAM address of the machine language code that you wish to put into the EPROM. In this case, this is the address that we found in locations $0071 and $0072. The low order byte of the address is in $0071, the high order byte in $0072. Convert this address to its decimal value, in the case of a 16K PET, 16025, and enter it. The next address that the program asks for is the ending address of the code. The DOS Wedge is 359 bytes long, so your ending address is 16383 and press return. Next, the program asks where you want the code to start in the EPROM. If you are using a new EPROM, without any code previously programmed into it, you will probably want to start in location zero. Enter the EPROM starting address and press RETURN. The machine language code for the DOS Wedge is now being programmed into your EPROM.

After it has finished programming the EPROM (approx. 1.5 minutes), the program will give you the last EPROM address programmed. If you started at EPROM address zero, this address will be 358. Write this address somewhere for future reference. Turn the power switches off in the order given by the program. Leave the EPROM in the socket for now.

Now that we have the Wedge in EPROM, we must tell the PET where we are going to put it. We'll use the same method that the Wedge loader uses. For the sake of this demonstration, we will assume that the EPROM is going to be placed in the PET's empty EPROM slot at address $9000. Go to the machine language monitor and display locations $03E4 through $03F0. Enter the following code starting in $03EF, "A9 4C 85 70 A9 00 85 71 A9 90 85 72 60." This code, when executed, will load a jump to location $9000 into locations $0070 thru 0072. Get out of the Monitor and execute the EPROM programmer program again. This time your starting RAM address will be the starting address of our little loader patch, 996 (hex 03E4). The ending RAM address will be 1008 (hex 03F0). The EPROM starting address will be one more than the EPROM ending address that we saved after our last run (358 + 1 = 359). When the program finishes, the EPROM contains everything we need to run the DOS Wedge from EPROM.

Turn the PET off and install the programmed EPROM in the EPROM socket at $9000. Turn the PET back on and jump to the EPROM starting address of our little loader program, SYS(37223), ($9000 = 36864 + 359 = 37223). You should now be able to execute any DOS Wedge command from the EPROM.

I don't know how long the 9 volt batteries will last. I have programmed a couple of EPROMs, and have done a lot of testing using the original set, they still seem to have a lot left.

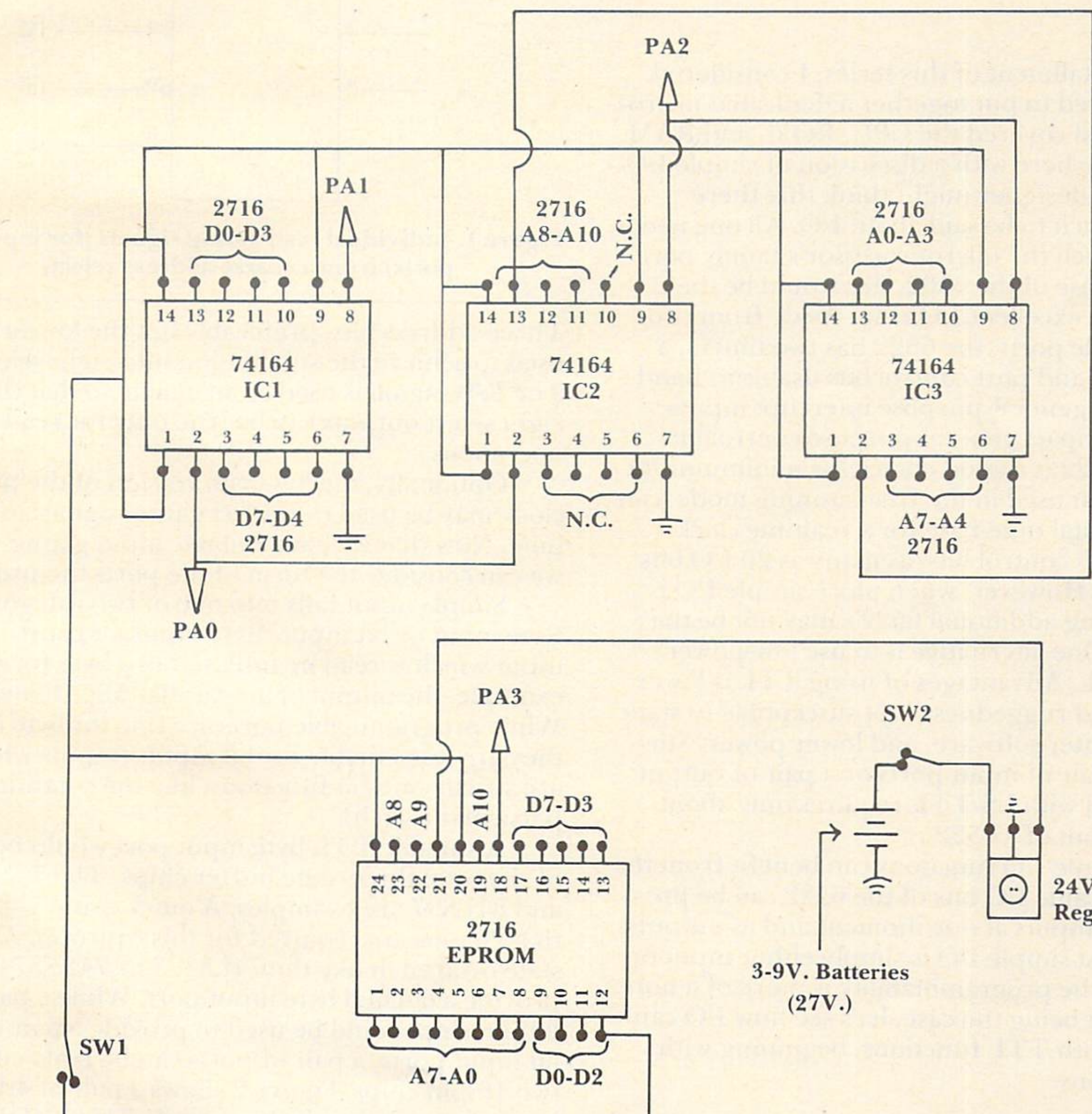
With the software written partly in BASIC and partly in machine language this system will program approximately three bytes per second. If you wish to make it operate faster, the software could be written entirely in machine language.

If you are careful to follow the schematic, and in entering the program code, you will have an EPROM programmer costing around fifteen or twenty dollars. That is a lot cheaper than buying one. In addition, there is a lot more fun, satisfaction, and knowledge to be gained by building your own.

```
170 ** LOAD ML PROGRAM MODULES ***
180 POKE53,16:POKE4102,0:POKE4103,0
190 READX,Y
200 FORI=XTOY
210 READ Z
220 POKEI,Z
230 NEXTI
240 :
250 POKE59459,15:POKE59471,0
260 PRINT"{03 DOWN}TURN ON THE 5 VOLT POWER
     SUPPLY.
270 PRINT"AND PRESS RETURN.
280 PRINT
290 GETA$:IFA$=""THEN290
300 PRINT"NOW TURN ON THE 24 VOLT POWER SUP
    PLY
310 PRINT"AND PRESS RETURN.
320 PRINT
330 GETA$:IFA$=""THEN330
340 INPUT"STARTING RAM ADDRESS (DEC.)";R1ST

350 INPUT"ENDING RAM ADDRESS (DEC.)";R2E
360 PRINT:INPUT"STARTING EPROM ADDRESS (DEC
    .)";R3ST
370 IFR3 > 2047 THEN PRINT:PRINTR3" EXCEEDS
     2716 EPROM SIZE.":PRINT:GOTO360
380 N=R3: GOSUB740
390 HI$=LEFT$(N$,2):LO$=RIGHT$(N$,2)
400 N$="00"+HI$:GOSUB910:POKE4102,N
410 N$="00"+LO$:GOSUB910:POKE4103,N
420 :
430 REM *** SHIFT THE DATA OUT ***
440 :
450 PRINT"{CLEAR}PROGRAMMING FROM RAM LOCAT
    ION:":PRINT
460 FORK=R1TOR2
470 PRINTK
480 A=PEEK(K)
490 POKE 4104,A
500 SYS(4152)
510 :
520 REM *** SHIFT ADDRESS OUT
530 :
540 SYS (4184)
550 :
560 REM *** SEND OUT THE M/L PROGRAM PULSE 
    ***
570 :
580 SYS (4107)
590 :
600 NEXTK
610 PRINT
620 HI=PEEK (4102)
630 LO=PEEK (4103)
640 N=HI:GOSUB 740:HI$=RIGHT$(N$,2)
650 N=LO:GOSUB 740:LO$=RIGHT$(N$,2)
660 N$=HI$+LO$:GOSUB 910
670 PRINT "LAST EPROM ADDRESS WAS "N-1
680 PRINT:PRINT "TURN OFF THE 24 VOLT SUPPL
    Y
690 PRINT"AND THE 5 VOLT SUPPLY, IN THAT OR
    DER!"
700 PRINT"
710 END
720 :
730 REM ** DECIMAL TO HEX **
740 A=N:X=0
750 A=A-4096
760 IF A<0 THEN A=A+4096:Z=X:GOSUB970:E$=X$
    :GOTO 780
770 X=X+1:GOTO 750
780 B=A:X=0
790 B=B-256
800 IF B<0 THEN B=B+256:Z=X:GOSUB970:F$=X$:
    GOTO 820
810 X=X+1:GOTO790
820 C=B:X=0
830 C=C-16
840 IF C<0 THEN C=C+16:Z=X:GOSUB970:G$=X$:G
    OTO860
850 X=X+1:GOTO 830
860 D=C:Z=D:GOSUB970:H$=X$
870 N$=E$+F$+G$+H$
880 RETURN
890 :
900 REM ** HEX TO DECIMAL **
910 X$=RIGHT$(N$,1):GOSUB1000:D=Z
920 X$=MID$(N$,3,1):GOSUB1000:C=Z*16
930 X$=MID$(N$,2,1):GOSUB1000:B=Z*256
940 X$=LEFT$(N$,1):GOSUB1000:A=Z*4096
950 N=A+B+C+D
960 RETURN
970 IF Z<10 THEN X$=STR$(Z):X$=RIGHT$(X$,1)
    :RETURN
980 X$=CHR$(Z+55):RETURN
990 :
1000 IF X$<"A" THEN Z=VAL(X$):RETURN
1010 IF X$=" " THEN Z=0:RETURN
1020 Z=(ASC(X$)-55):RETURN
1030 :
1040 REM *** DATA STATEMENTS FOR M/L CODE **
     *
1050 DATA 4100,4276
1060 DATA 169,5,141,248,3,32,169,169,5,141,9
     ,16,32,29,16,173
1070 DATA 9,16,201,0,240,2,208,244,96,169,8,
     160,144,141,79,232
1080 DATA 162,48,202,208,253,136,208,248,169
     ,0,141,79,232,206,9,16
1090 DATA 96,1,2,4,162,8,173,8,16,45,53,16,1
     41,79,232,13
```

```
1100 DATA  54,16,141,79,232,169,0,141,79,232
    ,202,240,6,78,8,16
1110 DATA 76,58,16,96,173,6,16,141,4,16,173,
    7,16,141,5,16
1120 DATA162,8,173,7,16,45,53,16,141,79,232,
    13,55,16,141,79
1130 DATA 232,169,0,141,79,232,202,240,6,78,
    7,16,76,102,16,162
1140 DATA8,173,6,16,45,53,16,141,79,232,13,5
    5,16,141,79,232
1150 DATA169,0,141,79,232,202,240,6,78,6,16,
    76,133,16,24,216
1160 DATA173,5,16,105,1,141,7,16,173,4,16,10
    5,0,141,6,16
1170 DATA 96
```

©

The previous article in this series appeared in **COMPUTE!**, *November, 1981, #18, pg. 160.*

Nuts And Volts

Build Your Own Controllers

Eugene M. Zumchak
Buffalo, NY

In the last installment of this series, I considered what is required to put together a dedicated microcontroller, and covered the CPU, ROM, and RAM. I will continue here with a discussion of simple I/O.

A novice designer might think that there really isn't much to be said about I/O. All one needs to do is to attach the microprocessor's family port chip. In the case of the 6502, this would be the 6522 VIA, an excellent I/O chip. Aside from two programmable ports, the 6522 has two timers, a shift register, and port control bits usable in handshaking or as general-purpose interrupt inputs.

For a compact general-purpose controller board, the 6522 is a good choice for minimum I/O. Timer 1, when used in the free-running mode, can provide a crystal time-base for a realtime clock. And including control bits, as many as 20 I/O bits are available. However, when more simple I/O is required, using additional 6522's may not be the best choice. One alternative is to use low-power Schottky TTL. Advantages of using TTL is lower cost, increased ruggedness (not susceptible to static damage), tighter software, and lower power. Surprisingly, a pair of input ports or a pair of output ports realized with LS-TTL requires only about half the current of a 6522.

While some I/O functions can benefit from the fact that the same I/O bits of the 6522 can be programmed as inputs at one moment and as outputs the next, most simple I/O is simply either input or output, and the programmability is more of a non-feature. That being the case, let's see how I/O can be realized with TTL functions, beginning with input functions.

Simple Input

The unit input function is a three-state gate and a suitable gating signal which will be true when a read operation is performed using a specific address. Assume that our input functions will use a coarse address select with base address $2000. (This could be a 1K, a 2K, a 4K, or even an 8K select). A single 74LS138 3-to-8 decoder can be used to further decode this coarse select into eight individual gating signals as shown in Figure 1.

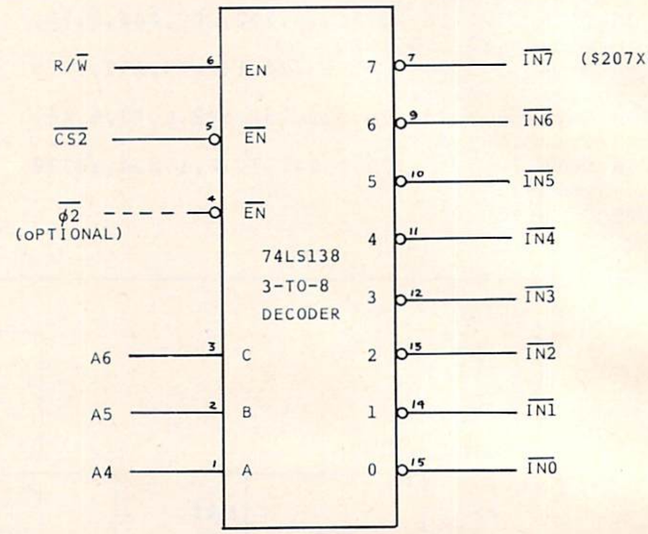


Figure 1. Individual read gating signals (for input ports) from a coarse address select.

Three address bits, preferably not the lowest, are used to achieve the subdivision into eight selects. The R/W signal is used as an enable so that the eight select outputs can be true only for read operations.

Optionally, the low-true version of the phase-2 clock may be used to restrict gating signals to ϕ2 time. Now that we have suitable input gating signals we can consider the form of the ports themselves.

Simple input falls into one of two categories. Byte input or bit input. Byte input, of course, is input which is read in and used in a byte form; for example, the output of an parallel ASCII keyboard. While programmable ports are fine for byte inputs, they are less suitable for bit inputs (inputs which are strictly one-bit functions, like the condition of a particular switch).

A suitable TTL byte input port would be any of the octal three-state buffer chips. The 74LS244 and 81LS97 are examples, A quick count will show that 20 pins are required for this function. A three-state octal latch like the 74LS373 or 74LS374 can provide a latched byte input port. While a pair of 20-pin chips could be used to provide a pair of 8-bit input ports, a pair of ports can be realized with two 16-pin chips. Figure 2 shows a pair of 4-bit input ports, realized with a single 74LS257 three-state quad 2-to-1 select chip. (Two chips will give two 8-bit wide ports.) An address bit, usually A0, is used to select between the two ports. A pair of these chips takes typically 20 mA, much less than a pair of ports from a 6522.

Most designers use byte-style input ports for

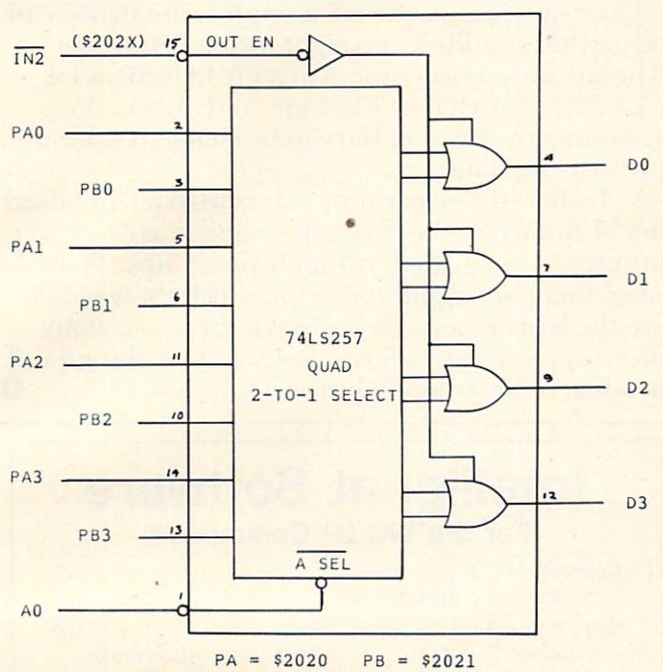


Figure 2. Two 4-bit input ports from a three-state quad 2-to-1 select gate.

bit input functions. A particular bit in a port can be isolated for making a decision by loading in the port data, ANDing with an appropriate mask, and branching based on the zero flag. However, an 8-bit input port can be realized with a 16-pin 74LS251, a three-state, 8-to-1 select function. The single output bit is brought in on data bit D7. Now any of the eight bits may be individually addressed and tested. Such a bit-addressable port is shown in

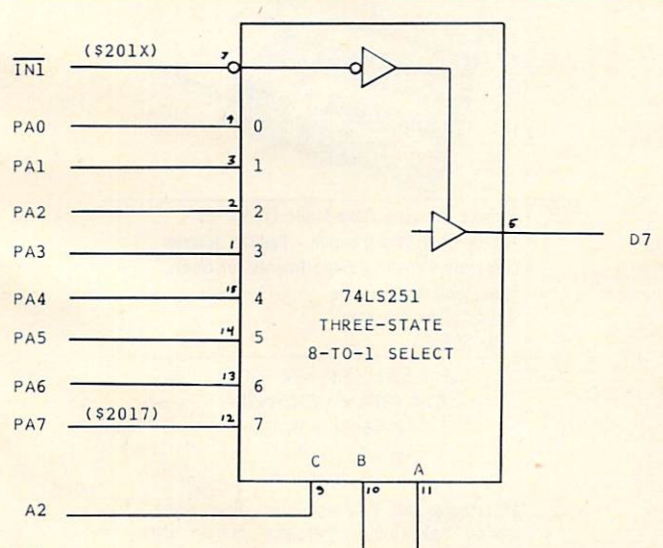


Figure 3. Eight-bit input port with individually addressable bits.

Figure 3. In particular, any bit can be tested with a BIT instruction to the bit address, followed by a branch based on the sign flag. Thus, input bit tests take only two, not three, instructions, and do not change the A register. Since each individual bit in the port has its own unique address, each bit function in software will have its own unique label, making the software somewhat more readable. That is far superior to calling a byte of eight switches *SWITCHES*.

Simple Output

Like simple input, simple output can be bit output and byte output, but also pulse output. As with inputs, the first job is to provide suitable control signals for our output functions. A single 3-to-8 decoder chip can provide eight individual write strobing signals from a coarse address select. (Figure 4.)

In this case, the ϕ2 clock signal is not optional since write strobing action is accomplished with

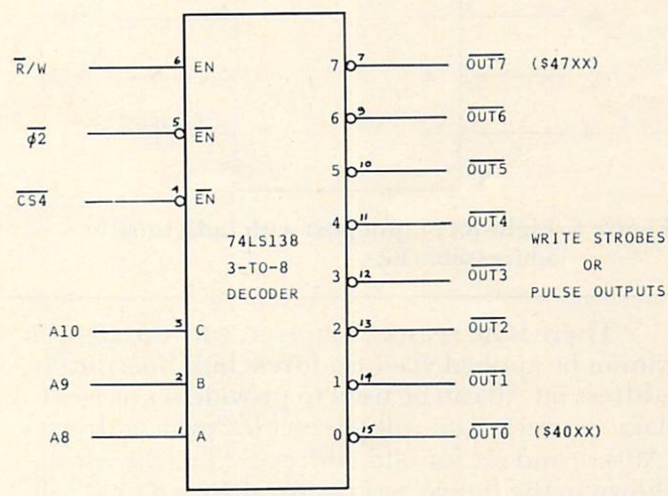


Figure 4. Individual write strobe signals for output ports from a coarse address select.

this signal in 6502 timing. The application of the low-true version of the ϕ2 delay in the individual outputs and an acceptable amount of write hold-time loss. (See the early Nuts & Volts columns on read/write timing).

Byte style output ports can simply be any of the numerous octal latches in 20-pin packages. While the board space requirement for a pair of such ports is about the same as for a 6522, the power is somewhat less. As with input functions, most designers use byte-style output ports to achieve single bit outputs. The differences are resolved with software. To turn off (or on) a particular bit, without changing other bits in a port, the output port (or status) is loaded, an appropriate AND (or OR) mask used, and the altered data is written to the port. This takes three instructions and eight bytes. However, if a port with individually addressable bits is used, any arbitrary bit can be turned on or off with a single three byte instruction, and without changing the accumulator. The port itself is shown in Figure 5, and uses the not-well-known 74LS259 addressable latch chip. This chip has three address lines to select one of eight output bits. A single data line is used to apply the one or zero to set or clear the selected bit.

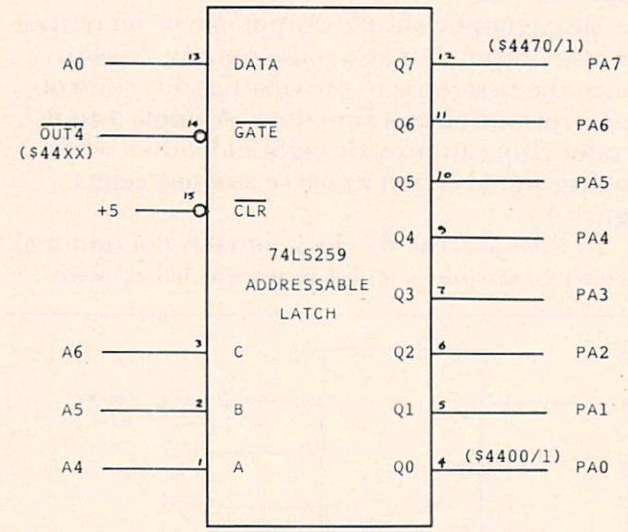


Figure 5. Eight-bit output port with individually addressable bits.

There is no reason, however, why this data bit cannot be applied via the address bits. Specifically, address bit A0 can be used to provide set or reset data. A selected bit will be reset for even addresses (A0 = 0) and set for odd addresses. For the wiring shown in the figure, writing to address $4400 will clear bit zero. Writing to $4451 will set bit five. That is, the least hex digit is either zero or one, and the next-to-least hex digit is used to select one of eight bits. Thus, a STA $4461 will set bit six where the contents of A is totally irrelevant. That is, it is unnecessary to load A (or X or Y) before the write. As with bit addressable inputs, the complementary bit addresses can be given software names like VALVE/ON, VALVE/OFF, BELL/ON, etc. Nothing, software-wise, could be simpler or faster.

Finally, some output functions have the form of one-bit pulses or triggers. These might be external clocks, resets, etc. To achieve a pulse output with a byte-style port requires loading output status, ANDing the selected bit off, writing out the new status, ORing the bit back on, and writing out the new status. This takes a minimum of five instructions and thirteen bytes. A pulsed output can be obtained directly from an unused individual write strobe as shown in Figure 4. If many pulse-type outputs are required, a single write strobe can be further decoded into eight (or more) strobes. The software now reduces to STA PULSE, STA CLEAR, STA COUNTUP, etc. and, again, the accumulator does not have to be changed since it provides no data.

It should be clear now that controller I/O need not be limited to the expensive, slow, fragile, power-hungry I/O provided by family port chips. To be sure, there are applications for which these chips are the best choices, but perhaps there are many more applications where a little creative design will produce a better (and cheaper) solution. ©

VIC's Perpetual Calendar

Robert Lewis
Decatur, IL

Have you ever wanted to see a calendar for next year? Last year? Or the year you were born? This program is able to produce a *correct* calendar for any year after 1900. There are other starting years in program lines 10-60. These years are 1950, 1800, and 1582; the year today's calendar was started.

To change the starting year, delete the REM statement from the year wanted *and* the next following line. Be sure to add REM statements to the old lines or you may get an error when the program is run.

To use the program simply INPUT the full year (example: 1982) when asked "YEAR WANTED?". After entering the date, the program uses a loop to find what day of the week January first falls on in that year. A large difference in the entered and starting years may cause a delay of several seconds.

When it leaves the loop, it checks if it is a leap year and adjusts the count. There are several checks for leap year by the formula: IF Y/4=INT (Y/4)THEN...

The last half of the program is devoted to displaying each month on the screen. The VIC's 22 character screen is an asset! The display consists of (starting from the top) the year, the number of days passed and left, the month, the days of the week and, finally, the dates. February and the days left and passed are automatically compensated for in leap years. All calendars start with January and each preceding month is displayed by pressing any key but (F1). When (F1) is pressed, the program asks for another "YEAR WANTED?" with a statement about the starting year. This also occurs when you enter a date before the starting year. At December, when you press a key, the program goes to January of the next year.

```
10 REM REMOVE REMARK FROM YEAR DESIRED AND
    ADJACENT LINE
15 REM AND ADD TO THE OLD LINES
20 REMYY=1800
21 REMCC=2
30 YY=1900
31 CC=1
40 REMYY=1950
41 REMCC=0
50 REMYY=1582
60 REMCC=2
65 REM INSTRUCTIONS
70 POKE36879,8
80 B1$="  VIC'S       ":  B2$="  PERPETUAL ~
    ":  B3$="  CALENDAR   "
90 PRINT"{CLEAR}":PRINT:PRINT"{REV}{GRN}    ~
     "
100 PRINT"{REV}         _"
110 PRINT"{REV}          _"
120 FORI=1TO13:PRINTTAB(I-1)"_{REV} "MID$(B
    3$,I,1);" ";MID$(B2$,I,1);" ";MID$
    (B1$,I,1);" _{OFF}
125 NEXTI
130 PRINT"{WHT}BY:{GRN}";SPC(I-4);"_{REV}  ~
        {OFF}"
140 PRINTSPC(I);"_{REV}       {OFF}"
150 PRINT"{WHT}ROBERT LEWIS{GRN}";SPC(I-11)
    ;"_{REV}      "
160 PRINT"{HOME}":PRINT:PRINT:PRINT"{REV}{Y
    YEL}"SPC(15);YY;"{LEFT} {OFF}":PRI
    NTSPC(17)"'%"
165 PRINTTAB(15);"{REV} ???? {OFF}"
170 PRINT:PRINTTAB(15)"{CYN}INST?"
180 GETA$:IFA$=""THEN180
190 IFA$="N"THEN330
200 PRINT"{GRN}{CLEAR} THIS PROGRAM MAKES  ~
    UP A CALENDAR FOR ANY"
210 PRINT"YEAR {RED}{REV}AFTER";YY;"{OFF}."

220 PRINT:PRINT"{CYN} OTHER STARTING YEARS ~
    ARE IN LINES 10-60."
230 PRINT:PRINT"{BLU} TO ENTER ANOTHER CAL-
    ENDAR YEAR PRESS {REV}F1{OFF}."
240 PRINT:PRINT"{YEL} THE YEAR ALWAYS BE-  ~
    GINS IN JANUARY.  TO "
250 PRINT"SEE THE FOLLOWING     MONTHS PRES
    S ANY OTHER KEY.";
260 PRINT" {PUR}(IN DECEMBER THE CALENDAR G
    OES TO THE NEXT YEAR)":PRINT
270 INPUT"{WHT}YEAR WANTED";Y
280 DA=365:IFY/4=INT(Y/4)THENDA=366
290 H=0:DB=DA
300 IFY>YYTHEN360
310 RESTORE
320 K=0
330 PRINT"{CLEAR}{WHT}I CAN'T START BEFORE ~
    ";YY
340 POKE36879,8
350 GOTO270
355 REM YEAR LOOP
360 Z=Y-YY:C=CC
370 FORI=1TOZ
380 C=C+1:YX=YY+I
390 IFYX/4=INT(YX/4)THENC=C+1
400 IFC>=7THENC=C-7
410 NEXTI
420 IFY/4=INT(Y/4)THENC=C-1:IFC<0THENC=7+C
```

```
425 REM READS MONTH
430 READM$,M:S=LEN(M$)
435 REM ADJUST FOR LEAP YEAR
440 IFM=3AND(Y/4=INT(Y/4))THENM=2
450 IFC>7ORC=7THENC=C-7
460 B=C
490 REM PRINTS MONTH
500 PRINT"{BLK}":K=K+1:IFK=6THENK=0
510 POKE36879,40+(K*16)
520 PRINT"{CLEAR}":PRINT"{REV}"TAB(8);Y;"{L
    LEFT} {OFF}"
530 PRINTTAB(1);H;TAB(16);DB
540 PRINTTAB(10-(S/2))"0";:FORI=1TOS:PRINT"
    @";:NEXTI:PRINT"."
550 PRINTTAB(10-(S/2))"]{REV}";M$;"{OFF}]"
560 PRINTTAB(10-(S/2))"-";:FORI=1TOS:PRINT"
    @";:NEXTI:PRINT"="
570 PRINT:PRINT" S  M  T  W  T  F  S ":PRIN
    T
580 PRINTSPC(3*B);
590 FORI=1TO31-M
600 PRINTI;:IFI>9THENPRINT"{LEFT}";
610 IFPOS(0)>19THENPRINT:PRINT
620 NEXTI:PRINT
700 GETX$:IFX$=""THEN700
705 REM NEW DATE WANTED
710 IFX$="{F1}"THEN310
715 REM SETS UP NEXT
716 REM MONTH OR YEAR
720 C=B+3-M
730 IFM$<>"DECEMBER"THEN790
740 RESTORE
750 Y=Y+1
760 H=0:DB=365:IFY/4=INT(Y/4)THENDB=366
770 DA=DB
780 GOTO430
790 DB=DB-(31-M):H=DA-DB
800 GOTO430
810 END
1000 DATA JANUARY,0,FEBRUARY,3,MARCH,0,APRIL
     ,1,MAY,0,JUNE,1,JULY,0
1010 DATA AUGUST,0,SEPTEMBER,1,OCTOBER,0,NOV
     EMBER,1,DECEMBER,0
```

©

A Subroutine Aid To Debugging Atari BASIC

Mark Thomas Greene, Ph.D.
Columbus, OH

Tired of searching through your BASIC manual, 850 manual, and DOS manual to find out what ERROR 175 means? This program should end that frustration once and for all.

The subroutine acts as a BASIC diagnostic that: a. describes in some detail what went wrong, b. indicates the line at which the error occurred, and c. lists the immediate environment of the problem line.

Here's how it works:

Line: 0 TRAP 30000
This causes the system to run the debugging routine whenever an error occurs. The subroutine should be stored on disk via a LIST command (e.g. L. "D:ERRORS.LST") so that it may be merged with any program by means of an ENTER command (e.g. E."D:ERRORS. LST"). Use of line 0 and lines 30000 through 31173 makes it easy to avoid overwriting an existing program with "ERRORS" and vice-versa.

Any time a different TRAP is needed, TRAP 30000 should be reset as soon as possible.

Line: 30010 ? " [ESC] [CNTL CLR]"
This statement clears the screen.

Line: 30020 STPLN1 = PEEK(187): STPLN2 = PEEK(186)
This statement retrieves the two byte representation of the line number at which the error occurred.

Line: 30030 ERR$ = "31"
We're going to build a line number beginning with thirty-one thousand and ending with the error number.

Line: 30040 IF PEEK(195) < 100 THEN ERR1$ = '0'

Line: 30050 IF PEEK(195) < 10 THEN ERR1$ = '00'
These lines create place holders in front of the error number if the error number is less than three digits long. Line 30050 overrides 30040 if the error number [PEEK(195)] is a single digit.

Line: 30060 ERR$(LEN(ERR$) + 1) = ERR1$
This line adds the appropriate number of zeros to ERR$ ('31'). If the error number is three digits long it will have no effect.

Line: 30070 ERR$(LEN(ERR$) + 1) = STR$ (PEEK(195)
This line adds the error number to our string so that we have a five digit string beginning with '31' and ending with the error number.

Line: 30080 GOSUB VAL(ERR$)
This statement converts our string to a five digit number. GOSUB then executes the sub-

Program 1.

DIM Error: Attempt to reDIM or, DIM > 32767 or, reference out of DIMed size or not DIMed.

The error occurred at line 3345.

```
3330 DB$(LEN(DB$)+1)=CHR$(155)
3340 DB$(LEN(DB$)+1)=STR$(LVL)
3345 DB$(LEN(DB$)+1)=ANS$
3350 DB$(LEN(DB$)+1)=CHR$(155)
3360 DB$(LEN(DB$)+1)=BONUS$
```

Program 2.

```
Readers with cassette systems can
use this program by inserting a
blank tape in the 410 recorder,
and rewinding to start.  Press PLAY
and RECORD, then enter:
     LIST "C:"
and then press <RETURN> twice.  To
add this program to a program you
already have in memory, insert your
```

"Errors" tape, rewind, press PLAY, and enter:

ENTER "C:"

and press <RETURN> twice. The program will be merged with yours. Remember that the routine uses line zero, so if you have a line zero in your program, it will be replaced. Also, you may have to change any TRAP 40000 or TRAP 32768 statements to TRAP 30000.

```
0 TRAP 30000
30000 REM *******ERROR TRAP*********
30005 DIM ERR$(10),ERR1$(10)
30010 ? "{CLEAR}"
30020 STPLN1=PEEK(187):STPLN2=PEEK(186)
30030 ERR$="31"
30040 IF PEEK(195)<100 THEN ERR1$="0"
30050 IF PEEK(195)<10 THEN ERR1$="00"
30060 ERR$(LEN(ERR$)+1)=ERR1$
30070 ERR$(LEN(ERR$)+1)=STR$(PEEK(195))
30080 GOSUB VAL(ERR$)
30090 POKE 195,0
30095 STPLN=256*STPLN1+STPLN2
30100 ? "The error occurred at line ";ST
PLN;"."
30110 LIST STPLN-20,STPLN+20
30120 END
31002 ? "Not enough memory to store stat
ement  or the new variable name or to DI
M a  new string variable.":RETURN
31003 ? "A value expected to be a + inte
ger    isn't: a value expected to be in
a   specific range isn't.":RETURN
31004 ? "Too Many Variables: A maximum o
f 128  variable names is allowed.":RETUR
N
31005 ? "String Length Error: Attempted
to     store beyond the DIMensioned stri
ng   length.":RETURN
31006 ? "Out of Data Error:READ statemen
t      requires more data than supplied
by   data statement(s).":RETURN
31007 ? "Number Greater than 32767:Value
 is nota positive integer or is greater
than 32767.":RETURN
31008 ? "Input Statement Error: Attempte
d to   input a non-numeric value into a
     numeric variable.":RETURN
31009 ? "DIM Error: Attempt to reDIM or,
DIM >  32767 or, reference out of DIMed
size or not DIMed.":RETURN
31010 ? "Argument Stack Overflow: There
are toomany GOSUBs or too big an express
ion.":RETURN
31011 ? "Attempt to divide by zero or re
fer to a number >10E98 or <10E-99.":RETU
RN
31012 ? "Line Not Found: A GOSUB, GOTO o
r THEN referenced a non-existant line nu
mber.":RETURN
31013 ? "No Matching FOR Statement: Nest
ed     FOR/NEXT statements do not match
or noFOR statement.":RETURN
31014 ? "Line too long: The Statement is
 too   long or complex for basic to hand
le.":RETURN
31015 ? "A NEXT or RETURN was encountere
d and  the GOSUB or FOR has been deleted
     since the last RUN.":RETURN
31016 ? "RETURN Error: A RETURN was enco
unteredwithout a matching GOSUB.":RETURN

31017 ? "Garbage Error: Execution of bad
 RAM   bits was attempted. Usually a har
dwareor POKE problem.":RETURN
31018 ? "String does not start with a va
lid    character, or string in VAL state
ment is not a numeric.":RETURN
31019 ? "LOAD Program too Long: Insuffic
ient   RAM to complete load.":RETURN
31020 ? "Device number larger than 7 or
equal  to 0.":RETURN
31021 ? "LOAD File Error: Attempt to LOA
D a    non-LOAD file.":RETURN
31128 ? "BREAK Abort: User hit |BREAK| k
ey duringI/O operation.":RETURN
31129 ? "IOCB already open.":RETURN
31130 ? "Nonexistent device specified.":
RETURN
31131 ? "IOCB Write Only: READ command t
o a    write only device.":RETURN
31132 ? "Invalid Command: The command is
       invalid for this device.":RETURN

31133 ? "Device or File not Open: No OPE
N      specified for this device.":RETUR
N
31134 ? "Bad IOCB Number:Illegal device
number.":RETURN
31135 ? "IOCB Read Only Error: WRITE com
mand toa read-only device.":RETURN
31136 ? "EOF: End of File has been reach
ed.":RETURN
31137 ? "Truncated Record: Attempt to re
ad a   record longer than 256 characters
.":RETURN
31138 ? "Device Timeout: Device doesn't
       respond.":RETURN
31139 ? "Device NAK: Garbage at serial p
ort or bad disk drive.":RETURN
```

```
31140 ? "Serial bus input framing error.
":RETURN
31141 ? "Cursor out of range.":RETURN
31142 ? "Serial bus data frame overrun."
:RETURN
31143 ? "Serial bus data frame checksum
error.":RETURN
31144 ? "Device Done Error: (invalid 'do
ne'    byte): Attempt to write on a writ
e-   protected diskette.":RETURN
31145 ? "Read after write compare error
(disk  handler) or bad screen mode handl
er.":RETURN
31146 ? "Function not implemented in han
dler.":RETURN
31147 ? "Insufficient RAM for operating
in     selected graphics mode.":RETURN
31150 ? "Port Already Open: Attempt to O
PEN an RS-232 port already open through
     another IOCB.":RETURN
31151 ? "Concurrent I/O mode not enabled
: Aux1 bit 0 not set for XIO 40.":RETURN

31152 ? "Illegal User Supplied Buffer: B
uffer  length and/or address inconsisten
t in concurrent I/O mode.":RETURN
31153 ? "Active Concurrent I/O Error: At
temet  to perform RS-232 I/O while concu
rrentmode I/O active.":RETURN
31154 ? "Concurrent Mode not Active: Con

currentI/O mode must be activated in ord
er toperform INPUT or GET.":RETURN
31160 ? "Drive number error."
31161 ? "Too many OPEN files (no sector
buffer available)]":RETURN
31162 ? "Disk full (no free sectors)]":R
ETURN
31163 ? "Unrecoverable system data I/O e
rror.":RETURN
31164 ? "File Number Mismatch: Links on
disk   are messed up.":RETURN
31165 ? "File Name Error.":RETURN
31166 ? "POINT data length error.":RETUR
N
31167 ? "File locked.":RETURN
31168 ? "Command invalid (special operat
ion    code).":RETURN
31169 ? "Directory full (64 files).":RET
URN
31170 ? "File not found.":RETURN
31171 ? "POINT invalid.":RETURN
31172 ? "Illegal Append: DOS 1 cannot ap
pend toa DOS 2 file.":RETURN
31173 ? "Bad Sectors at Format Time:Disk
 drive found bad sectors while formattin
g a  diskette.":RETURN
```

routine of that number. Each subroutine is contained entirely by that line. The subroutine prints the description unique to that error and then RETURNS to line 30080.

Line: 30090 POKE 195,0
This statement resets the error number to zero.

Line: 30095 STPLN = 256* STPLN1 + STPLN2
Line 30095 converts the two byte "binary" line number to its decimal equivalent. (See Line 30020.)

Line: 30100 ? "The error occurred at line ";STPLN;"."
This line prints the decimal value of the error line.

Line: 30110 LIST STPLN-20, STPLN + 20
This statement prints the statements immediately before, after and including the error.

The result of such an error is presented in Program 1.

Now you say, "This system will work for errors encountered during program execution but what about errors in direct mode?" Aha! It can still save trips to the manuals. As long as 'ERRORS' is loaded in RAM, just type GOSUB 31xyz where xyz is your error number, and a description of your latest error will appear. Disregard the information about line numbers.

I have loaded "ERRORS" on to my master diskette so that it is automatically transferred to each disk along with the DOS programs whenever I create a new workdisk by duplicating the master.

The errors listed here may be changed or expanded to adapt to your hardware and software (e.g., the line printer and word processor software). A similar method may be used to trap the ASSEMBLER errors replacing BASIC errors one through nineteen.

Program 2 is the complete program. ©

BASIC Program Merges: PET And VIC

Jim Butterfield
Toronto

If you have two BASIC Programs, it's hard to consolidate them together without getting typer's cramp. The command LOAD wipes the old program as it loads in the new one. This is a disappointment. There are times when you have a group of DATA statements in a program and would like to bring them into another program which will use them for a new set of computations. The same thing is true of your favorite subroutines: it's annoying to have to type them in all over again.

Merging – true merging, that is – solves this for you. You can arrange to slip extra lines into your program as if you had typed them in at the keyboard.

VIC To PET Transfers

Merging can be used to transfer BASIC programs from VIC to PET. A merged program will occupy the proper memory addresses as it arrives into the PET. LOADing a VIC program into the PET often doesn't work.

There are potential problems in moving a program between VIC and PET. For one thing, VIC color won't show on a CBM/PET.

If the program has PEEKs and POKEs, chances are it will take quite a bit of work to fit it into the other machine. If you're lucky, they can be changed to PEEK or POKE a new set of addresses; but it's not always possible to find a one-to-one translation between VIC and PET.

If the program contains machine language – look for a SYS command or USR function – you'll probably have problems cutting it over to the new machine. Some machine language programs won't even work on all models of PET – so a move to or from VIC would be much too big a shock. And the method that I will outline below won't work on machine language programs, anyway; just pure BASIC.

Writing Out The Program

To transfer a BASIC program, we're going to write cassette tape in an unusual way. It won't be a normal program tape: instead, it will be something called an ASCII listing tape. It will take about twice as long to write, and occupy about twice as much tape ... but it will be compatible.

Here's how to write this type of tape. Type:

OPEN 1,1,1,"PROGNAME" : CMD 1 : LIST

... and as soon as you press the RETURN key, you'll be requested to PRESS PLAY AND RECORD ... Do it, and the tape will start. If you watch carefully, you may see the tape hesitating every few seconds or so. Eventually the tape will stop. When it does, type:

PRINT #1 : CLOSE 1

... and tape will move one last time. When it stops, the computer will say READY and you may rewind the tape and take it out of the drive.

You have some options on the above procedure. You may call the ASCII listing anything you like: instead of PROGNAME you can call it WHISKERS or CLOUD 9. It's a good idea to give a meaningful name to tape files; when you have fifty or more tapes sitting around you'll be very happy to get a hint as to what's on a given tape. You could (if you wished) write part of a program to tape instead of the whole thing: for example, you might type LIST 300-400 instead of just LIST in the first line.

You have quite a miraculous thing on the cassette tape. It's a program, but it's written as a data file. We could read the program as if it were data, analyze it, and do any kind of computing on it we wanted to. That's unusual: programs are programs and data are data ... they seldom mix.

Getting Ready To Bring It Back

When we recall the program from this oddly formatted tape, we will bring in the lines, one at a time, and merge them with any program already in place in the computer. It will work just as if we typed the lines: new lines will fit into the program in the correct line number sequence; and if a new line number matches an old one, the new line will replace the old one.

If we are just transporting a program from VIC to PET, we must say NEW. This means that we are merging the program with nothing. The result will be the program by itself – but properly placed in the PET.

If we want to merge the program we have saved with another program, now's the time to bring that other program into your computer. The lines from tape will mix in.

The Magic Merge

Don't try to understand it. Just do it carefully. In the following, PET4 is for 4.0 Machines, PETU is for PET Upgrade Machines (they power up with ### Commodore Basic ###) and PET1 is for the Original ROM machines.

Put your "merge tape" – the one we just wrote – into the cassette drive of the computer. Now type:

```
 VIC:  POKE 19,1 : OPEN 1
PET4:  POKE 16,1 : OPEN 1
PETU:  POKE 13,1 : OPEN 1
PET1:  POKE  3,1 : OPEN 1
```

... and when you press RETURN, you'll be asked to PRESS PLAY... Do it, and the tape will move briefly and the computer will report FOUND PROGNAME.

We're almost there, but you must follow the next instructions very, very carefully. Clear the screen, and type exactly three cursor down characters. Watch it! The cursor-down key may repeat if you hold it too long. Type the following starting on line 4 (if you've followed instructions, you must be on line 4, right?):

```
 VIC: PRINT"[home]":POKE 198,1:POKE 631,13:
      POKE 153,1
PET4: PRINT"[home]":POKE 158,1:POKE 623,13:
      POKE 175,1
PETU: PRINT"[home]":POKE 158,1:POKE 623,13:
      POKE 175,1
PET1: PRINT"[home]":POKE 525,1:POKE 527,13:
      POKE 611,1
```

The designation [home] above means: press the home key; the computer will print a reverse-S character. Don't type the letters H-O-M-E; that won't get you anywhere. After you've input the above line, press RETURN and things will suddenly get very busy. The cassette tape will start to move, and it will keep moving with brief stops for some time. There will be no sign of activity on the screen, except that the word READY may mysteriously appear above the line you typed.

Eventually the tape will stop moving and an error notice will print. It might be ?OUT OF DATA and it might be ?SYNTAX ERROR – but, in either case, ignore it: it's not a real error. To be neat, you should now type CLOSE 1.

Your program is now in the machine. You may go ahead and use it, or SAVE it in the more conventional way for future use.

How It Works

It's magic.

The basic procedure was evolved by Brad Templeton. If you want more details and happen to run across Brad, ask him. But you'd better have a week to spare.

You can merge programs together. You can transfer programs from VIC to PET (or vice versa, for that matter). But we've only just begin to tap the treasures of the Merge sequence.

For a few glorious moments, the tape unit took over control from the keyboard. Everything "typed in" from tape was executed; it just happened to be program lines in this case. We have broken the distinction between data and program files, and a world of new possibilities emerges. Programs that write programs? Programs that control the computer's other activities? They are all possible. ©

Insight: Atari

Bill Wilkinson
Optimized Systems Software
Cupertino, CA

This month has been a most hectic one. We just finished exhibiting both our new and old products at the seventh West Coast Computer Faire. (The seventh? Is that possible? I remember attending the first!) And, of course, we saw many, many, many new products for Atari Computers there. (Oh, all right, there were some for those other brands, also.) As I have said before, I won't review other companies' software products in this column, but I hope my dear editor won't object if I mention some of the more prominent new hardware products. Presumably, we will be seeing full blown reviews of these products in these pages in the future. And, since **COMPUTE!** was also there, I won't do more than just the mentions.

New Atari Peripherals

There were two companies there with add-on disk drives for the Atari: Percom Data Corporation and MPC Peripherals. It is hoped that both will be delivering double density drives by the time you read this, and the word is that we can expect double-sided, double-density very soon.

32K Byte memory cards were in abundance. And, of course, there was already Axlon's RAM-DISK. And how about a 64K card for the Atari 400? It's availble now in Germany. I'm not sure when and/or how it will appear here.

The long-awaited 24-by-80 display (24 lines of 80 characters, instead of Atari's 40 characters) was shown by BIT3 Corporation (who make a similar board for the Apple II).

And Stargate Enterprises (an Atari dealer near Pittsburgh, PA) brought and demonstrated the most innovative prototype: a small, radio-controlled robot. This might not sound exciting until you realize that the controlling end of the radio link was being driven by an Atari.

And wouldst that I could go into the software. Some of the latest arcade games have been, or are being, converted to Atari. And many of the best Apple II games will shortly appear for us, also. The best is yet to come, I believe. My aching pocketbook.

Anyway... as a consequence of all this, I simply didn't have time this month to do a fancy, full-blown program like last month's. Instead, I will just note a couple of the things I've been carrying around on spare scraps of paper before they get lost. But this won't be a short column; part five of my series on the internals of Atari BASIC is a fairly long and complex article on how variables are used and accessed and more. But first, the tidbits.

Control One Atari Screen

I am constantly amazed at the number of Atari owners (and not necessarily new owners) who are not aware that you can temporarily halt text screen output. They are forever typing LIST (for example) and then trying to hit the BREAK key at exactly the right time. For shame! You didn't read your manuals.

To temporarily pause, simply hit CONTROL-1 (hold down the CTRL key and hit the numeral 1 key). To continue, hit CONTROL-1 again. That's all there is to it.

Now, don't you feel silly? Would it help if I told you that somebody had to tell me, too?

Y Not Do It Later?

There is a minor, but terribly frustrating, bug in the Atari Assembler/Editor cartridge. There is no fix, but it is relatively easy to avoid if one is aware of it. So, if you haven't already been bitten, here is some bug repellant.

The problem has to do with using the ComPare-Y immediate instruction (CPY #xxx) when using the cartridge's debugger. One cannot always Step or Trace through such an instruction. Usually, an attempt to do so will cause the instruction to be treated as a BReaK (though I have heard tales of systems crashing).

The sort-of-a-solution is simply to avoid the instruction altogether. If possible, use CPX instead. Or try the following:

```
WAS:        NOW:
CPY #7      CPY VALUE7
            ...
            VALUE7 .BYTE 7
```

This new method eats up two more bytes of memory, but the CPY# should be a fairly rare instruction so this technique won't make a lot of difference.

Using Print Without Using

Every now and then, I see a routine listed and/or used that is supposed to simulate PRINT USING on a BASIC that doesn't have such a capability. (For those of you who don't know what PRINT USING is, suffice to say that it is a very nice tool which allows beautifully formatted numeric output.) Well, I couldn't let these routines go unchallenged, since I had also designed such a routine many years ago. So here is that routine spruced up for Atari BASIC:

```
32000 REM formatted money
32010 TRAP 32020 : DIM QNUM$(15) : TRAP 40000
```

```
32020 IF ABS(QNUM) >= 1E8 THEN QNUM$ =
      STR$(QNUM) : RETURN
32030 QNUM$ = "^$^^^^^^^^.^^^^": IF QNUM<0 THEN
      QNUM = -QNUM : QNUM$ = "($^^^^^^^^.^^)^"
32040 QNUM$(11-LEN(STR$(INT(QNUM))),10) =
      STR$(INT(QNUM))
32050 QNUM$(11,13) = STR$(100 + INT((QNUM-INT
      (QNUM))*100 + 0.5)): QNUM$(11,11) = "." :
      RETURN
```

Alternatively, you might replace the last statement of line 32030 with

```
QNUM$(14,15) = "CR"
```

NOTE: to facilitate your counting, I have used an up arrow ("^") where you should type a space.

To use the routine, simply place the number you want formatted into QNUM and GOSUB 32000. The routine returns with the formatted string in QNUM$. Some things to observe about the routine: it uses no temporary variables, it dimensions its own string (but only once; notice the TRAP), it could be easily translated to any Microsoft BASIC that allowed MID$ on the left side of the equal sign.

Inside Atari BASIC: Part V

Last month we discussed the seven main memory pointers used by Atari BASIC and BASIC A+, and I promised to make the variable table the main topic for this month. In addition, I said that we would learn how to fool BASIC in useful ways. Many of the techniques I will present this month are *not* my original ideas: I must credit many sources, including *De Re Atari* and *COMPUTE!'s First Book of Atari*. However, the material bears repeating; and perhaps I can give some deeper insight into why and how some of the tricks work.

The Structure Of The Variable Value Table

Please recall from previous articles that the variable value table (VVT) of Atari BASIC is kept distinct from the variable name table. The reason for this is to speed run-time execution. Recall that the tokenized version of a variable is simply the variable's number plus 128 (80 hex), resulting in variable tokens with values from 128 to 255 ($80 to $FF). Since each entry in the VVT is eight bytes long, the conversion from token to address within VVT is fairly simple. For those of you who are interested, the following code segment is a simplified version of the actual code as it appears in BASIC:

```
; we enter with the token value
; ($80 through $FF) in A register
;
     LDY  #0
     STY  ZTEMP+1      ;a zero page temporary
     ASL  A            ;token value * 2
                       ; but ignore the high bit
     ASL  A            ; token value * 4
     ROL  ZTEMP+1      ; carried into MSB also
     ASL  A            ; token value * 8
     ROL  ZTEMP+1      ; again, into MSByte
     CLC  ; (not needed ... included for clarity)
     ADC  VVTP         ; add in LSB of VVT Pointer
     STA  ZTEMP        ; gets LSB of pointer to var
     LDA  ZTEMP+1
     ADC  VVTP+1       ; add the two MSBs
     STA  ZTEMP+1      ; to obtain complete pointer
     LDA  (ZTEMP),Y    ; see text
```

When we exit this routine, ZTEMP has become a zero-page pointer that points to the appropriate eight-byte entry within the variable value table. But just what does it point to? The A-register contains the first byte of that entry. What is that first byte? Read on...

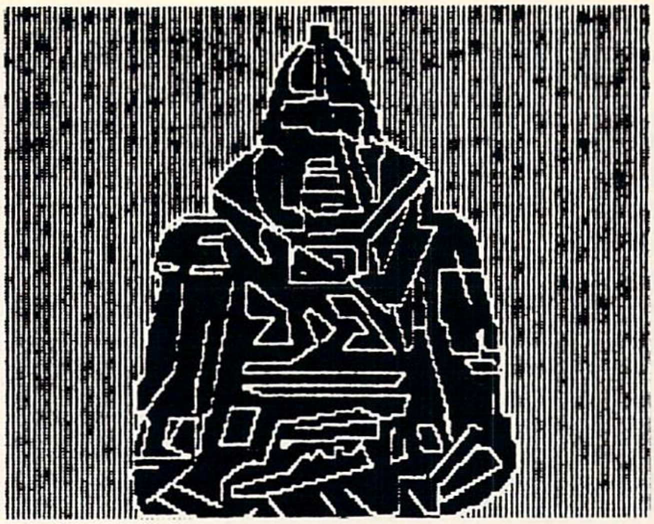
Since each entry in the VVT is eight bytes long (yet may be a simple numeric variable, a string, or an array) obviously the entries must vary in contents. However, the first two bytes always have the same meanings. In particular, the first byte is the "flags" byte, and the second byte is a repeat of the variable number (without the MSBit on). We could probably have dispensed with the repeat of the variable number; but including that byte made the entry size come out to eight bytes (more convenient), and we found several uses for it in the actual implementation.

The "flags" byte is the heart of the whole VVT scheme: until BASIC examines a variable's flag byte, it doesn't even know whether it is working with a string, array, or scalar. But note how neatly we managed to arrive at the end of the routine above with the appropriate flag byte safely in the A-register, where it can easily be checked, compared, or whatever. This, then, is the meaning of the individual bits within the flags byte:

Bit Number	Hex Value	Meaning (if bit is on)
0	$01	Array or String is DIMensioned
6	$40	this is an Array
7	$80	this is a String

Note that there is no special flag that says "this variable is a simple scalar numeric." Instead, the absence of all flags (i.e., a $00 byte) is used to indicate such variables. Since we have now used the first two bytes of each VVT entry, we now have to figure out what to do with the remaining six bytes. It is no coincidence that Atari floating point numbers consist of six bytes (a one byte exponent and a five byte mantissa): that numeric size was purposely chosen as one that gave a reasonable degree of accuracy as well as reasonable efficiency on the VVT layout. (Yes, I know, seven bytes would have worked well also, especially if we hadn't used the

FOR ATARI



FOR ATARI

The above is a graphics 8 screen printout on the EPSON, with our new AESD II(tm).
It was drawn with Versawriter graphics tablet.

★ ★

002 — ATARI EPSON SCREEN DUMP II
This is a screen dump program which allows you to copy anything from the screen. It also supports all graphics modes and text modes. It supports all the features of the EPSON(tm) MX-80 and MX-100. The program is in machine language and is relocatable. . . . . (C) $26.95
(D) $29.95

007 — DOWNLOADER
This is a true SMART TERMINAL EMULATOR PROGRAM which allows you to upload and download files between computers and save to DISK, CASSETTE, or a PRINTER. ALSO WORKS WITH THE D.C. HAY'S SMARTMODEM.
(C) $26.95
(D) $29.95

009 — ELECTRONIC CALCULATOR
This program is a tool for the electronics hobbyist. It makes the necessary resistive and capacitive calculations for both series and parallel circuits. It shows formula, decodes resistors, plus power calculations for both AC and DC circuits. . . . . (C) $19.95
(D) $21.95

011 — ELBBARCS
This is a word game program which is in high resolution graphics. . . . . (C) $19.95
(D) $21.95

012 — UTILITY PAK 1
These four utility's are for the serious programmer. XREF is a variable cross reference utility which tells you where and when a variable is used in a program. VARIABLE-CHANGER is a program that allows you to easily change the name of any or all of the variables in your program. Lister and Denumber are also included. . . . . (C) $19.95
(D) $21.95

013 — PIE BAR UTILITY
This utility is designed to provide a screen dump capability for the ATARI® GRAPH IT(tm) using EPSON® MX-80 printer. Features STORAGE and RECALL of both Pie and Bar Charts. Runs in 32K of RAM Screen Dump feature can be used separately in 24K of RAM. . . . . (C) $19.95
(D) $21.95

014 — BACKUP MASTER
A machine language program that allows you to make backup copies of boot load diskettes. Also displays any sectors that the disk drive had trouble reading and skips over them. . . . . (D) $39.95

COMPUTER AGE SOFTWARE

9433 GEORGIA AVE.
SILVER SPRINGS, MD 20910
(301) 588-6565

CASH, CERTIFIED CHECK,
MASTERCARD & VISA ACCEPTED
FOR IMMEDIATE SHIPMENT.

PERSONAL CHECKS
ALLOW 10 DAYS TO CLEAR.

redundant variable number. Oh well.)

So scalar numeric variables obviously have their value contained directly in the VVT (hence the name, variable *value* table). But what about strings and arrays, which might be any size? The answer is yet another set of pointers, etc. Before proceeding, let us examine the layout of the three kinds of VVT entries, including the already-discussed scalar type:

BYTE NUMBER	0	1	2 3	4 5	6 7
SCALARS	00	vnum	(floating point #, 6 bytes)		
STRINGS	80/81	vnum	address	LENgth	DIM
ARRAYS	40/41	vnum	address	DIM1+1	DIM2+1

For strings and arrays, byte zero (the flag byte), varies depending upon whether or not the variable has yet been DIMensioned. (Incidentally, BASIC always resets bit zero of the flag byte and zeros bytes two through seven for all variables whenever you tell it to RUN a program.)

The "address" in bytes two and three of string and array variables is *not* the actual address where the string or array is located. Instead, it is actually an offset (or, if you prefer, relative address) within the string/array space allocated to the program. Recall from last month that location $8C (140 decimal), names STARP (STring and ARray Pointer), points to the base location of such allocated space. Thus, for example, when BASIC receives a request for "ADR(XX$)", it simply uses the variable number (for XX$, which was generated when the program was typed in) to index into VVT (as above), and then retrieves the "address" from the VVT entry and adds it to the current contents of STARP.

For strings, the length and dimension values seem obvious: the DIM value is what you specify with the BASIC DIM statement, and the length is the same as that returned by the LEN function.

For arrays, we need note that DIM1 and DIM2 are as specified by the programmer in the DIM statement [e.g., DIM ARRAY(3,4)]. The reasons they are incremented by one in VVT are twofold: a zero value is used to indicate "dimension not in use" (obviously only effective for DIM2, since flag bit 0 will not be set if neither is in use) also, since the zeroeth element of an array is accessible (whereas the zeroeth character of a string is not), using DIM+1 makes out-of-range comparisons easier.

And that's it. There really are no other magic tricks or secrets. Once DIMensioned, strings and arrays don't change their offsets (relative addresses) or dimensions. There are no secret flag bits that mean funny things. Turning on the MSBit of the variable number only spells disaster. I really have told all.

Making Use Of What We Know

BASIC is not smart enough to check entries in these tables for validity. It assumes that once you have declared and/or DIMensioned a variable the VVT entry is correct (it must be...BASIC made it so). Thus the implication is that one can go change various values in VVT and BASIC will believe the changes. So let's examine what we can change and what effects (good and bad) such changes will have.

First, as usual, some cautions: BASIC DIMensions variables in the order the programmer specifies. Thus "DIM A$(100),B(10)" will ensure that the address of array B will be 100 higher than that of string A$. Neat, sweet, petite. *However*, the order in which variables appear in the VVT (and Variable Name Table) depends entirely upon the order in which the user ENTERED his program. An example:

```
NEW
20 A=0
40 DIM B$(10)
10 DIM C$(10)
30 DIM D(10
LIST
[and BASIC responds with:
  10 DIM C$(10)
  20 A=0
  30 DIM D(10)
  40 DIM B$(10)
```

Assuming that you typed in the lines above in the order indicated, the variables shown would appear in VVT in alphabetical order (A,B$,C$,D). But, if you RUN the program, the DIMensioned variables would use string/array space as follows:

```
C$,  10 bytes, offset  0 from STARP
D(), 66 bytes, offset 10 from STARP
B$,  10 bytes, offset 76 from STARP
```

Though you can figure out this correspondence (especially if you list the variable name table, with a short program in Atari BASIC or with LVAR in BASIC A+), it is probably not what you would most desire. It would be handy if the VVT order and the string/array space order were the same. Solution: (1) Place all your DIMensions first in the program, ahead of all scalar assignments. (2) LIST your program to disk or cassette, NEW, and reENTER – thus insuring that the order you see the variables in your program listing is the same order that they appear in the VVT. From here on in this article I will assume that you have taken these measures, so that variable number zero is also the first variable DIMensioned, etc.

So let's try making our first change. The simplest thing to change is STARP, the master STring/ARray Pointer. A simple program is prob-

ably the easiest way to demonstrate what we can do:

```
100 DIM A$(24*40) : A$(24*40)=CHR$(0)
110 WAIT = 900
120 A$(1,24)="THIS IS ORIGINAL A$ !!! "
130 A$(25)=A$
140 PRINT A$ : GOSUB WAIT
150 SAV140=PEEK(140) : SAV141=PEEK(141)
160 TEMP = PEEK(560)+256*PEEK(561) + 4
170 POKE 140,PEEK(TEMP) : POKE 141,PEEK
    (TEMP+1)
180 PRINT CHR$(125);
190 A$(1,11)="HI there..." : GOSUB WAIT
200 A$(12)=A$ : GOSUB WAIT
210 POKE 140,SAV140 : POKE 141,SAV141
220 PRINT A$
230 END
900 REM WAIT SUBROUTINE
910 POKE 20,0 : POKE 19,0
920 IF NOT PEEK(19) THEN 920
930 RETURN
```

BASIC A+ users might prefer to delete line 160 and change the following lines:

```
150 sav140 = dpeek(140)
170 dpoke 140,dpeek(dpeek(560)+4)
210 dpoke 140,sav140
910 dpoke 19,0
```

" 'Simple', he said. Who's he kidding!" Honest, it's simpler than it looks. Lines 100 through 140 simply initialize A$ to an identifiable, printable string and print it. The WAIT routine is simply to give you time to see what's happening. Note that A$ is DIMensioned to exactly the same size (in bytes) as screen memory. We then save BASIC's STARP value and replace it with the address of the screen (lines 150 through 170). Since A$ is the first item in string/array space, its offset is zero. Thus pointing STARP to the screen points A$ to the screen.

We then clear the screen and initialize A$ again – to a short string. Notice the effect on the screen: capital letters and symbols are jumbled because of the translation done on characters to be displayed. (Recall that Atari has three different internal codes: keyboard code, ATASCII code, and screen code. Normally we are only aware of ATASCII, since the OS ROMs do all the conversions for us.)

At line 200, we proliferate our short string throughout all of A$ – look at the effect on the screen. Finally, lines 210 through 230, we restore STARP to its original value and print what BASIC believes to be the value of A$. Surprised?

As interesting as all the above is, it is of at best limited use: moving all of string/array space at once is dangerous. In our example above, if there had been a second string DIMensioned, it would have been reaching above screen memory, into never-never land. Let me know if you can find a real use for the technique.

A better technique would be one which would allow us to adjust the addresses of individual strings (or arrays). While a little more complex, the task is certainly doable. Our first task is to find a variable's location in the VVT. If the variable number is "n", then its VVT address is [VVTP]+8*n (where "[...]" means "the contents of ...").

In BASIC:

```
PEEK(134)+256*PEEK(135)+8*n
```

or BASIC A+:

```
dpeek(134)+8*n
```

We can then add on the byte offset to the particular element we want and play our fun and games. Again, a sample program might be the best place to start:

```
100 DIM A$(1025),B$(1025) : A$(1025)=CHR$(0) :
    B$=A$
110 STARP=PEEK(140)+256*PEEK(141)
120 VVTP=PEEK(134)+256*PEEK(135)
130 CHARSET=14*4096 : REM HEX E000
140 VNUM=1 : REM the variable number of B$
150 LET NEWOFFSETB=CHARSET - STARP
160 TEMP1=INT(NEWOFFSETB/256)
170 TEMP2=NEWOFFSETB - 256*TEMP1
180 POKE VVTP+VNUM*8+2,TEMP2 : POKE
    VVTP+VNUM*8+3,TEMP1
190 A$=B$
200 PRINT ADR(B$),CHARSET
```

optionally, in BASIC A+:

```
100 dim a$(1024),b$(1024) : a$(1024)=chr$(0) : b$=a$
110 starp=dpeek(140)
120 vvtp=dpeek(134)
130 charset=14*4096
140 vnum=1
180 dpoke vvtp+vnum*8+2,charset-starp
190 a$=b$
200 print adr(b$),charset
```

equivalently:

```
100 DIM A$(1024)
110 CHARSET=14*4096
120 FOR I=1 TO 1024
130 A$(I)=CHR$(PEEK(CHARSET+I-1))
140 NEXT I
```

or again, optionally, in BASIC A+:

```
100 dim a$(1024) : a$(1024)=chr$(0)
110 move 14*4096, adr(a$), 1024
```

The intent of all four of the above program fragments is the same: to move the Atari character set font from ROM (at $E000) into the string A$. The third method will probably be the most familiar to most of you. Unfortunately, it is also the slowest. The fourth method, admittedly is clearest in BASIC A+, though: its line 110 summarizes what we are trying to do in each of the other three.

The first method is of course the one which deserves our attention since it relates to this article.

Line 100 simply allocates and initializes our two strings. We must DIMension these strings one greater than we need because of the bug in Atari BASIC which moves too few bytes when string movements involve moving exact multiples of 256 bytes. Lines 110 and 120 simply get the current values of the two pointers that we need, VVTP and STARP.

Lines 130 and 140 actually simply set up some constants. The Atari character set is always located at $E000, of course. The VNUM is set to one, in accordance to what we noted above. *Be careful!* The VNUM will *not* necessarily be one if you did not type this program in the order shown! When all else fails, use LIST and reENTER.

We use line 150 to figure out how much B$ must move (and it will always move "up," since the ROM is always above the RAM) and then calculate its new "offset" within STARP. Of course, it is now actually outside of string/array space, but BASIC doesn't know that. Why should it care?

Unfortunately, lines 160 and 170 are needed in Atari BASIC (and most other BASICs) to manipulate 16-bit numbers into digestible, byte-sized pieces.

Finally, with line 180 we establish B$ as pointing to the character set memory. Line 190 moves the entire 1025 bytes, with one simple operation, from there to the waiting arms of A$, in RAM, where it can be manipulated.

With Atari BASIC (and, indeed, with most BASICs), the only other way to get the speed demonstrated here is to write an assembly language subroutine to do the move. Obviously, if you were simply moving the character set once, this is not the way to do it. But if you are interested in manipulating a lot of different memory areas with great speed (player missile graphics? multiple screens?), this works.

A couple of comments: We did not really need to DIMension and set up B$ in our example. After all, as long as we are faking the address, why not fake the DIMension, LENgth, and flags as well? We could accomplish all that this way:

POKE VVTP+8*VNUM, 65: REM say B$ is dimensioned ($41), see above)
POKE VVTP+8*VNUM+4,1: REM 1sb of 1025 ($0401), the length
POKE VVTP+8*VNUM+5,4: REM msb of ditto
POKE VVTP+8*VNUM+6,1: REM and the DIM is the same as the len
POKE VVTP+8*VNUM+7,4:REM msb of the DIMM

Now we have fooled BASIC into thinking B$ is set up properly but we haven't actually used any memory for it. P.S.: can you think of any reasons to have two variables pointing to the same memory space? A string and an array pointing the same space? We'll discuss all that next month. ©

COMPUTE!'s Listing Conventions

Many of the programs which are listed in **COMPUTE!** use special keys (cursor control keys, color keys, etc.) To make it easy to tell *exactly* what should be typed in when copying a program into the computer, we have established the following listing conventions.

For The Atari

All the editing and cursor control characters are spelled out and surrounded by brackets in the program listings: {CLEAR} for "clear screen." Other characters, such as CTRL-T (the "ball" character) will be listed as the "normal" character, but it will be within brackets: {T}. A series of identical control characters will be indicated by a number within the brackets: {3DOWN} means type ESC CURSOR-DOWN three times; {12 R} would mean type CTRL-R twelve times. Remember to press the ESC (escape) key before each cursor control key. If you should see {ESC} itself in a program listing, you would press ESC *twice*.

Two of the control characters, {=} and {-}, should be shifted. Any reverse field text will be enclosed within vertical lines. (In other words, any time you see a vertical line within a program listing in **COMPUTE!**, press the Atari logo key {人}.)

Atari Conventions

```
{CLEAR}= SHIFT-< (Clear Screen)
{UP}= CTRL-minus (Cursor Up)
{DOWN}= CTRL-equals (Cursor Down)
{LEFT}= CTRL-plus (Cursor left)
{RIGHT}= CTRL-asterisk (Cursor right)
{BACK S}= BACK S (Back space)
{DELETE}= CTRL-DELETE (Delete character)

{DEL LINE}= SHIFT-DELETE (Delete Line)
{INSERT}= CTRL-INSERT (Insert character)

{INS LINE}= SHIFT-INSERT (Insert line)
{ESC}= ESC (ESCape key pressed twice)
{TAB}= TAB (Tab key)
{CLR TAB}= CTRL-TAB (Clear tab setting)
{SET TAB}= SHIFT-TAB (Set tab stop)
{BELL}= CTRL-2 (Ring buzzer)
```

For 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.

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*.

For The Apple

Programs listed as "Microsoft" are written for the PET/CBM, Apple, OSI, etc. Although the programs are general in nature, you may need to make a few changes for them to run correctly on your Apple. Microsoft BASIC programs written for the PET/CBM sometimes contain special cursor control characters. The following table shows equivalent Apple words. Notice that these Apple commands are *outside* quotations (and even separate from a PRINT statement). PRINT"[RVS]YOU WON" becomes INVERSE: PRINT"YOU WON":NORMAL

[CLEAR[(Clear Screen) HOME

[DOWN] (Cursor down)
Apple II +: Call -922
POKE 37,PEEK(37)+(PEEK(37)<23)

[UP] (Cursor up)
POKE 37,PEEK(37)-(PEEK(37)>0))

[LEFT] (Cursor left) PRINT CHR$(8);

[RIGHT] (Cursor right)
PRINT CHR$(21)

[RVS] (Inverse video on. Turns off automatically after a carriage return. To be safe, turn off inverse video after the print statement with NORMAL unless the PRINT statement ends with a semicolon.)
INVERSE

[OFF] (Inverse video off) NORMAL

Shifted characters can represent either graphics characters or uppercase letters. If within text, just use the non-shifted character, otherwise substitute a space. Some "generalized" programs contain a POKE such as POKE 59468,14. Omit these from the program when typing it in. One final note: you will probably want to insert a question mark or colon within an INPUT prompt. PET/CBM and many other BASICs automatically print a question mark:

INPUT "WHAT IS YOUR NAME";N$
becomes
INPUT "WHAT IS YOUR NAME?";N$

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 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}		

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}		

©

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:

January, 1981: Load PET Programs Into The Apple II, Player-Missile Graphics for Atari, The Atari DOS, The Kernel of the OSI Operating System, Fixing LOADing Problems on the PET, Spooling with the PET Disk, Expanding KIM.

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.

March, 1981: Machine Language Programming for Beginners, Getting the Most from your PET Cassette Deck, Apple and PASCAL, Flipping your Apple Disk, Designing your own Atari Character Sets, Renumber for Atari, An Atari Disassembler, Six-gun Shootout Game for OSI C1P, PET Machine Language Graphics.

April, 1981: How to be a VIC Expert, Resolving the Applesoft and Hires Graphics Memory Conflicts, Atari SuperCube, String Arrays in Atari, Memory Partition in PET, Pet Relative Files, Working with BASIC 4.0, Commodore File I/O, ROM Expansion for Commodore PET.

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.

September, 1981: The Column Calculator, What is a Modem and Why Do I Need One?, PET, Apple, Atari: On Speaking Terms, A Tape "EXEC" for Applesoft, A Self-altering Program for Apple II, Positioning P/M Graphics and Regular Graphics in Memory, An Atari BASIC Sort, Shoot, an Arcade Game for Atari, Exploring OSI's Video Routine, PET Tape Append and Renumber, All About LOADing PET Cassettes.

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.

November, 1981: SuperPet: A Preview, Japanese Micros: A First Look, Introduction to Binary Numbers, An Apple Primer, Page Flipper for Apple, An Atari Database System, A Program for Writing Programs on the Atari, Atari Textplot, OSI Relocation, The PET Speaks, Inversion Partitioning, A Personal News Service on PET, Bits, Bytes, and Basic Boole.

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.

Back issues are $3.00 each or six for $15.00. Price includes freight in the US. Outside the US add $1.00 per magazine ordered for surface postage. $3.00 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.

New Products

Three New Enhancements For Commodore Computers

Three new products that upgrade Commodore computers have been introduced from England by Small Systems Engineering, Inc. of Mountain View, California.

Called the SoftBox, HardBox, and Petspeed compiler, these enhancements bring the benefits of the widely used CP/M operating system, Winchester hard disk mass storage, multi-user capability, and high-speed BASIC compiling to all PET and CBM microcomputers.

• The SoftBox permits Commodore users to run the hundreds of CP/M compatible applications packages, as well as interface with up to four Corvus Winchester hard disk drives. RS232 interfacing capability is also included.

• The HardBox, teamed one to a computer, will allow up to 64 users to access simultaneously the same Corvus hard disk storage – up to 80 Mbytes using the Corvus Constellation multiplexer.

• The Petspeed compiler allows Commodore BASIC programs to run up to 30 times their normal interpretive speed. The software includes optimization procedures that permit faster execution than other compilers.

The SoftBox, containing a Z80-based 64K RAM board, modifies the CP/M operating system for the Commodore disk drive, using the PET or CBM computer itself as a terminal. CP/M version 2.2 software is included, and runs at 4MHz with no wait states, for rapid execution.

A proprietary SoftBox system utility called NEWSYS gives users much latitude in reconfiguring the operating system for their own requirements. Menu-driven options include disk drive, I/O, and RS232 assignment, as well as allowing the computer to emulate a Lear Siegler ADM3A, Televideo 912, or Hazeltine 1500 terminal.

Small System's HardBox device enhances the PET disk operating system (PET DOS versions 1 or 2), allowing one to four Corvus drives to emulate the Commodore floppy disk unit for up to 64 users.

Seven HardBox utilities are also included: user reconfiguration, password security, file transfer between hard disk and floppies, diagnostics, and use of a video cassette as a backup device.

The Petspeed compiler uses a four-pass algorithm that gives priority to frequently used variables, removes unnecessary code, and utilizes integer arithmetic wherever possible. In addition to its fast execution, the compiler is available with unrestricted use. Software writers may sell their compiled programs without incurring additional royalties.

The SoftBox and HardBox systems are respectively priced at $895 and $695. The Petspeed compiler lists for $350.

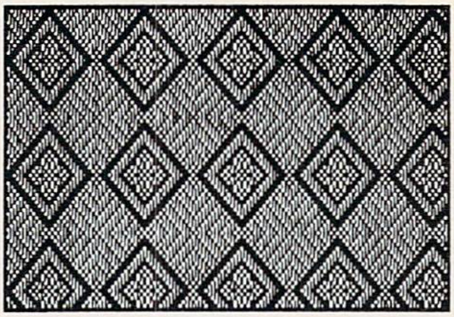
Small Systems Engineering, Inc.
71 Park Lane
Brisbane, CA 94005
(415) 468-2900

Computer Weaving From Laurel Software

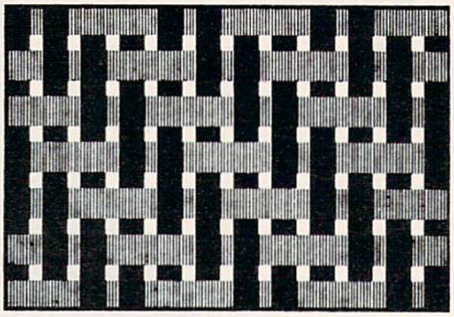
A computer weaving program, called Video Loom II, has been released by Laurel Software. With Video Loom II and an Apple II+ personal computer, a weaver can simulte a handloom design and see a high resolution color picture of the resultant textile in seconds.

The program is easy to use. No computer experience is required. The computer, via a TV screen, asks the weaver for information about how the "loom" is to be set up, what colors are to be used, yarn sizes, treadling order, etc. The weaver responds by typing answers on an ordinary typewriter keyboard.

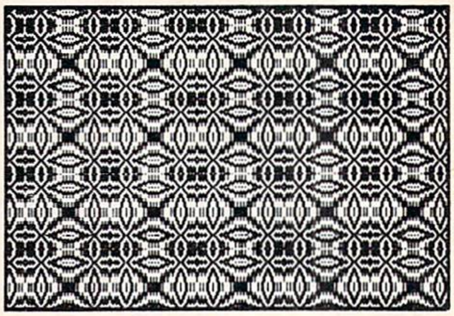
Once the loom is "warped" and the weaving sequence estab-



Sixteen Harness Diamond Twill



Englarged View of a Four Harness Twill



Four Harness Overshot Pattern

lished, the computer draws a full color television picture of the design. The designs can be saved on small, inexpensive (and reusable) magnetic diskettes. The design images can easily be photographed if color slides or polaroid pictures are desired.

The original Video Loom program, released in early 1979, was the subject of a special exhibit at the 1979 Conference of Northern California Handweavers' Guilds. At the Lawrence Hall of Science, in Berkeley, California, it was used to introduce teachers, textile hobbyists and designers to the principles of computer aided design. For more than three years it has been enabling weavers to preview their designs before going through the time consuming process of warping and creating samples.

The new version features fast machine language routines for weaving and drawing, and a foolproof screen editor for setting up the loom. This means that weaving is 20 to 30 times faster than with the original program and that designs can be changed more easily than ever.

Video Loom II simulates a 32 harness loom with 64 treadles. The loom can be instantly changed from rising to sinking shed. Designs can be displayed in up to six colors (including black and white).

In addition to choosing a threading draft, tieup and treadling, the program allows the weaver to vary warp and weft colors, thicknesses and the spacing between threads. This makes it possible to create design images that look very much like

photographs of woven textiles. Alternatively, by using very thick threads and wide spaces, a "magnified" view of a textile can be created. This feature can be used by teachers to realistically display and compare the structures of different kinds of weaves.

The program has a mode that permits the weaver to use the computer's keyboard keys as treadles. In this way, a design can be developed one weft at a time, just as if a real loom were being used.

During a weaving session, all, or only some, of the design elements can be saved on reusable, magnetic diskettes. Any design element that has been saved can be recalled at any time. This permits, for example, the tieup or treadling from one pattern to be combined with the threading draft from another, without typing in any new data.

Hardcopy printouts of the draft, tieup, treadling and all other design elements are available. If the computer is equipped with the proper accessories, a graphics printout of the design images can be obtained at the touch of a key.

Video Loom II requires a 48K Apple II+ (or an Apple II with Applesoft in ROM or a Language Card) and one disk drive. It comes on a DOS 3.3 diskette with complete, easy to follow instructions.

The cost is $59.95 plus $4.00 for postage and handling. Overseas orders should be paid in US currency and require $10.00 for postage and handling.

A brochure can be obtained by sending a stamped, self-addressed envelope.

Laurel Software
Suite 1234
1310 College Avenue
Boulder, Colorado 80302
(303) 442-6363

Atari 800 And 400 Computers Now Able To Handle Many Personal And Business Applications

A new family of intelligent interface modules from Compu-Mate will enable the Atari 800 and 400 personal computers to accomplish many additional personal and business applications which, until now, were beyond the computers' capabilities.

The first member of the new family is the Model CM-1000 printer interface. This unit includes one EIA standard serial port (standard synchronous protocol) and one 8 bit parallel port (Centronics compatible). Both of these ports are for use with standard serial or parallel printers, including letter quality.

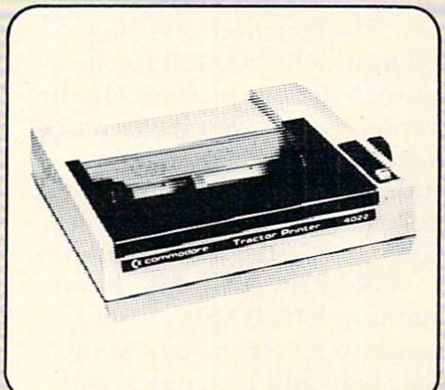
The unit comes with a simple customization program that allows the user to tailor special control codes for each port for use with many popular printers. The Atari 800 and 400 computers no longer need be limited to use with the Atari 825 or Epson MX 80 and 100 printers. Additional features include: standard baud rates from 300 to 38,400, bi-directional communications for use with modems, etc; software selectable port addressing; automatic powerup compatibility with the Atari 825 printer on the parallel port, and Diablo compatible printer on the serial port.

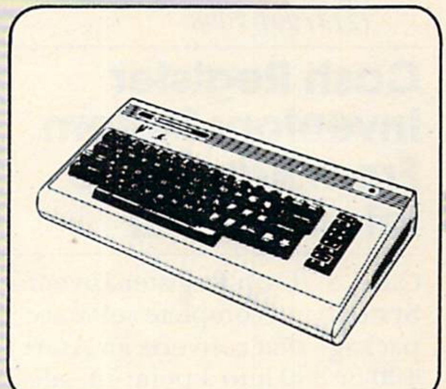
The second member is the CM-1000/V. This unit includes the CM-1000 printer interface described above. In addition, it includes an 80-column video display generator. This generator enables the 800 and 400 computers to perform many functions such as full page width word processing and other tasks requiring an 80 column instead of a 40 column display. This display generator includes many other advanced features normally found on more expensive terminals. Additional features include: connects directly to the 800 and 400 video monitor ports; software selectable 80 or 40 column display; full 96 ASCII character set; upper and lower case characters with below the line descenders; reverse video; software downloadable character set; compatible with any video display capable of 80 columns.

Features common to both

commodore
SHOW SPECTACULAR

Item	Regular	Price
8032-32K 80 COL CRT	REG $1495	$1065
64K ADD-ON MEMORY	REG $500	$395
9000 134K SUPER PET	REG $1995	$1795
4032 32K 40 COL CRT	REG $1295	$965
4016 16K 40 COL CRT	REG $995	$765
8050-DUAL DISK 950K	REG $1795	$1295
4040-DUAL DISK 343K	REG $1295	$995
2031-SINGLE DISK 170K	REG $695	$525
C2N-CASSETTE DRIVE	REG $75	$65
4022-80 COL PRINTER	REG $795	$595
8023P-136 COL PRINTER	REG $995	$849
8300P-40CPS LTR QLTY	REG $2250	$1995
8024-MANNESMAN TALLEY	REG $1995	$1595
8024L-LETTER TALLEY	REG $2495	$1995
25CPS-STARWRITER	REG $1895	$1445
CBM-IEEE MODEM	REG $279	$229
VOICE SYNTHESIZER	REG $395	$329





Item	Regular	Price
PET TO IEEE CABLE	REG $39.95	$34
IEEE TO IEEE CABLE	REG $49.95	$39
VIC 20	REG $299	$265
VIC 1540 DISK 170K	REG $599	$499
VIC 1515 30CPS PRINTER	REG $395	$349
VIC 1011 RS 232 INTER	REG $49.95	$39
VIC 1112 IEEE INTER	REG $99.95	$79

MATOR SHARK HARD DISK DRIVE

*Winchester Disk
*Commodore DOS 2.5
*24 Megabyte
$6475

Strobe 100 Plotter
*PET Interface
*Software Pack
$975

SCHOOL SPECIALS
3 for 2 Pricing
CALL US!

20% Off on Software

Special pricing on Atari, Apple, Digilog, Epson, Hayes Modem, Printwheels

PHILADELPHIA COMPUTER DISCOUNT ©

P.O. Box 170 St. Davids, PA 19087 (215)687-8540

MAIL ORDER PHONE 1-800-345-1289

PREPAID ORDERS SHIPPED FREE
VISA & MASTERCARD ADD 3%
COD - UPS
PA RESIDENTS ADD 6% SALES TAX

H1181C

interfaces include: no physical changes to computers required; both interfaces connect directly to the computers' standard serial I/O port; no cables to kludge out of the computer; both operate under DOS 2.OS and future DOS versions. One year warranty.

The CM-1000 is priced at $289.00. The CM01000/V is priced at $489.00. A kit, (Model CM-10/V) to upgrade the CM-1000 to the CM-1000/V (adds video display generator) is available for $225.00.

For additional information:

Compu-Mate Corp.
6305 Arizona Avenue
Los Angeles, CA 90045
(213) 991-7098

Cash Register Inventory System From Adventure International

C.R.I.S. (Cash Register Inventory System) is a complete software package that converts an Atari 400 or 800 into a point-of-sales terminal.

A complete inventory control system, C.R.I.S. supports up to 1000 separate inventory items. Information on backorders, total items sold, items in stock, stock ordered, and vendor numbers are all instantly accessible. The activities of up to ten salesmen can be tracked, and a user-defined commission rate individualized to each salesman is available.

The system also prints complete inventory reports, including purchase order, stock inventory, end of period, and customer sales receipts, just like the "Big Machines."

C.R.I.S. comes complete with attractive binder and user-friendly documentation, and is completely menu-driven. The 42-page manual leads the user step-by-step from initial power-up to final report generation. For the small business owner, it offers superior point-of-sale and inventory management.

Adventure International
Dept. G, Box 3435
Longwood, FL 32750
$249.95

Graphics Package For The VIC-20

Abacus Software announces the availability of GRAPHVICS which gives the VIC user both HiResolution and Multicolor display modes.

GRAPHVICS gives you two screens – one for normal text and the other for graphics display. The function keys allow you to switch back and forth between the two screens. On the graphics screen you have control over 24,000 individual points. And you can mix both hires and multicolor modes on the same screen to create graphic pictures.

GRAPHVICS adds 18 commands to VIC BASIC – commands to set colors, plot points, draw lines and rectangles, and display text on the graphics screen. The commands are simple to use and make programming in BASIC with GRAPHVICS easy.

GRAPHVICS is interactive allowing you to type the commands at the keyboard and watch the graphics immediately appear on the screen. Furthermore, GRAPHVICS allows you to save any of your screen displays to tape or disk. You can restore those saved displays for viewing at a later time.

GRAPHVICS will run on any VIC that has either a 3K or 8K expander. It comes on cassette or diskette with the user's manual and sample programs.

Price is $25 in the US and Canada, $30 elsewhere. Postage is included.

Abacus Software
P.O. Box 7211
Grand Rapids, MI 49510
(616)241-5510.

Insoft Announces GraFORTH: An Apple II Graphics Language

GraFORTH is a graphics programming language for entertainment, educational, and other graphics software creation. Designed for novices and professionals alike, GraFORTH features fast 3-D color animation graphics including rotation, scale, transposition, and perspective. Character set graphics are also included with full color, variable character sizes, upper and lower case text entry, and a block print command for easy manipulation of large shapes. Lines are drawn much faster than in BASIC and colored lines are never broken. Turtle-graphics are included to rapidly draw line shapes at any angle. In addition, GraFORTH has a software-based music synthesizer for adding music or sound effects to your programs. Music can be played in any one of several possible instrument voices.

GraFORTH is highly structured, providing easy-to-read code and programs that are fully compiled to machine language for fast execution. A 220 page tutorial manual provides complete descriptions of the program's operation.

Insoft
10175 S.W. Barbur Blvd.
Suite 202B
Portland, OR 97219
(503) 244-4181

CUE's Third Annual Conference

Computer-Using Educators will sponsor the Third Annual Fall Conference on Classroom Applications of Computers in San Jose, California on Friday and Saturday, October 1 and 2, 1982.

Based on previous years' registration figures, attendance of over 2000 persons is expected.

The schedule for Friday includes numerous workshops, field trips, school visitations, and a keynote speaker at a banquet dinner; participation in all events is by pre-registration only.

The schedule for Saturday includes over 100 commercial exhibits, as well as workshops and six hour-long sessions covering many varied computer-related classroom activities. Curriculum topics will include all levels of education from preschool through post-secondary, and will cover a broad range of applications of interest to the novice and the experienced computer user.

Computer-Using Educators is a non-profit corporation founded in 1978, with over 3000 member-educators in all but five states. More information concerning membership or the Conference can be had by writing to:

Don McKell
Conference Coordinator
Computer-Using Educators
P.O. Box 18547
San Jose, CA 95158

Computer Magazine Index Special Edition Released

Twenty-four magazines are indexed in the 1980-81 special edition of *The Periodical Guide for Computerists.*

The "Computerist" is extensively cross-referenced, easy to use and lists major articles, product reviews, editorials and miscellaneous items published in 24 computer and electronics magazines in 1980 and 1981. Over 10,000 entries are included in the latest edition of the "Computerist" which retails for $11.95.

The "Computerist" index was first published in 1976 by Berg Publications of Bothell, Washington. ACE, the new publisher, is a small publishing firm located in rural southern Oregon, using a NorthStar microcomputer to compile and generate index publications.

ACE is issuing the double 1980-81 edition of the index as its introductory edition. ACE expects to publish the index annually and to add to the number of magazines listed in future editions.

More information and copies of the 1980-81 "Periodical Guide for Computerists" at $11.95 and back issues of the "Computerist" at $5.00 each are available from:

Applegate Computer Enterprises
P.O. Box 288N
Applegate, OR 97530

Business Information Wire

Columbus, Ohio – The Canadian Press (CP), Canada's national news-gathering cooperative, announced in Toronto the availability of its Business Information Wire through the computer facilities of the CompuServe Information Service.

Since 1973, CP has provided the Business Information Wire (BIW) to Canada's major corporations, businesses and government agencies. Now, through CompuServe's videotex service, anyone from across North America can gain access to the BIW using virtually any terminal, microcomputer or communicating word processor.

News stories of interest to the business community are easily found under one or more of 18 different categories. A small monthly subscription fee allows users to read stories as they are filed on the news wire throughout the day. After 6 p.m. local time, that day's complete news report is available without additional charge to all CompuServe subscribers.

Canadian subscribers may gain access to the Business Information Wire on CompuServe through toll-free numbers in Toronto and Vancouver as well as through the TransCanada Telephone System's Datapac network across the country.

CompuServe
5000 Arlington Centre Blvd.
Columbus, OH 43220
614-457-8600

Sources For Courses From Talmis

Talmis will be publishing the first of its annual multi-indexed reference guides to microcomputer courseware for K-College education. The index will list more than 900 programs by courseware title, topic, style, grade level, running time, publisher, price and availability by computer and by format (disk, cassette, etc.). It will also indicate related print or A/V materials and basic hardware requirements. Most importantly, it will reference software reviews published in major educational or software magazines so that teachers may compare qualified critiques of programs which interest them.

For convenience, the courses will be indexed by topic, by grade level and by publisher.

The index, entitled Sources for Courses, will be available in the fall for $9.95 per copy plus shipping. Educators wishing to reserve a copy should send name and address to:

Sources for Courses
TALMIS
115 North Oak Park Avenue
Oak Park, IL 60301

Software Arts Establishes Data Interchange Format

Software Arts, Inc., the creator of the VisiCalc microcomputer program, has established a standard for exchanging data among personal computer programs by making available DIF, a data interchange format. The DIF format was originally developed by Software Arts for use with the VisiCalc program.

Software Arts has also created a DIF Clearinghouse to provide the technical information necessary for other program developers to access DIF format files with their own application programs.

The DIF format, already supported by a variety of commercially available programs, eliminates the need for time consuming re-entry and modification of data that was previously required. For example, DIF is invaluable where several different programs use the same information for financial planning, billing, and inventory. Another advantage of conforming to the DIF file format, and a major goal in its design, is the ease with which it can be used by novice programmers as well as experts.

The DIF file format has sufficient power for many applications, is easy to use and understand, and is independent from the features of any one particular computer. It also is independent from any one programming language: DIF files may be used with any language. Some of the programs currently available that can use DIF to share data with one another include VisiPlot, VisiTrend/VisiPlot, CCA Data Management System, Trend-Spotter, DB Master, and most versions of VisiCalc. Other supporting programs are planned for the future.

Two examples of uses for the DIF file format are:

1. A payroll program can use DIF to store personnel information from week to week. Then the data can be accessed by such other software as report-generating programs, without additional programming.

2. If a school keeps student records in a DIF format file, the staff and faculty can use various programs to manage and summarize information without having to re-enter the data each time.

Software Arts established the DIF Clearinghouse to coordinate and distribute information about DIF and the commercially available programs that use it. Anyone adopting DIF will have his or her program added to the list of DIF-compatible programs if they notify the Clearinghouse. Additional information about DIF, including the technical specifications and list of programs supporting DIF, is available for $6.

DIF Clearinghouse
P.O. Box 527
Cambridge, MA 02139

Universal Mailing List And New Software For The VIC 20

This mailing list will run on any Commodore 6502 machine with any memory size.

It will build a list and save it onto a cassette tape for recall. VIC20 is the machine this program was designed for, but it runs on the 40 column PET as well as the 8032. Capacity ranges from 21 names on a 3.5K VIC to 120 names on a VIC with an 8K expander, to 359 names with 32K machines. If you have a list and the 19th address in it changes you can recall just that one and change it.

The Universal Mailing List Deluxe edition comes in a three ring binder with user's manual and a cassette page that holds six cassettes.

The deluxe edition sells for $29.95. The regular edition is $24.00.

Computer Barn
319 Main St. Space #2
In the Off Main Mall
Salinas, CA 93901
(408)757-0788

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. ©

CAPUTE!: Corrections And Amplifications

1. "Using Textplot for Animated Games," April, 1982, #23, pg. 146: Because the character set is not located on a 1K boundary, any error messages will not make sense. The program will work as printed, but mistakes made during typing will not be easily understood because of the nonsense "error messages." To avoid this, you can relocate the address of the character set by these changes: Line 20 POKE 756,PEEK(106) and change line 32000 START=PEEK (106)*256.

2. "Micromon," January, 1982, #20, pg. 160: The following helpful relocating information and an associated modification to Micromon came from R. Lewsey of the Cossor Computer Club, Harlow, Essex, England.

We initially used the N command parameters in the published example but subsequently found that the code cannot relocate itself in situ – only a copy of itself made using the transfer command. The commands we eventually succeeded with after fixing the bug (see below) were:

```
.T 1000, 1FFF,2000
    Copy Micromon to $2000-$2FFF
.N 2000,2FFF,1000,1000,1FFF
    Relocate code
Relocation stops on an unrecognized code
.N 2FB0,2FFF,1000,1000,1FFFW
    Relocate word tables
```

[*To test the relocated version, first Kill the Micromon you're in (at $1000) then .G 2000 will initialize the relocated version – Ed.*]

The following changes were made which correct the N command, but slightly change the syntax of the command for word table relocation. The W at the end of the command should immediately follow the fifth address parameter; there must be no intervening space.

```
1770 A9 00      LDA #0
1772 8D 8C 02   STA $028C
1775 20 06 10   JSR $1006
1778 EA         NOP
1779 EA         NOP
```

It may not be obvious to some readers that downward relocation may also be achieved by using a suitable value for the third parameter. Subtract the required offset from $10000 (yes, four zeroes) to give the necessary parameter value (e.g. $E000 moves the code down by $2000).

I hope this information will be helpful to other Micromon enthusiasts and would like to thank Ron Cason for his assistance in sorting it all out. ©

Announcing the Printing Breakthrough of the Century: Smith-Corona® TP-1™ Text Printer

- **Low Cost Daisy Wheel Printer**

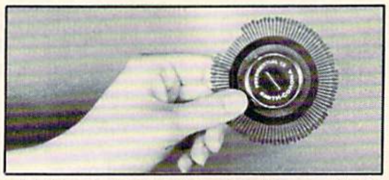
$845.00

- Microprocessor Electronics
- Serial or Parallel Interface
- Simple, Reliable Mechanism

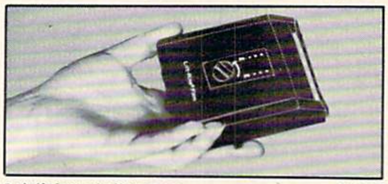
ACT NOW: Limited Supply, Low, Low Cost

Smith Corona, one of the largest manufacturers of small printers in the world, gives a whole new perspective to printing with their electronic text printer—**TP-1.** The **TP-1** is a microprocessor controlled, high quality **daisy wheel printer.** It produces perfectly formed, executive quality printouts at the speed of 120 words per minute. Typewriter quality printing at dot matrix prices.

Simple, durable and dependable, **TP-1** may be used with word processing systems, microcomputers and most small business systems. Compact and attractively styled, the **TP-1** blends well with any setting.



Additional daisy print wheels $4.95



Additional ribbons $2.95

Now, all your letters, documents forms and reports can have the crisp, professional look you demand—for business or personal use—at an affordable price. **TP-1,** the electronic text printer.

Don't delay. Order your **TP-1** TODAY at the low price of **$845.**

Micro Printer Marketing offers same day shipping, nationwide service and invites dealer inquiries. Catalogues available. No shipping charges on pre-paid orders.

Micro™ Printer Marketing

Call Micro-Printer Marketing

CALL TOLL FREE **1-800-523-9859**

MasterCard and Visa Accepted

In PA Call Collect **215 / 433-3366**

Advertisers Index

The 6502 Resource Magazine
COMPUTE! PET•ATARI•APPLE OSI•KIM•SYM•AIM

For Fastest Service, Call Our **Toll-Free** US Order Line **800-345-8112**
In Pennsylvania call 800-662-2444

My Computer Is:

- ☐ PET ☐ Apple ☐ Atari ☐ OSI
- ☐ KIM ☐ SYM ☐ AIM ☐ 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
☐ MasterCard ☐ American Express
Account No. ______ Expires ____/____

6 7 8 9 10 11 12

The 6502 Resource Magazine
COMPUTE! PET•ATARI•APPLE OSI•KIM•SYM•AIM

Foreign Readers

Subscription rates outside the US:

- ☐ $25.00 Canada FI=2
- ☐ $38.00 Europe/Air Delivery FI=3
- ☐ $48.00 Middle East, North Africa, Central America/Air Mail FI=5
- ☐ $88.00 South America, South Africa, Australasia/Air Mail FI=7
- ☐ $25.00 International Surface Mail (lengthy, unreliable delivery) FI=4,6,8

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 ____/____

6 7 8 9 10 11 12

COMPUTE! Books

Quan.	Title	Price	Total
____	**The Beginner's Guide** (Add $1.00 shipping and handling. Outside US add $4.00 air mail; $2.00 surface mail.)	$ 3.95	____
____	**COMPUTE!'s First Book of Atari** (Add $2.00 shipping and handling. Outside US add $4.00 air mail; $2.00 surface mail.)	$12.95	____
____	**Inside Atari DOS** (Add $2.00 shipping and handling. Outside US add $4.00 air mail; $2.00 surface mail.)	$19.95	____
____	**COMPUTE!'s First Book of PET/CBM** (Add $2.00 shipping and handling. Outside US add $4.00 air mail; $2.00 surface mail.)	$12.95	____
____	**Programming the PET/CBM** (Add $3.00 shipping and handling. Outside US add $9.00 air mail; $3.00 surface mail.)	$24.95	____

For Fastest Service Call Our **TOLL FREE** US Order Line **800-334-8068**
In NC call 919-275-9809

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.

6 7 8 9 10 11 12

The Editor's Feedback:

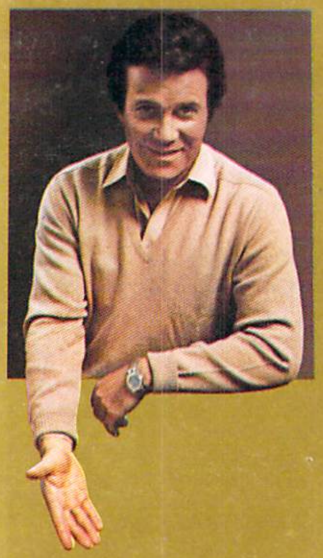
Computer: ☐ PET ☐ ATARI ☐ APPLE ☐ VIC-20 ☐ OSI ☐ OTHER ______
Are you a **COMPUTE!** Subscriber? ☐ Yes ☐ No I would like to see:

- ☐ 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?

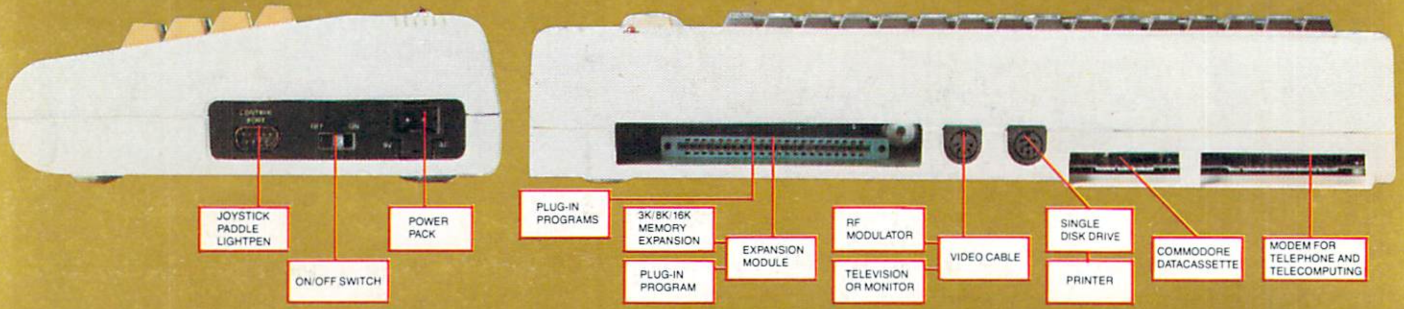
6 7 8 9 10 11 12



COMMODORE VIC-20®
"THE WONDER COMPUTER OF THE 1980s. UNDER $300."

—WILLIAM SHATNER

"The best computer value in the world today. The only computer you'll need for years to come."



VIC-20® VS. OTHER HOME COMPUTERS

Product Features	Commodore VIC-20	Atari® 400™	TI® 99/4A	TRS-80® Color Computer
Price*	$299.95	$399.00	$454.00	$399.00
Maximum RAM Memory	32K	16K	48K	32K
Keyboard Style	Full-Size Typewriter Style	Flat Plastic Membrane	Full-Size Typewriter Style	Calculator Style
Number of Keys	66	57	48	53
Programmable Function Keys	4	0	0	0
Graphic Symbols On Keyboard	62	0	0	0
Displayable Characters	512	256	192	256
Microprocessor	6502	6502	TMS9900	6809
Accessible Machine Language	YES	YES	YES	YES
Upper/Lower Case Characters	YES	YES	YES	NO
Operates with all Peripherals (Disk, Printer and Modem)	YES	NO	YES	YES
Full Screen Editor	YES	YES	YES	NO
Microsoft Basic	Standard	N/A	N/A	$ 99.00
Telephone Modem	$109.95	$399.95	$450.00	$154.95

*Manufacturer's suggested retail price Mar. 1, 1982

Read the chart and see why COMPUTE! Magazine[1] calls the VIC-20 computer "an astounding machine for the price." Why BYTE[2] raves: "...the VIC-20 computer unit is unexcelled as a low-cost consumer computer." Why Popular Mechanics[3] says "...for the price of around $300, it's the only game in town that is more than just a game." And why ON COMPUTING INC.[4] exclaims: "What is inside is an electronic marvel... if it sounds as if I'm in love with my new possession, I am."

The wonder computer of the 1980s. The VIC-20 from Commodore, world's leading manufacturer of a full range of desktop computers. See the VIC-20 at your local Commodore dealer and selected stores.

1 April '81 issue 2 May '81 issue 3 November '81 issue 4 Fall '81 issue

VIC-20
commodore COMPUTER

Commodore Computer Systems
681 Moore Rd., King of Prussia, PA 19406
Canadian Residents: Commodore Computer Systems
3370 Pharmacy Ave., Agincourt, Ont., Canada, M1W 2K4

Please send me more information on the VIC-20.

Name ____________________

Address ____________________

City __________ State ________ Zip ________

Phone ____________________