Telecommunications: How To Use A MODEM

COMPUTE!

$2.50
April, 1982
Issue 23
Vol. 4, No. 4
63379

The Journal For Progressive Computing™

The World Inside The Computer
A New Learning Column For Parents And Children

Intelligent Input Subroutines For Atari, PET And Apple

Shooting Stars
A Game For Apple, PET/CBM, Atari or OSI Computers

Using the VIC-20 Paddles: Two Games

Disk Checkout Programs For PET/CBM Disk Drives

Atari Mailing List Programs

Reviews: PET EPROM Programmer, VIXEL #1 For VIC Atari Microsoft BASIC Part II.

0 74470 63379 04

This is all you really need!

An Apple II®, a printer, a video screen, a modem, a disk drive, CPS Multifunction, and RAMPLUS+.

The CPS Multifunction card and cables are all you need to connect almost any printer, modem, or terminal to your Apple II®. It's really three cards in one. It provides the capabilities of a serial interface, parallel output interface, and real-time clock/calendar. It saves you money, power, and slots. The CPS Multifunction is compatible with DOS, CPM, PASCAL, and BASIC. Suggested retail price $239, including software. Cables $24.95 each.

The RAMPLUS+ card allows you to expand the available memory of your Apple II® to 80K. RAMPLUS+ has two 16K banks of RAM. The card is supplied with 16K of installed RAM, with an additional 16K of plug-in RAM available. It is compatible with DOS, CPM, PASCAL, and BASIC, also. Suggested retail price $189. Additional 16K RAM $24.95.

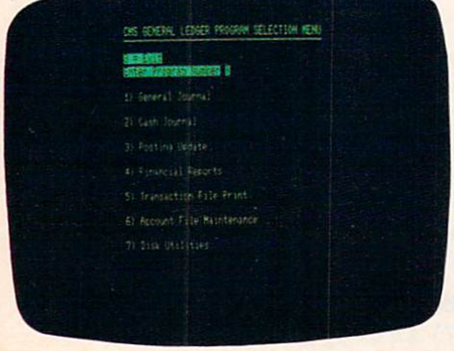
Mountain™Computer
INCORPORATED
300 El Pueblo, Scotts Valley, CA 95066
TWX: 910 598-4504 (408)438-6650

The modem is a product of Hayes Microcomputer Products, Inc.
The video screen is a product of NEC.
The printer is a product of Epson America.
The Apple II® is a product of Apple Computer, Inc.
The disk drive is a product of Apple Computer, Inc.

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.

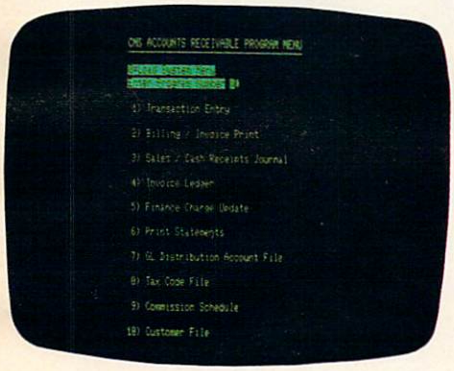


General Ledger

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.

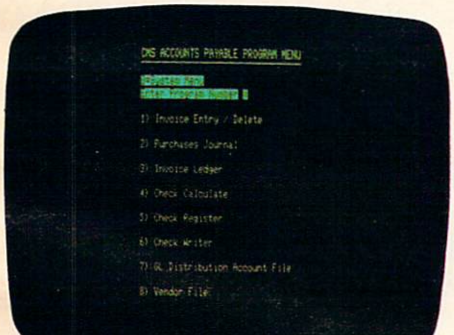
re Systems
nting System II

Series Computer System



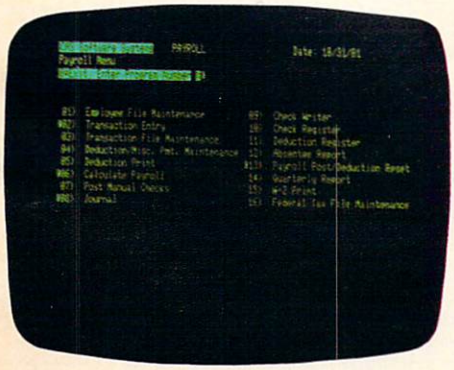
Accounts Receivable

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

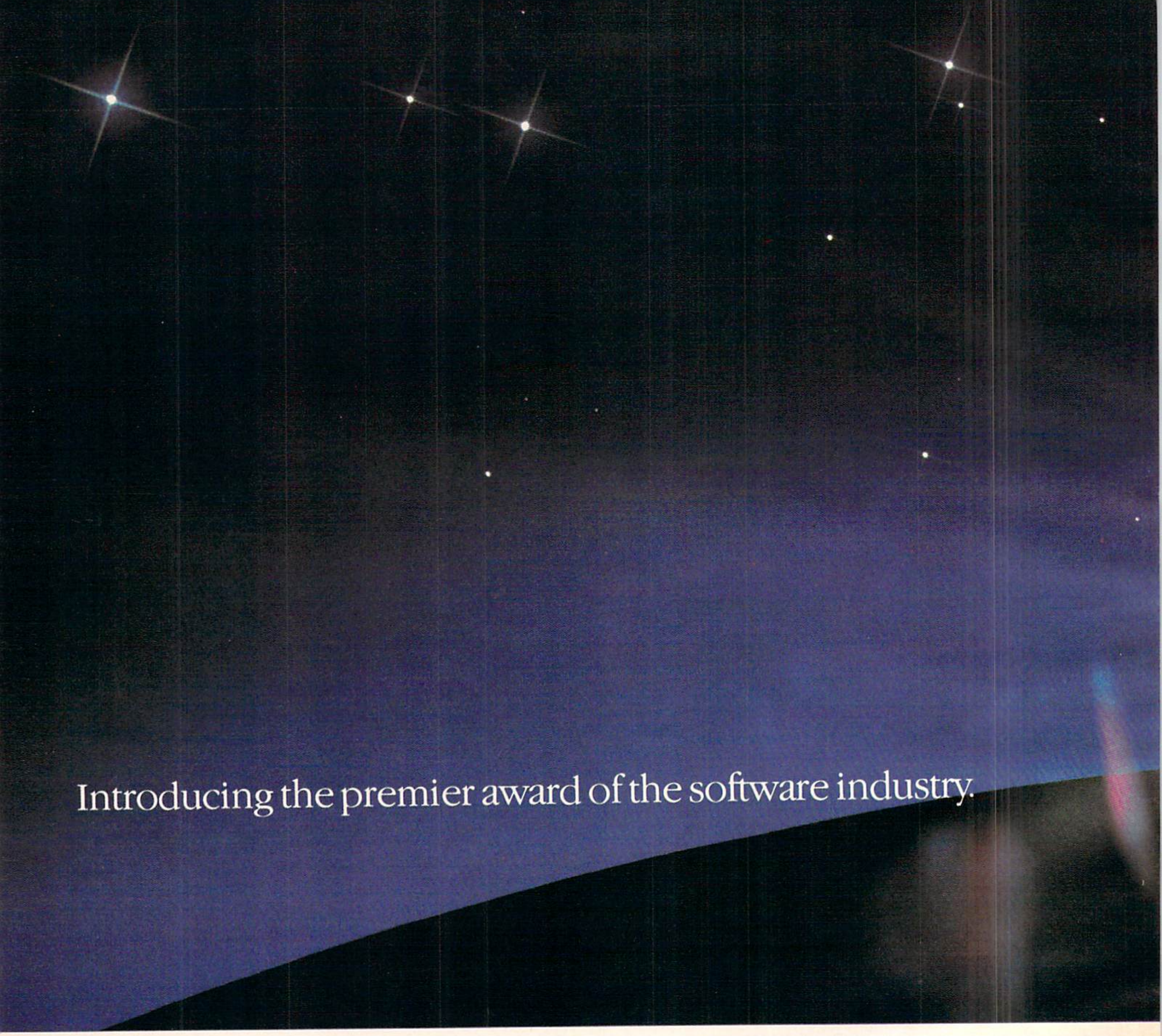

MICROSOFT
CONSUMER PRODUCTS
BASIC-80
CP/M
Z-80
apple II

Table of Contents

April, 1982 Vol. 4, No. 4

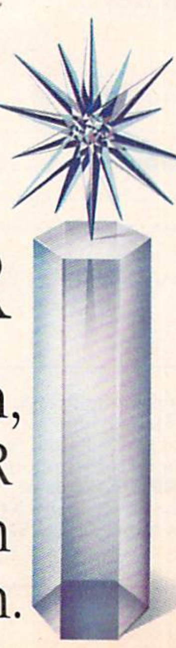
AP = Apple, **AT** = Atari, **P** = PET/CBM, **V** = VIC-20, **O** = OSI, **C** = Radio Shack Color Computer, * = All or several of the above.

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 ATARI® STAR

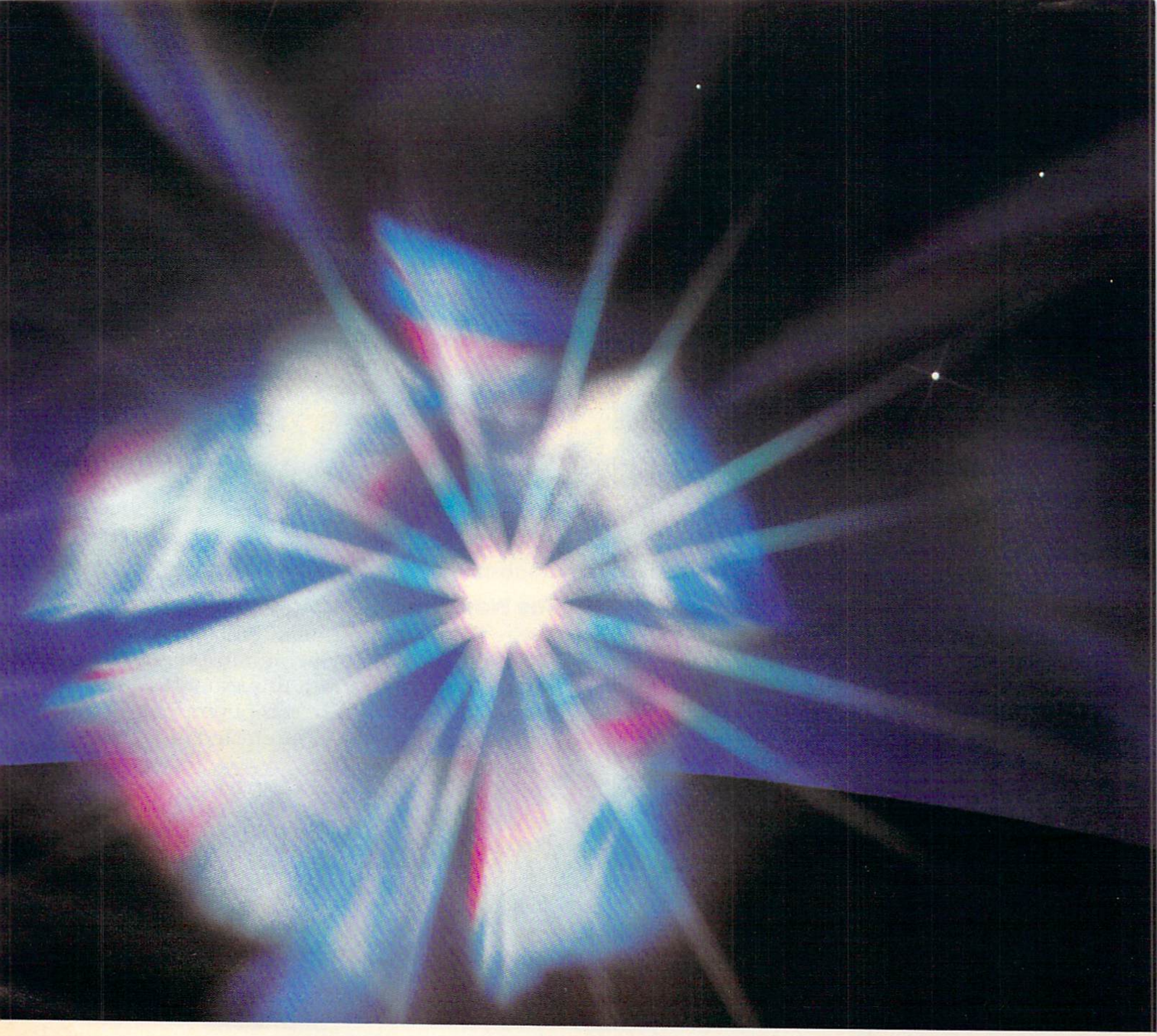
Win $25,000 in cash, plus prizes and an ATARI STAR by entering your software in the ATARI® ASAP competition.



ATARI is looking for some of the greatest software from the brightest stars in the software field. And naturally, we're willing to reward you for it.

Every three months, ATARI will be presenting ATARI STARS to the writers of software programs judged first, second and third place in the following categories: consumer (including entertainment, personal interest and development); education; home business (personal finance and record keeping); and system software. The decision of the ATARI judges is final.

Quarterly prizes will be from $750 to $3,000 worth* of selected



The Editor's notes...

Robert C. Lock
Publisher/Editor

"High" End Competition Heats Up

The Apple III and Commodore SuperPet markets continue to see increasing competitive entries. First Xerox, then IBM, and now Hewlett Packard, who last week announced a new system, the HP-XX, designed to compete for the entry level business, scientific, and upper end personal market. Apple, Inc., as we've heard frequently of late, immediately said they weren't worried. IBM promptly announced a sophisticated industrial robot (at slightly under $30,000), a new robot programming language, and guess what? The computer that sits at the heart of these robotic activities is none other than the IBM personal computer. It can control several of the robots at once, and the new language is well developed. ("Grasp" is one of its commands, for example.)

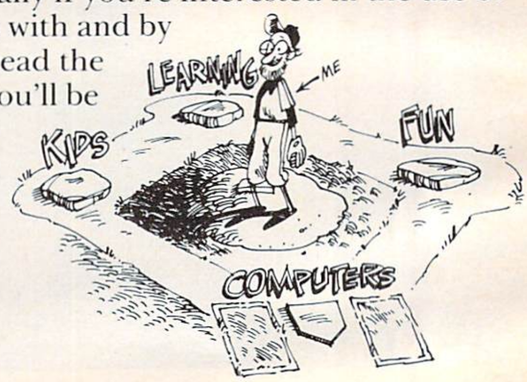
One thing's for sure. Our industry predictions of 200-300% growth this year can't be hurt at all by the increasing exposure and "big media" coverage. The world of megabusiness is finally starting to admit that there really is a future in personal computers. Witness the newsstands the second week in February... the personal computer industry was the cover feature on four different magazines. Time, Newsweek, Forbes, and Business Week all carried feature articles on our "emerging" industry.

Atari, Inc. Rumored To Be Developing The "Ultimate" In Software Protection

We hear that Atari will be investing significant research and development towards funds establishing a true protection method for their software. Suggested possibilities include a CPU-dependent encryption process. We'll be interested in seeing their progress, and will keep you posted. How big is the problem of software copying? In a recently published interview, Dan Fylstra of VisiCorp (formerly Personal Software) estimated that for every copy of Visicalc sold, two were pirated. That's a substantial impact on any company's sales.

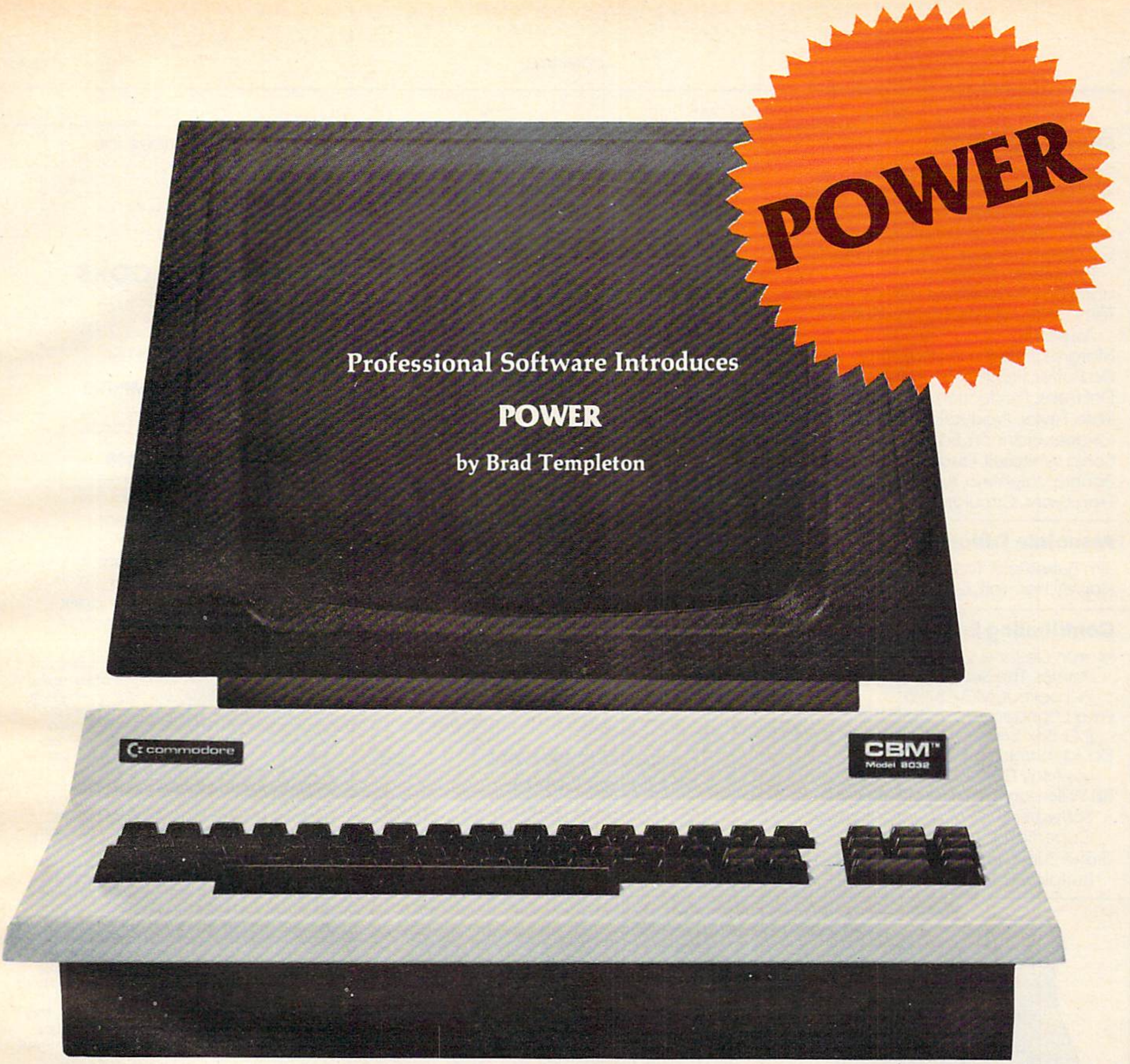
The World Inside The Computer

We're happy to welcome Fred D'Ignazio to **COMPUTE!** as a regular columnist. Fred's brand new column begins with this issue. If you're excited about the use of computers in the home, and especially if you're interested in the use of computers with and by children, read the column. You'll be intrigued.



COMPUTE!'s New Format

Your first Editor's Feedback Cards, commenting on the new format, are starting to arrive. Please keep them coming. We do use your input and suggestions in our planning of the growth and scope of the magazine. We've already refined the format a good deal more to help you identify articles of particular interest to certain machine owners. You'll see more evidence of this in this issue, and (hopefully) complete success by May. Beyond that, we're much more able to identify articles of general interest. Our staff now takes quite a few major articles each issue and generates programs for the various computers. We're certain this helps broaden the utility of the magazine to programmers and users of all machines. ©



ADD POWER™ TO YOUR COMMODORE COMPUTER

$89.95

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

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

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

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

TM POWER is a Registered Trademark of Professional Software, Inc.

Staff of COMPUTE!

Robert C. Lock, Publisher/Editor
Kathleen Martinek, Managing Editor
Richard Mansfield, Assistant Editor
Alice S. Wolfe, Publisher's Assistant
Georgia Papadopoulos, Art Director/ Production Manager
Harry Blair, Advertising Sales Manager
Terry Cash, Typesetting/Production Assistant
Margret Jackson, Typesetting
De Potter, Production Assistant
Dai Rees, Production Assistant
Kate Taylor, Production Assistant
Charles Brannon, Editorial Assistant
Sonja Whitesell, Dealer Coordinator
Bonnie Valentino, Accounting Coordinator
Fran Lyons, Circulation Coordinator

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

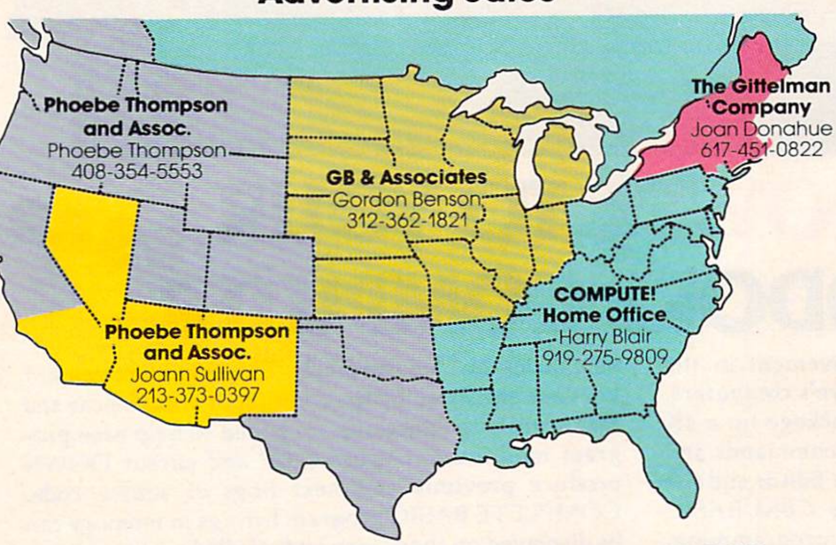
Corporate office:
625 Fulton Street,
Greensboro, NC 27403 USA

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

The Gittelman Company
Statler Office Building
Suite 518
20 Providence Street
Boston, MA 02116
JOAN DONAHUE

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 May Issue Of COMPUTE!

Managing Technostress

Getting Your Atari Disk Drive Up To Speed

Recovering From NEW On Apple And CBM

Putting The Squeeze On Your VIC: Getting The Most Out Of 5000 Bytes

Screen Input For PET/CBM

Modifying Apple's Floating Point BASIC

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

Ask The Readers

Robert Lock, Richard Mansfield, And Readers

Please address any questions or answers to: Ask The Readers, **COMPUTE!** *Magazine, P.O. Box 5406, Greensboro, NC 27403.*

Answers

"The Commodore PET has screen editing capabilities that are easy to use, due to the cursor and INST/DEL keys. The screen editor allows editing on the screen, with insert, delete, and type-over. The problem is BASIC, which doesn't like text without line numbers and PRINT" or REM preceeding the text. But there's an easy way around that. As long as RETURN is not pressed, BASIC doesn't know what you've done. There are, therefore, two solutions: at the end of a line, RIGHT CURSOR (CRSR = ›) to the beginning of the next line. Alternatively, shift-RETURN moves to the beginning of the next line without "entering" the line in BASIC.

Aha! We now have a 40 column by 24 line word processor, except there is no way to copy it. No? Yes! There's a program (published in **COMPUTE!**) [November/December, 1980, #7] *called Keyprint. You press a certain key (backslash) and – zzap! – the contents of the screen are printed out. Great! Read your printout, HOME, go back, and re-edit. The last problem is – how can you save it for later use or editing? We need a clever program (Keysave) which will save screen memory to cassette, or transfer screen memory to the start of BASIC and then save it. The program can reside in memory, since we're only using screen memory.*

Anyone out there want to try a Keysave? Such a capability would make possible an introduction to word processing, and use of the computer for writing, for many elementary schools where a very simple word processor could be used easily by students." Glenn Fisher

---

"This is in response to a question ["Ask The Readers," **COMPUTE!**, February, 1982, #21] *asked by Mr. Michael A. Ivins concerning the saving of screen data to cassette* [on the Atari]. *The following program can be used as a subroutine to do just that.*

```
0 POKE 82,0:GR. 0
10 OPEN #3,12,0,"E:":OPEN #5,4,0,"K:"
15 ? "Do you want to Make or Retrieve a picture?":
   GET #5,A: IF A = ASC("R") THEN 200
20 ?"Use the keyboard to draw": ?"Hit RETURN to
   save to cassette"
30 GET#3,A:IF A = 155 THEN 100
40 GOTO 30
99 REM SAVE TO TAPE
100 POKE 752,1 :? CHR$(126);:OPEN #4,8,0,"C:":
    REM "D:FILESPEC" FOR DISK
110 FOR Y = 0 TO 23
120 FOR X = 0 TO 39
130 LOC. X,Y,Z
140 PUT #4,Z
150 NEXT X
160 NEXT Y
170 CLOSE #3:CLOSE #4:CLOSE #5
180 END
199 REM LOAD A PICTURE FROM CASSETTE
200 ?"(ESC CTRL CLR)";:OPEN #4,4,0,"C:":REM
    "D:" FOR DISK AGAIN
205 OPEN #6,8,0,"S:"
210 FOR I = 1 TO 960
220 GET#4,A:PUT #6,A
230 NEXT I
240 CLOSE #4:CLOSE#3:CLOSE#2:CLOSE#5:
    CLOSE#6
250 GOTO 250
```

The cassette version takes only 50 seconds and the disk version is probably much faster.

Also, for all of you who have the assembler/editor cartridge there is a command to save source programs to cassette. It is LIST#C‹, to save and LOAD#C to retrieve. Just ignore the inevitable ERROR -4 that you get when loading."

"Please advise Mr. Robert Fersch [who asked about customized computer furniture in this column, **COMPUTE!**, January, 1982, #20] *that I am a manufacturer's representative and handle three lines of computer furniture. One, solid wood; another, combines wood and formica; the third, metal and formica.*

The prices are quite low, anywhere from $150 to $500, depending on material, size, storage areas.

Just drop a rough sketch showing dimensions and mention the type of material wanted and I'll send you a quote. (Wood most expensive, metal/formica, least.) Delivery is usually four weeks, all wood is six to eight weeks." Warren Modell

P.O. Box 11
East Station
Yonkers, NY 10704

"I would like to make a couple of small contributions. The first is in the form of an amendment to [The Atari version of] *the maze-generating program of Charles Bond that appeared in the December, 1981, issue (No. 19). The maze generator works perfectly as printed, but the added program to move a "mouse" through the maze apparently contained a bug as well as a mouse. I have found that a single line of code will fix it:*


NASIR
GEBELLI
PRESENTS:
PATHFINDER
PATHFINDER
Never before released, a super arcade-adventure game for Atari . . . alone in a vast transparent maze inhabited by radioactive monsters and nuclear waste, you must blast the waste and free the maze of mutant monsters. By Solitaire Group, requires 24K Atari with disk drive.
MATCH
RACER
MATCH RACER
Brand new for the Atari, this one or two player game tests your driving skills on a race track filled with treacherous obstacles. See how far you can get before your luck and skill run out! By Solitaire Group, requires 16K Atari with disk drive.
ANDROMEDA
ANDROMEDA
Introducing a spectacular, original arcade-adventure for the Atari . . . a battle in inner space . . . you must invade a living organism and destroy its tissues before its antibodies destroy you. By Solitaire Group, requires 16K Atari with disk drive.
HOLLYWOOD
SOFTWARE INC
GEBELLI
Gebelli Software Inc.,
1771 Tribute Rd., Suite A, Sacramento, CA 95815
©1981 Gebelli Software, Inc. Atari is a registered trademark of Atari, Inc.

1025 IF PEEK(A) = 84 THEN POKE A,0

This allows the program to erase the last mouse drawn, and lets the mouse turn back on its own path when it comes to a dead end.

I was given the Assembler/Editor cartridge for Christmas, and were it not for Richard Druse's article, Atari Tape Techniques, *in* **COMPUTE!**, *August, 1981, #14, I would have spent a lot more time tracking down what I thought was a serious malfunction in the cartridge. When discussing the transferral of programs to and from cassette, so that they can be run by the BASIC cartridge, the A/E manual never mentions that one must use the LIST"C" and ENTER"C" commands from BASIC. All it says is to follow the procedures in the 410 recorder manual, which never mentions these commands."* William E. L. Grossman

---

Questions

"I am planning to buy a VIC-20 computer, and rather than spend all that extra money for a printer to go with it, I would like to use the Model 35 teletype, which I already have.

The teletype printer takes an 8-bit ASCII code, ignoring the 8th bit, which the keyboard adds as a "parity check." (I'd guess the speed to be about ten characters per second.)

What I need is some kind of an interface that can plug directly into the VIC-20, and connect directly to the selector magnets on my teletype printer.

It would also be nice if the interface could take signals from the teletype keyboard contacts (altering the 8th bit of each character if necessary) and transmit back to the VIC-20 computer...this way, when I'm working with the video display, I can use the regular keyboard that is on the VIC-20; and when I'm getting a hard-copy printout through the teletype, and I need to key-in something, I can do it right there from the teletype keyboard.

Do you know anybody who can build such an interface for me, or show me how to build one myself?"

R. O. Danvers

"I'm 14 years old and own a Commodore PET 2001 with an 8K memory. I once heard of a program in **COMPUTE!** *that would enable my PET to locate programs on fast forward. I also am making a RADAR with D/A and A/D converters and any information would be greatly appreciated on these topics. Thank you."*

Matthew Silveira

This program, originally written by David Wilcox in 1978, can be adapted to any Commodore machine and can add to the speed and usefulness of a cassette-based computer system. The program is to be the first program on each tape, and lines 40 on should be updated to reflect whatever is stored on the tape. The computer then controls, and times, the cassette motor to quickly find a program's location.

The cassette is a *serial* device (where a series of items must be gone through, from item one, before a particular item can be reached). This program, however, makes the cassette a *random access* device when SAVEing or LOADing programs. It behaves the way a disk drive does, moving quickly and directly to the desired location.

```
10 PRINT" PRESS THE FAST-FORWARD K
    EY ON THE CASSETTE DRIVE N
    OW.
20 PRINT" THEN TYPE THE NUMBER OF
    THE PROGRAM YOU WISH TO US
    E.
30 PRINT" THE PROGRAMS ON THIS TAP
    E ARE:
40 PRINT" 0. FIRST PROGRAM'S NAME
50 PRINT" 1. SECOND PROGRAM'S NAME

60 PRINT" 2. ETC...
70 PRINT" 3. ETC...
80 PRINT" 4. ETC...
90 PRINT" 5. ETC...
100 GET N$:IFN$<>""THEN130
110 IF PEEK(519)=0THEN POKE 519,52:
    POKE59411,61
120 GOTO 100
130 N=ASC(LEFT$(N$,1))-48:PRINT"SEA
    RCHING FOR";N
140 POKE59411,53:TS=TI+N*600
150 IF TI<TS THEN 150
160 POKE59411,61
170 PRINT"{CLEAR} PRESS THE STOP KE
    Y ON THE CASSETTE DRIVE AN
    D THEN
180 PRINT"LOAD THE SELECTED PROGRAM
     NORMALLY.
190 PRINT"OR:
200 PRINT" SAVE A NEW PROGRAM ON TH
    IS TAPE AT THIS LOCATION
```

"The Manpower Society is intending, next September, to organize a conference on Computing Personnel Records especially Microcomputerization. As part of that process, they have asked me to survey the packages available on the market. Could I enlist your support in any of the following:

a) finding any existing Reviews of this area

b) obtaining a list of software companies who manufacture such software

c) obtaining any evaluation data available on these packages

I would be grateful for any help you or any of your colleagues could offer." P. W. Hare

---

"I enjoyed Peter Shufe's question in [**COMPUTE!**, November, 1981, #18, pg. 16, on Commodore self-modifying programming.] *I, too, have the problem of having to list out and amend DATA statements in a grading program that I use quite frequently (I'm a teacher). Unfortunately, your brief answer and program embedded therein did not work. Can you possibly expand upon this answer and give a more complete reply?"* Mel Billik

Line 100 was printed incorrectly (see below for correction). The final POKE should be a 13 (carriage return), as our answer to Mr. Shufe's question indicated.

Line 100 is the workhorse here. Whenever you have GOTO 100 within your program, the value of the variable *L* will print a line number, and *Y*, and *Z* will be the data for that DATA line. For example, if your program is keeping a running balance of your checkbook, you might only need to update the current balance. This means that you want the program to change only the one datum, perhaps called CURRENT (variable name).

To accomplish this, when the program is finished with whatever else it does, you could have a line: 700 L=10: Y=CURRENT: GOTO 100 .

Line 100 would not need any additional values so *Z* could be left out. Also, since you are not going to go on with additional DATA updates, you could make line 500 END. When you saved your program, the value of CURRENT would be in a DATA statement in line 10. When you ran the program next month, the DATA in line 10 would again be updated to reflect any changes.

The purpose of *going back into the program* (after line 100 does an update) is to allow you to update large amounts of data, not just one datum. You can use a loop to keep increasing the value of L (line number) for the DATA lines until you reach a limit you set. (This is the reason for line 500 in the original answer to Mr. Shafe).

Line 100 is unused until the end, so line one or zero should jump over it to wherever your program actually starts (1 GOTO 200).

```
100 PRINT"{CLEAR}{03 DOWN}"L"DATA"Y
    ","Z"{DOWN}L="L+2":GOTO500
    {04 UP}":POKE525,2:POKE527
    ,13:POKE528,13:END
```

"My PET 2001 did something very weird the other day. I had my CB2 speaker on in order to listen to a file which was loading (this is very helpful in detecting load errors, try it). The cassette was brand new and only had the file on it and the tape deck was recently cleaned, demagnetized, and realigned. However, instead of the usual buzzing after the file header, I heard, 'This is King's Army!' It couldn't have been more clearly spoken (much better than 'The PET Speaks' program in **COMPUTE!**, *November, 1981, #18). My PET crashed after those famous last words and the speaker emmitted a soft whine typical of a crashed PET. Should I now address my PET as King's Army? Is my PET on the verge of AI? Does the CB2 line listen in on tapes that are loading and CB's too? Now that it has talked, will it soon start to walk? What happened?"* Michael Hall

Anyone else had similar visitations? ©

Computers And Society

David D. Thornburg
Los Altos, CA

Last year I devoted two columns to the use of computer games in the classroom. Based on the mail, phone calls, and personal discussions that followed, I am convinced that this topic continues to be a sensitive one for many readers – especially those readers who are teachers.

Most of us have seen newspaper stories describing what some parents and teachers consider to be the negative influence of game arcades. Communities all over the nation are passing ordinances prohibiting or severely restricting children's access to these arcades.

One popular hypothesis is that the game arcades are responsible for truancy, increased juvenile crime, and a multitude of other assorted problems.

The fact that truant officers are often able to find children at the arcades is not surprising – but this hardly means that the arcades are the cause of truancy. To my knowledge (based partly on personal recollections of a distant past), if kids are going to skip classes, they are going to skip classes. Period. When I was a kid, you couldn't go to a soda shop when school was in session because parents and teachers felt that soda shops contributed to truancy.

Every time truants find something to do with their time, this new activity is blamed as the cause of truancy. When viewed in the context of earlier "causes of truancy" such as hanging around pool halls, drinking booze, and shooting dope, I fail to see what makes a few games of Asteroids less desirable.

If people are concerned with truancy, that's fine with me. I think that our children's education is very important. And no matter how motivated a teacher is, he or she can't teach students who don't show up for class. But if you want to find out why kids have been skipping classes, you might want to examine the two areas which seem to be more stable than the latest fad – you might want to evaluate the home and the school. All video games could be destroyed tomorrow, and I doubt that truancy statistics would show any noticeable changes.

Of course, it may be argued that the games are addicting and, for many people, they are. My piano instructor recently acquired blisters on her hands from an overdose of Pac Man. As for myself, Centipede manages to diminish my supply of quarters with great regularity.

By what magic do these microprocessor-based marvels extract billions of quarters from a public eager to fill the games' coin boxes? Let's look at what is happening when someone plays an arcade game. The first few times the player's skill level is quite low and the game ends quickly with a low score. Yet something in the game encourages continued practice and, after a while, the player's proficiency starts to increase. Once a certain level of excellence is achieved, the game provides new challenges. And the quarters keep on flowing.

The player is pursuing a self-directed course of study, and is acquiring new skills with a very high level of self-motivation. The player is actually learning something! Students who show little interest in learning new material in school are spending immense amounts of their own money to learn how to master a game.

Presumably, the student's general goal of learning is shared by teachers (although directed toward different subject areas). However, just because teachers are interested in topics other than eye-hand coordination, it isn't clear that their success couldn't be improved by discovering the reason bored, disinterested students become ardent scholars the minute school lets out and they can zip down to the local arcade to play Galaxian.

A student who is unwilling to practice repeated pen movement patterns (to improve handwriting) will gladly spend several weeks' allowance to practice repeated joystick motions to master the latest space game. Is it possible that people concerned with teaching the eye-hand coordination needed to write clearly might be able to learn something from the appeal of arcade games? It might be worth checking out.

I'm not suggesting that our schools need to become giant arcades with electronic action versions of the Peloponesian Wars, or Fraction Blaster. But classroom computers are a reality, and there are exciting games for these computers. If the students are showing us that they want to learn from games, and if they are showing us that they really *can* learn something by spending a lot of time with these machines, I think we should pay attention to this message and see if we can't make our students think of the classroom being as important a place of learning as the corner arcade.

Next month I will write about some of the most exciting educational games I have seen – including those from the Children's Television Workshop venture with Busch Gardens: Sesame Place.

Until then, let me know your feelings on this topic. ©

K-RAZY SHOOT-OUT™

For use on the *Atari 400™ or Atari 800™ Personal Computer Systems.

It's a battle of wits and fast action, this challenging game in ROM form. You're trapped in an Alien Fortress. Your mission is to clear your sector of the Alien Droids and progress to the next sector. The first sector seems easy enough...BUT–WATCH-OUT...as you progress from sector to sector, the Droids become more numerous and aggressive. A great challenge and fun too!

K-RAZY SHOOT-OUT is destined to be a favorite of Atari enthusiasts across the country with features like:

- Solid-state ROM pack
- Seven progressive levels of play
- Every game different
- Bonuses and penalties
- Full-color T.V. graphics and space-age sound
- Full-color instruction booklet
- Millions of variations
- A constant challenge

[WARNING: Playing the game could be HABIT forming]

Now at your Atari Dealer!

Coming soon:

- K-DOS™
- K-STAR PATROL™
- BUG-A-BOO™

FROM

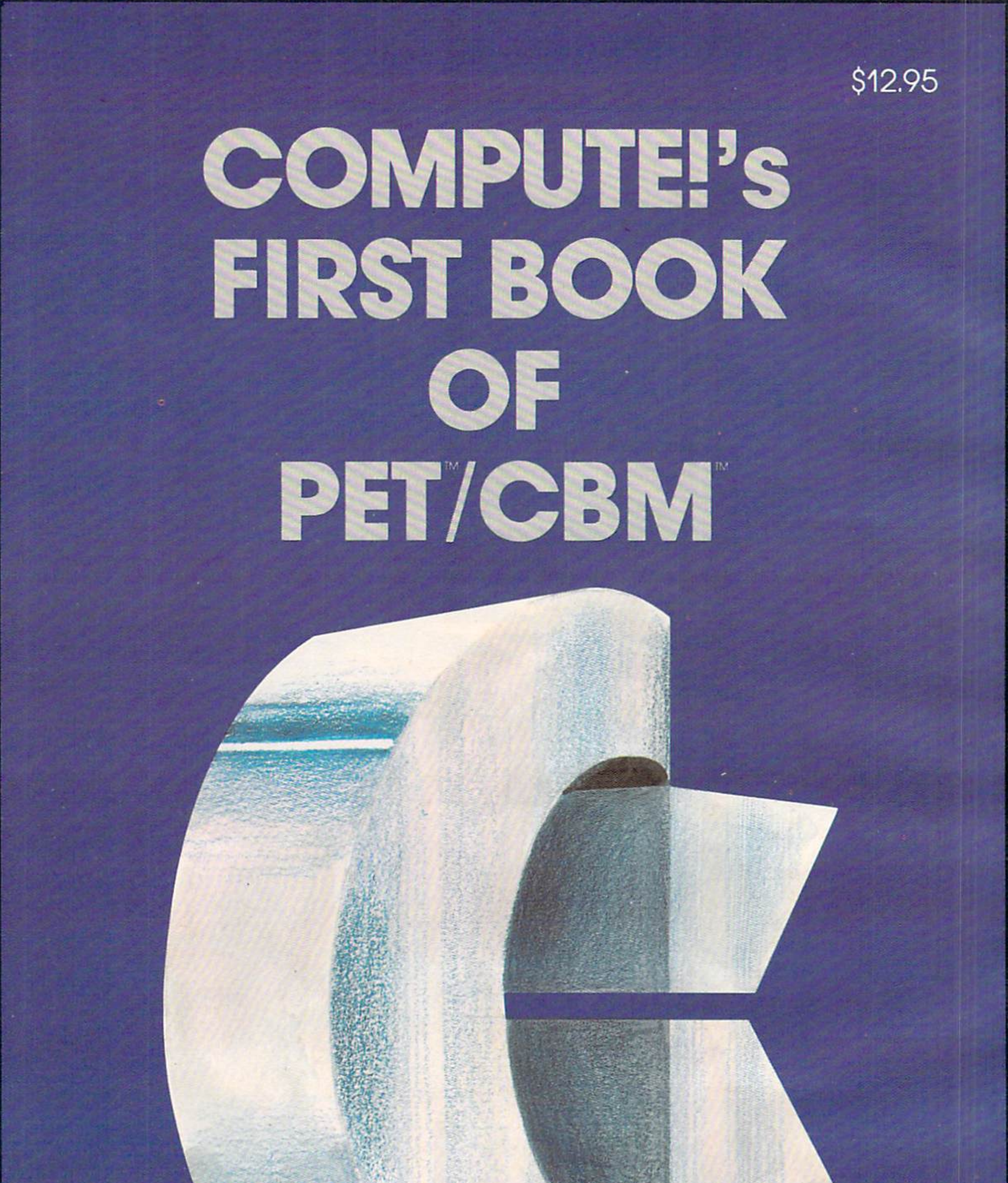
K•BYTE™

*Atari is the Registered Trademark of Atari, Inc.

FOLD & TEAR

If not available in your area, send a check or money order directly to:
K-BYTE • P.O. Box 456 • 1705 Austin Street • Troy, Michigan 48099
Toll Free Phone Orders: **1-800-521-0560** • In Michigan: **(313) 524-9878**
☐ $49.95, plus $1.50 shipping and handling, Michigan residents add 4% sales tax.
☐ $1.00 for instruction book only.

NAME ________________
ADDRESS ________________
CITY ________ **STATE** ________ **ZIP** ________
Please allow 3 weeks for delivery.
☐ **MasterCard**
☐ **VISA**
CHARGE CARD NO. ________________
EXPIRATION DATE ________________



Available Now
At Your Computer Retailer

The Beginner's Page

Making Decisions

Richard Mansfield
Assistant Editor

A major distinction between a computer and an ordinary pocket calculator is that the computer can make decisions. It can be programmed to examine something, to test it, and then to react appropriately to the test results. To see how this works, we'll write a program which gives the computer the personality (and the powers) of Madame Mona, a fortune teller.

First, let's see what is available to us. BASIC has only two decision-making words, IF and ON, but that's all we need. IF decides between two alternatives; ON chooses between several. This is like the difference between two types of tests given in school: true-false or multiple-choice.

When A Decision Is Either/Or

When you write a program, you make a list of things for the computer to do and the list is numbered so it knows in what order you want these commands carried out:

```
10 DIM Q$(200): REM THIS IS FOR ATARI BASIC
   ONLY
100 PRINT "HELLO, MY NAME IS MADAME
    MONA"
110 PRINT "WELCOME TO MY CHAMBERS..."
120 PRINT "HOW OLD ARE YOU ?"
130 INPUT AGE
```

So far, the computer is simply introducing itself and getting your age. It will always start out with these same actions whenever this program is run. It hasn't made any *decisions* yet.

To make decisions, the computer must have some information to work with. Now it has your age, so it can make a decision based on that:

```
140 IF AGE > 17 THEN GOTO 170
150 PRINT "STATE LAW DOES NOT PERMIT
    ME TO"
160 PRINT "TELL THE FORTUNE OF ANYONE
    UNDER 18.": END
```

The IF...THEN... in line 140 decides between two alternatives: whether to continue on with the program or to end it. Whatever you put after the THEN will happen *only if* the item between IF and THEN is *true*.

How does the computer decide what's true and what's false? It tests the item between IF and THEN and leaves itself a *flag* with a zero or a one "written on it" (on PET/CBM computers, the flag is zero or -1). When it finishes, THEN takes over and looks at the flag. If the flag is a zero, the computer ignores anything after the THEN on the line and goes to the next line (line 150 in this case). Should the flag be *true*, (1, or, on Commodore computers, -1) THEN performs the tasks listed after it *before going on to the next line number*.

You can try out these various "truth tests" directly, without even writing a program. Just type: PRINT 5 = 6. This is obviously not true, so your computer should print 0. Type: ? (shorthand for PRINT) 5 > 6. This is a proposal to the computer that five is greater than six. We'll get to the group of symbols called *relational* (because they show a relation between two numbers) in a minute.

For now, just notice that the computer again says this is false by printing a zero. Type: ? 5 < 6 and you get a *true* response because five *is* less than six. (An easy way to remember the *less-than* < and *more-than* > symbols is to look at the symbols themselves. They look like what they mean. The wide open side of the symbols points to the larger (more-than) number and the closed, pointed side points to the smaller (less-than) number.)

In our example, the item between IF and THEN in line 140 says that AGE is greater than 17. If someone answers the age question with the number 19, the item is true and the computer will GOTO 170. If AGE is 14, the item is false, so the instructions following THEN are ignored and the program moves on to lines 150 and 160.

IF...THEN decisions are made using the following symbols:

= Equals
< Less than
<= Less than or equals
> More than
>= More than or equals
<> Does not equal

You can also use the "logic" symbols (AND and OR) to make IF...THEN tests. They allow the computer to test *several items at the same time* between the IF and the THEN. We could change the program above by typing in three lines differently. In the new version, Mona's ethical position becomes somewhat clearer:

```
120 PRINT "HOW OLD ARE YOU ?" : INPUT AGE
130 PRINT "HOW MUCH MONEY DID YOU
```

```
    BRING MONA ?" : INPUT CASH
140 IF AGE > 17 OR CASH > = 10 THEN GOTO 170
```

In this version, as before, line 140 allows the program to continue if the customer is over 17, but it also takes into account any youngsters with ten dollars or more. Here, either age *or* money can influence Mona's decision. You could type this in directly, too: ? 5 = 6 OR 6 = 6 will give you a *true*, but ? 5 = 6 AND 6 = 6 will be *false* because AND expects them *both* to be correct (OR only needs one of them to be correct).

Selecting Between Several Possibilities

ON...GOTO (or GOSUB) does not test or create flags. It simply expects a number and sends the program off to the lines which follow the GOTO. ON X GOTO 100, 200, 375, 400 means that if X is one, the program will jump to line 100, if X is three, the program starts following the instructions listed on line 375. You can have as many line numbers after X as will fit on one line. If, by some mistake, X is beyond the range of line numbers listed after the GOTO (99447 or 0 or a negative number), the program will just continue on without jumping to any of the lines listed.

```
170 PRINT " ASK MADAME A QUESTION THAT
    CAN"
180 PRINT " BE ANSWERED BY YES OR NO..."
190 INPUT Q$
200 X = INT (RND(1) * 5) + 1
210 ON X GOSUB 230, 240, 250, 260, 270
220 PRINT: GOTO 170
230 PRINT " NO ": RETURN
240 PRINT " YES ": RETURN
250 PRINT " MAYBE ": RETURN
260 PRINT " IT IS FAIRLY LIKELY ": RETURN
270 PRINT "MONA DOESN'T KNOW. ASK ME
    LATER ": RETURN
```

We ask for a random number between 1 and 5 in line 200 which will determine what GOSUB takes effect in line 210. We could have written a series of IF...THEN lines:

```
210 IF X = 1 THEN GOSUB 230
211 IF X = 2 THEN GOSUB 240
```

but the ON structure makes things simpler. We can just create a list of choices and let the computer select from them using ON.

To make this program more interesting, you might want to add to Mona's repertoire of answers. Simply put some more lines in and then add them to the list on line 210. Don't forget to change the number five in line 200 to equal the number of answers you've created. RND will then give a random number between one and the total number of possible answers. ©

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 {⅄}.)

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

[HOME] (Home cursor) VTAB 0:HTAB 0

[DOWN] (Cursor down)
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)
POKE 36,PEEK(36) + (PEEK(36)>(PEEK(32) + PEEK(33)))

[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}		

©

The Microsoft Version of this program will work on Apple, OSI, PET/CBM, and any Microsoft BASIC computer which has enough memory to absorb the dimensioning in line 40. Changes for the Atari are in Program 2.

Moving Averages

Jerry W. O'Dell
Eastern Michigan University

One of the nice things about having a microcomputer in your business is the fact that you can do calculations easily that would be almost impossible to do by hand. A good example of this is the *moving average*. They are commonly used in businesses (we'll see why in a minute), and yet they are a perfect nightmare to calculate by hand. I'll bet that it would take a whole day to do one, and then it probably wouldn't be right.

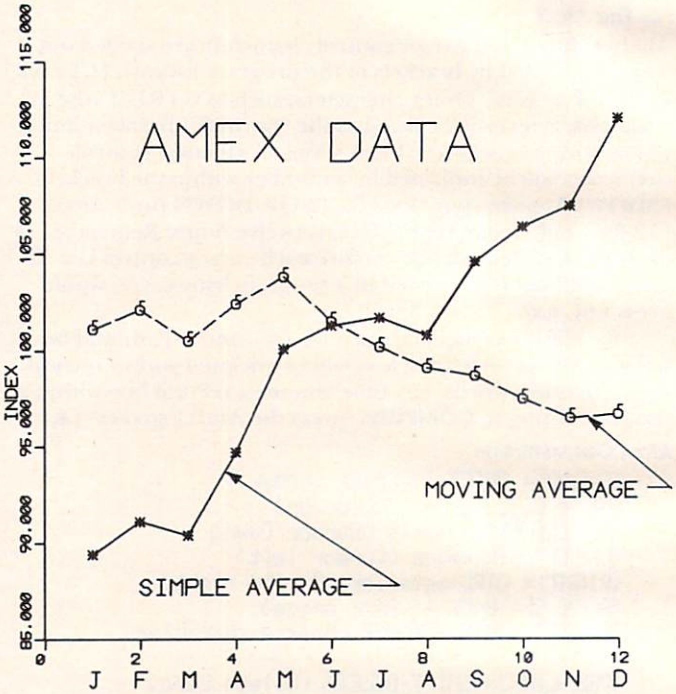
What is the moving average? This is best explained by using some data. Table 1 contains American Stock Exchange indices, by month, for the boom years of 1963 to 1968. The problem involves whether there are systematic monthly variations in AMEX prices. There are theories in the stock market that prices go up during certain times of the year. Is there any truth to this?

In Table 1, the *simple average* is just the plain old average that everyone knows how to figure, for each month. The row, "% simple average," is calculated by dividing the simple average values by the overall average (15.35) and multiplying by 100. These percentages are plotted in Figure 1 as the "simple average" values. The plot was done by an expensive CalComp plotter on a large computer (but you could do as well with a MIPLOT, I'll bet).

Table 1.

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1963	7.5	7.6	7.5	7.7	8.5	8.6	8.6	8.6	9.0	9·1	9.0	8.5
1964	8.9	9.0	9.2	9.2	9.1	9.2	9.7	9.7	10.2	10.3	10.4	10.2
1965	11.0	11.5	11.7	12.0	12.1	11.5	11.0	11.5	12.0	12.5	13.5	14.5
1966	15.0	16.0	15.9	16.5	16.3	15.5	15.4	14.0	13.0	12.5	12.6	13.0
1967	14.9	16.0	17.0	17.8	19.0	19·0	20.5	22.5	22.6	22.8	22.0	24.0
1968	25.0	23.8	22.0	24.0	27.2	29.5	28.5	26.5	29.5	30.8	31.5	33.0
Simple Average	13.7	14.0	13.9	14.5	15.4	15.6	15.6	15.5	16.1	16.5	17.2	
% Simple Average	89.4	91.1	90.4	94.7	100.1	101.3	101.7	100.8	104.6	106.4	$07.5	112.1
% Moving Average	101.1	102.1	100.5	102.4	103.8	101.5	100.2	99.1	98.7	97.5	96.5	96.7

Figure 1.



Look at the figure and the values in the "% simple average" row. Notice that they keep going up throughout the year. Aha! We can conclude from the AMEX data that stock prices gradually go up during the year. If we buy stocks in January, then we can make a killing of about 20% during the year. It sounds too good to be true. What is going on?

The confusing factor is simply the fact that the prices increased *continuously* throughout the period; it's called inflation! Oh, there are a few places in

.. 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.

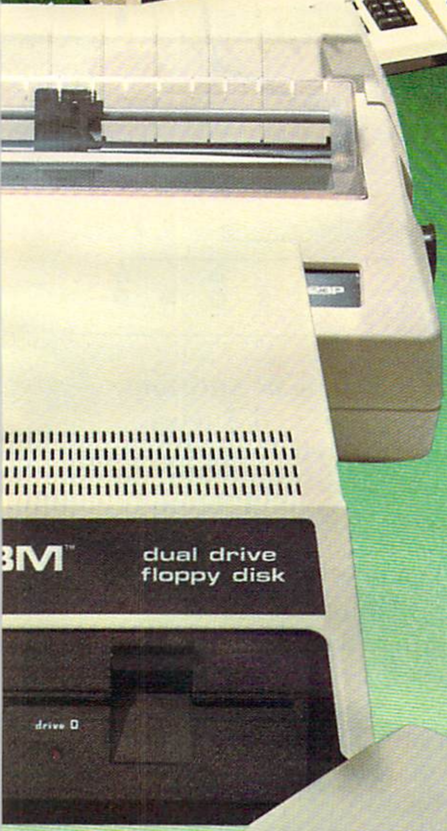
Mupet works hard, now priced from $750.00.

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

which prices went back for a month, but that's rare. If a constant factor, such as inflation, gradually and relentlessly pushes prices up, it will show up in these simple averages. Statisticians call such factors *secular trends*, but you can see how they work without all the fancy terminology.

Eliminating Secular Trends

What is needed is some way of getting rid of the secular trend. There are many ways of doing this, and many variations on the many ways, but perhaps the most widely used method is called the *moving average*. The moving average is the other line in Figure 1, and it removes the trend. Notice here that, instead of stock prices going up during the year, with the moving average method, we find that prices go down a bit. They don't go down a lot, only from about 102% to 97%, but the real direction, with the *trend* removed, is slightly downward.

In short, moving averages lead to precisely the opposite conslusion in this case. You shouldn't buy in January and sell in December. With moving averages, you'd buy in December, and sell in May.

Moving averages are useful. Now, how do you calculate them? That is a complicated business! What you do is add up the twelve months surrounding each month, and get an average for each month (for each year) from that. The trouble with that is that this gives you halfway month sums (June-July, say). So then you have to average those to get back to whole months. And on and on. Obviously, it's too complicated to explain in full here. If you want details, look at McElroy's *Applied Business Statistics* (Chapter 13), or Storkton and Clark's *Introduction to Business and Economic Statistics*. The explanation takes pages.

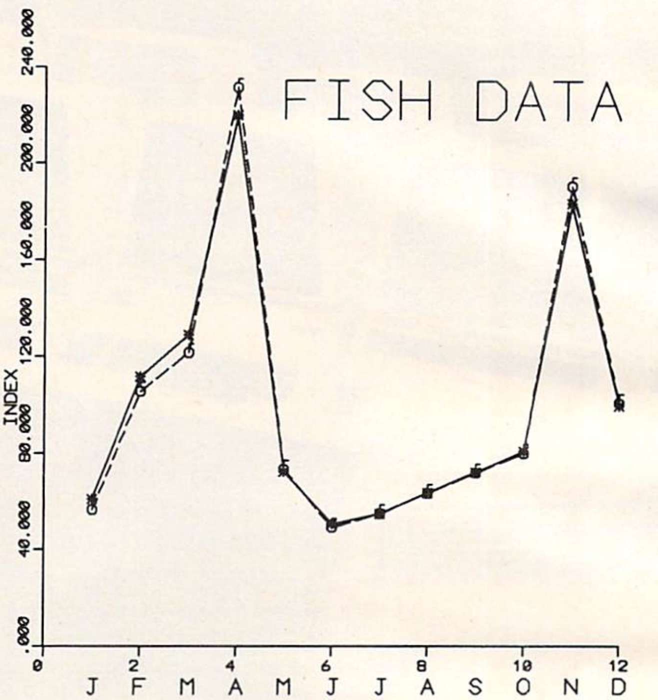
With your microcomputer you can simply type in Program 1, and it will do all the work for you. The listing has comments for those interested in how the program works, but I'd leave them out. They'll only slow down the program and, after all, you have a copy (this one!) of the comments.

The data is put in, by year, in the DATA statements at the bottom. And you have to put in the number of years, in line 30. Then the program takes over and is finished in about 30 seconds.

Are Moving Averages Always Reliable?

Do moving averages *always* smooth out data? Not if there is a real monthly variation. Take, for example, Figure 2, which is a plot of the quantity of fish taken (in millions of pounds) from Lake Michigan in the years 1952-1957. One curve is the simple average, and the other curve is the moving average. Notice here that the two curves are almost identical, so much so that I didn't try to identify them. Here, there really is a seasonal variation, and it shows up even when the long term trend is removed. It's

Figure 2.



obvious that you should go fishing in April and November, and avoid June, July, and August, when most people, of course, go fishing (data from Michigan Department of Conservation).

You can do moving averages in about 30 seconds that would take many hours to do by hand. And, if you're making business decisions, you'd better do so, if you suspect monthly trends.

A few warnings are in order. There are all sorts of moving averages: you can do them monthly, quarterly, and I think that in the stock market they do them at 29 day intervals. The program can be changed to do that, if you change the indices. And there are variations in the way people calculate them. Some people use medians, some drop extreme values, and so on.

The program has been carefully tested. There is one potential problem, however, that might crop up on some other computers. Some of the subscripts on some of the variables may go out of range occasionally, and refer to values that don't exist. I like my programs to be as simple as possible, and the BASICs that I've tried are very forgiving in that way. However, perhaps somewhere someone has a BASIC that won't be that nice, and you might have to add some statements to fix that; it would make the program a good deal longer, so I didn't. As listed the program will handle 20 years. If you want more years, you'll have to change the dimension statements in line 40 (12 times the number of years desired).

Program 1: Microsoft Version

```
10 REM MOVING AVERAGE PROGRAM
20 REM CHANGE N TO SUIT N OF YEARS
30 N=6
40 DIM IN(240),SM(240),Y(240),Z(12
   ),MO$(12)
50 DATA "JANUARY  ","FEBRUARY ","M
   ARCH    ","APRIL    "
60 DATA "MAY      ","JUNE     ","J
   ULY     ","AUGUST   "
70 DATA "SEPTEMBER","OCTOBER  ","N
   OVEMBER ","DECEMBER "
80 FOR I=1 TO 12
90 READ MO$(I) :REM READ PRINTOUT ~
   HDRS
100 NEXT I
110 T=N*12  :REM TOTAL N OF MONTHS
120 KV=0  :REM COUNTER FOR MONTHS
130 FOR I=1 TO N :REM LOOP TO READ ~
    YRS
140 FOR J=1 TO 12  :REM LOOP TO RD ~
    MOS
150 KV=KV+1
160 READ IN(KV)  :REM READ IN DATA
170 NEXT J
180 NEXT I
190 KB=5      :REM ST COUNT B AT JUNE
200 KM=T-12
210 FOR I=1 TO N :REM BIG LOOP FOR ~
    HWY
220 FOR J=1 TO 12  :REM LOOP FOR MO
    S
230 KB=KB+1
240 KC=KB-6     :REM INDEX FOR WHOLE
     YR
250 FOR K=1 TO 12
260 KC=KC+1
270 SM(KB)=SM(KB)+IN(KC) :REM HF MO
     SUMS
280 NEXT K
290 NEXT J
300 NEXT I
310 FOR J=6 TO (T-7) :REM RANGES FOR 4
320 Y(J+1)=(IN(J+1)/((SM(J)+SM(J+1)
    )/24))*100
330 NEXT J
340 FOR I=1 TO 12 :REM MAIN SUM LP
350 KQ=I+6
360 IF KQ<13 THEN 380
370 KQ=KQ-12
380 KR=KQ
390 FOR J=1 TO N :REM LOOP FOR YEAR
    S
400 Z(KQ)=Z(KQ)+(Y(KR)/(N-1))
410 KR=KR +12
420 NEXT J
430 G=G+Z(KQ)
440 NEXT I
450 CR=1200/G
460 PRINT TAB(1);"MONTH";TAB(17);"A
    VERAGE"
470 FOR I=1 TO 12
480 PRINT MO$(I),Z(I)*CR  :REM ANSW
    ERS
490 NEXT I
500 PRINT
8000 DATA  7.5, 7.6, 7.5, 7.7, 8.5, ~
     8.6
8005 DATA  8.6, 8.6, 9.0, 9.1, 9.0, ~
     8.5
8010 DATA  8.9, 9.0, 9.2, 9.2, 9.1, ~
     9.2
8015 DATA  9.7, 9.7,10.2,10.3,10.4,1
     0.2
8020 DATA 11.0,11.5,11.7,12.0,12.1,1
     1.5
8025 DATA 11.0,11.5,12.0,12.5,13.5,1
     4.5
8030 DATA 15.0,16.0,15.9,16.5,16.3,1
     5.5
8035 DATA 15.4,14.0,13.0,12.5,12.6,1
     3.0
8040 DATA 14.9,16.0,17.0,17.8,19.0,1
     9.0
8045 DATA 20.5,22.5,22.6,22.8,22.0,2
     4.0
8050 DATA 25.0,23.8,22.0,24.0,27.2,2
     9.5
8055 DATA 28.5,26.5,29.5,30.8,31.5,3
     3.0
```

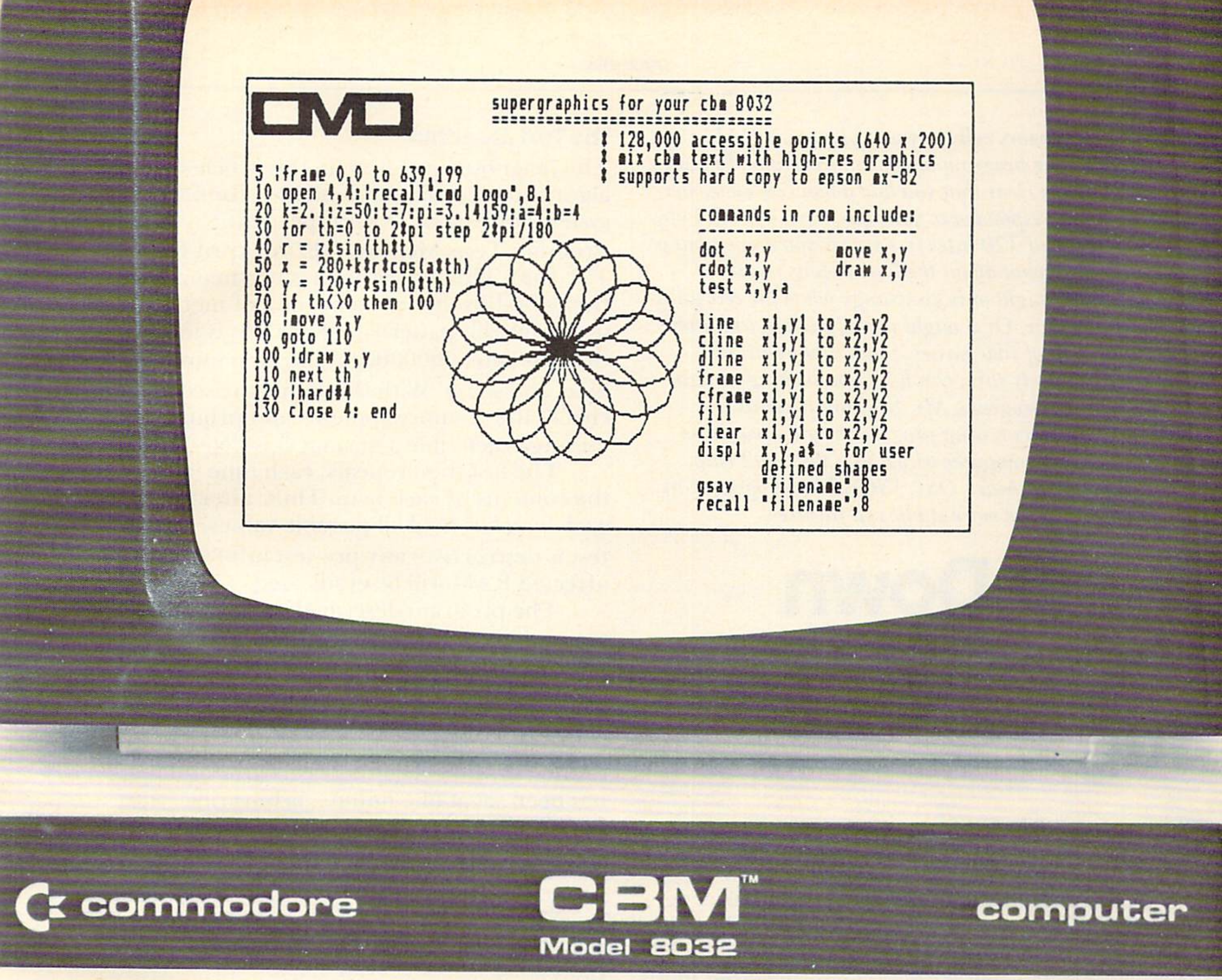
Program 2: Atari Version

Minor changes were necessary to convert this program to run in Atari BASIC, such as string array simulation for the reading of the twelve months of the year. Because Mr. O'Dell has the months padded to the right with spaces, this is easy. Note the period at the end of each month in lines 50-80. This is necessary to preserve the trailing blanks in some of the items. The period makes the string ten characters long, but, because T$ was only DIMensioned for nine characters, that's all that is read in. The POKE 85,n simulates a TAB statement. — The Editors

Make these changes to the Microsoft version:

```
40 DIM IN(240),SM(240),Y(240),Z(12),MONT
H$(9*12),T$(9)
50 DATA JANUARY  .,FEBRUARY .,MARCH    .
,APRIL    .
60 DATA MAY      .,JUNE     .,JULY     .
,AUGUST   .
70 DATA SEPTEMBER.,OCTOBER  .,NOVEMBER .
,DECEMBER .
80 FOR I=1 TO 12
90 READ T$:MONTH$(I*9-8,I*9)=T$
100 NEXT I
160 READ TEMP:IN(V)=TEMP
460 PRINT " MONTH";:POKE 85,17:PRINT "AV
ERAGE"
480 PRINT MONTH$(I*9-8,I*9);:POKE 85,18:
PRINT Z(I)*CR
```

©



SUPERGRAPHICS

128,000 ACCESSIBLE POINTS

Features

* 128,000 accessible points arranged in 200 x 640 grid
* 16K static ram supplies memory mapping of pixels on separate board
* supported through extended Basic commands supplied in ROM
* uses no memory from the existing system
* mix standard CBM text and low-res graphics with hi-res displays
* easy to design and display user defined shapes
* normal or inverted display mode for partial or total screen
* switch graphic display on or off - continue output in passive mode
* save graphic images on a disk file
* recall images from disk with option to mix with existing display
* supports hardcopy to any hi-res Epson printer
* easy to follow installation instruction included

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

DISTRIBUTORS OF:
MUPET • DOUBLE-MUPET • SPOOLER
THE MANAGER • I/O PRODUCTS

Problems with memory cells in your computer's RAM (where you do your programming) can sometimes be quite subtle. It would be clear that you had a bad cell somewhere if your computer responded to your question, ? FRE(1) by saying that you had 320 bytes free before you'd even put a program in. But what about the less obvious memory problems? A cell might only go strange when the cell next to it *contains a five. Or it might work fine, but fail after a certain amount of time passes. Such subtle failures, fortunately relatively rare, can have you looking in vain for a bug in your program. Mr. Scanlon presents a sophisticated memory testing program here (in machine language, for any computer which uses the 6502 chip, Apple, CBM/PET, Atari, OSI, VIC). This program can uncover some of those more subtle cell failures.*

Track Down Those Memory Bugs!

Leo J. Scanlon
Inverness, FL

If you just went out and bought a new tape recorder, a pair of jeans, or a quart of milk, you'd find out in short order just how good or bad the product is. If the tape recorder makes a Sousa march sound like a funeral dirge, back to the store it goes. Likewise, the jeans will show their quality after the first washing, and your nose knows if the milk is bad!

Unfortunately, faults in a computer memory board may not be that evident. Certainly, you will spot obvious defects – a crack in the board, missing chips, and the like – but if the board is operating at all, you'll probably need a diagnostic program to pinpoint any specific problem. This article presents one such program. It was developed on a Rockwell AIM 65 microcomputer, but it can run on any 6502-based computer, provided you alter the output routines. The general *principles* can, of course, be applied to other types as well.

Besides "dead" chips, which cannot store any data, memory boards have a variety of other potential problems. For example, some chips contain one or more bits that will not accept information, or bits that just hold the information briefly, then lose it. Other chips will not accept certain bit *patterns,* or affect other memory chips in the array. We can't hope to write a single program that will identify all possible errors, but the program given here will isolate most errors – or at least give you enough information to delve deeper into the problem.

The Test Algorithm

The diagnostic program in this article uses an algorithm that was implemented for 68000-based systems by Robert D. Grappel ("M68000 Diagnostic Program Tests Memory," *EDN*, April 15, 1981, pp. 157-158). This algorithm has two main loops. The first loop fills the tested portion of memory with increasing bit patterns; 00000000_2 is written into the first byte, 00000001_2 is written into the second byte, and so on. With this done, the second loop checks the memory contents and prints an error message each time a mismatch is detected.

The test then repeats, each time incrementing the contents of each byte. Thus, after 256 cycles, each byte has held all possible values. Note that the test is destructive; any pre-test information in the affected RAM will be eradicated.

The program described here also allows you to check for either *hard failures* or *soft failures*. Hard failures are those that cause the loss of ability to change the state of one or more bits, whereas soft failures allow a change, but revert back to the original state after a period of time. For soft failure testing, the time delay between write and read/verify has been set at one minute, arbitrarily.

Program Flowcharts

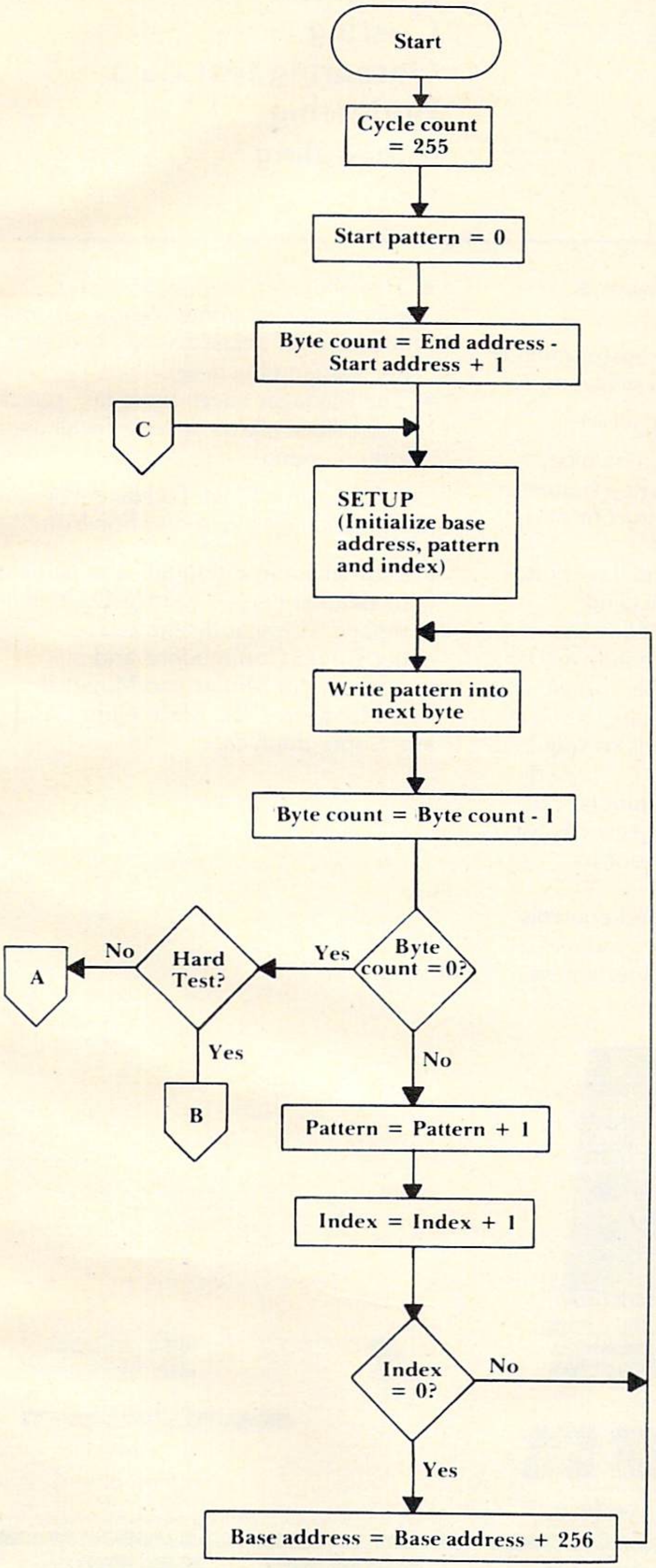
With the two tasks now defined, we can look at a program that will do the job. Before doing so, however, it will be helpful to investigate the overall structure of the two loops, by looking at their flowcharts.

Figure 1 shows the sequence of operations for the write loop, the loop that fills the test portion of memory with increasing bit patterns. This loop is preceded by some necessary initialization. First, the cycle count is set to 255. We actually want *256* cycles, but with a microprocessor that has only eight-bit registers (and memory locations), it is convenient to use 255, and plan ending the test when the cycle count has decremented *past* zero, to an all-ones (hex FF) state.

Next, the start pattern – the pattern that will be written into the first test location – is initialized to zero. Admittedly, zero is arbitrary. Since we will cycle through memory 256 times, each location will eventually receive every possible bit pattern. Therefore, it really doesn't matter which value goes into the first byte.

The microprocessor then calculates the byte count, by taking the difference between the specified end address and start address, and adding one. This is followed by a call to a set-up subroutine, which sets the "base address" equal to the starting address, fetches the start pattern and sets a byte index equal to zero. (The use of the terms base address and index here show that we plan to use

Figure 1: Initialization and Write Sequences



one of the 6502's *indexed* addressing modes. These modes calculate an effective address by adding the contents of an index register – X or Y – to an absolute or indirect base address.)

At this point, the microprocessor enters the actual write loop. It starts by writing a pattern into memory, decrementing the byte count, then checking whether test memory has been entirely filled with test patterns (byte count = 0). When the byte count is zero, the microprocessor branches to the read/verify loop, at either its soft error test or hard error test entry point; path A or path B, respectively. Otherwise, pattern and index are incremented, in preparation for writing to the next byte.

When the index has been incremented past hex FF, and reaches zero, a new base address is calculated, by adding 256 to the existing base address. Again, this is necessary because our index registers are only eight bits wide. A nonzero index causes the microprocessor to loop back to write the next pattern into memory. This concludes our discussion of the write loop.

As expected, the sequence of the read/verify loop (Figure 2) is very similar to that of the write loop. However, the read/verify loop has two separate entry points, one for soft error testing (in which the loop is preceded by a one-minute time delay), and the other for hard error testing. The read/verify loop begins with a call to the SETUP subroutine, to fetch the starting test pattern and reinitialize the base address and index.

With this initialization completed, the byte-by-byte comparisons begin. This consists of comparing the contents of each memory location against the test pattern that was written into it. If a mismatch is detected, the microprocessor calls an error subroutine, to print out the pertinent information – bad address, expected pattern and the pattern read. The rest of the read/verify loop is identical to the bottom of the write loop, except that when all locations have been read, the read/verify loop increments the start pattern and checks for end-of-test (cycle count less than zero). If further testing is necessary, the microprocessor branches back to the beginning of the write loop.

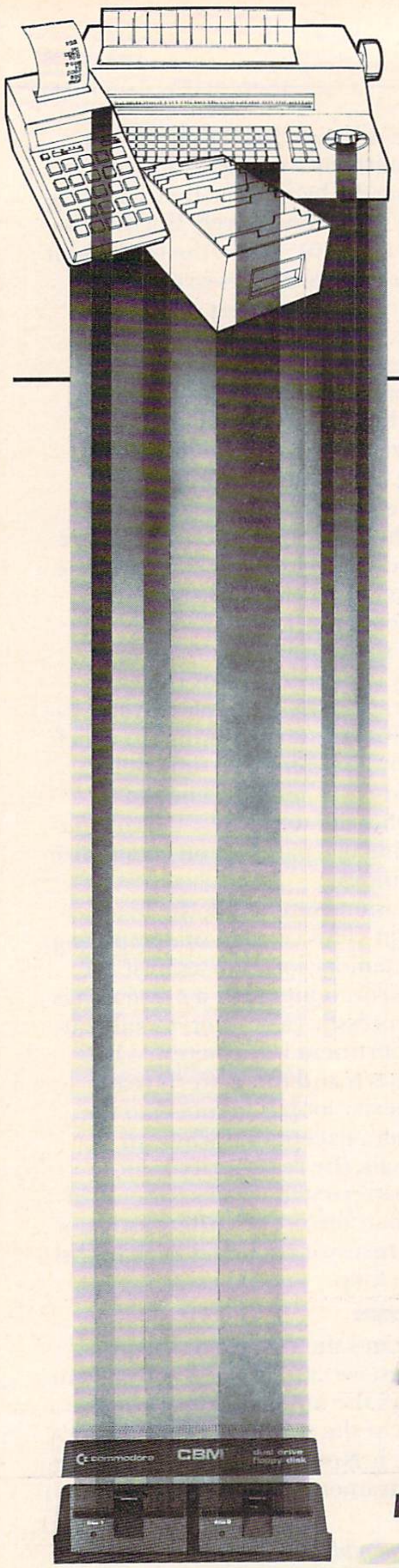
The Diagnostic Program

Now that you understand the criteria of the program and its sequences, we can look at the program itself. Program 1 shows the initialization and write sequence, the portion of the program that was flowcharted in Figure 1. Note that before executing the program, three parameters must be stored in zero page:

1. Store the *starting address* in locations 00 and 01, with the low byte in 00.

2. Store the *ending address* in locations 02 and 03, with the low byte in 02.

3. Select soft error testing or hard error testing by storing a value of 00 or 01, respectively, in location 04.



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

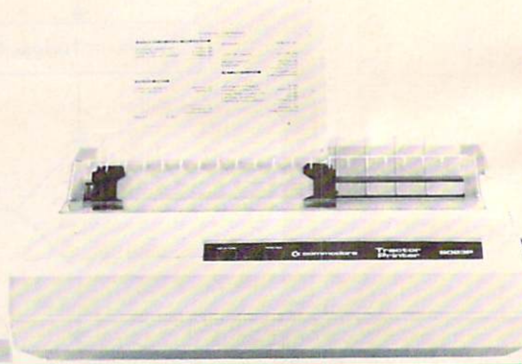
The easy to use Database System suited for both businessman and programmer.

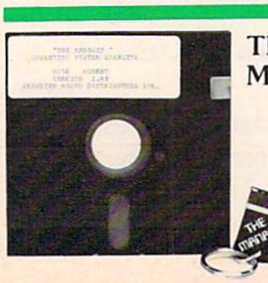
Today's Filing Cabinet for CBM

- 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.





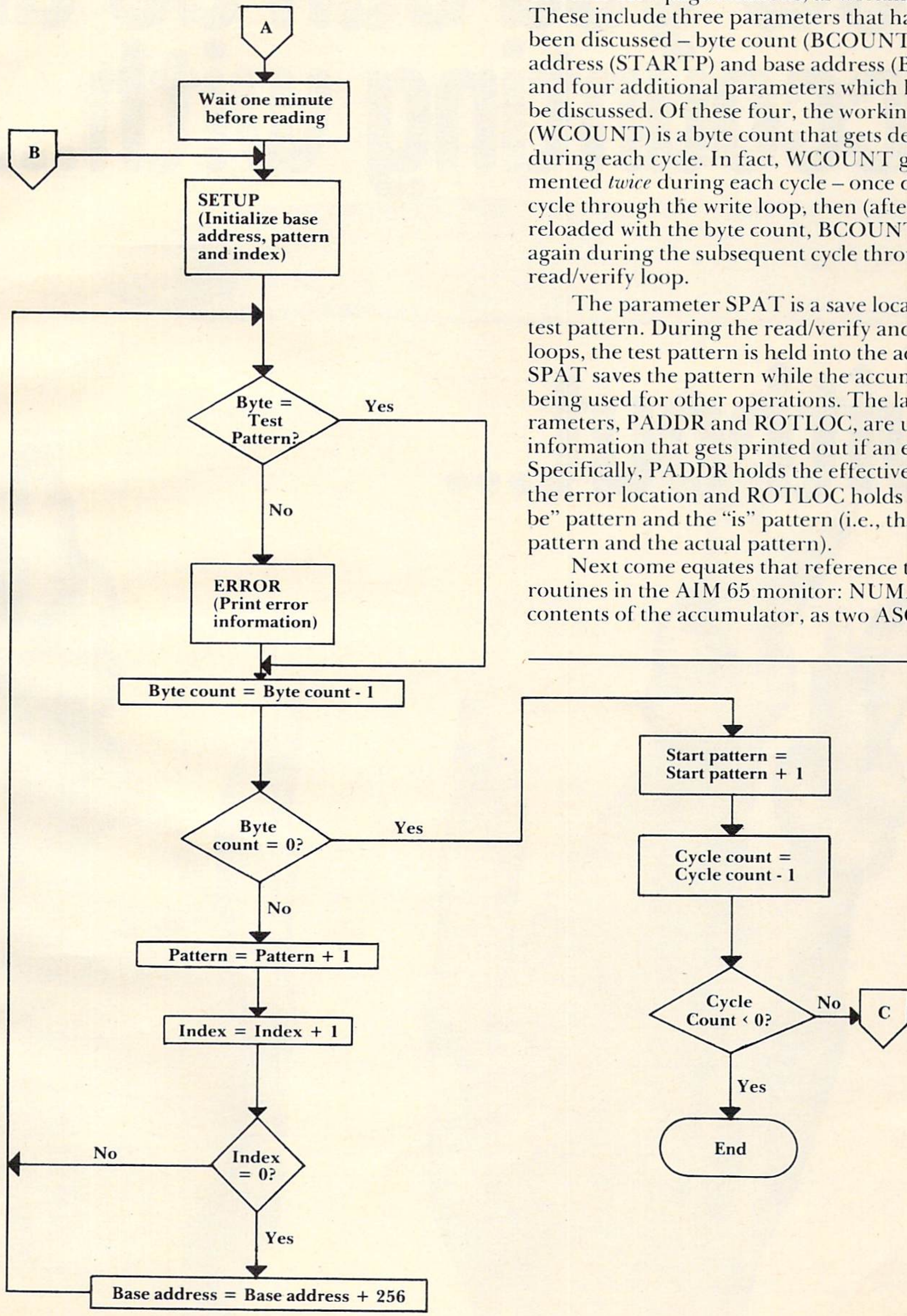
THE MANAGER

$250.00

CMD

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

Figure 2: Read/Verify Sequence

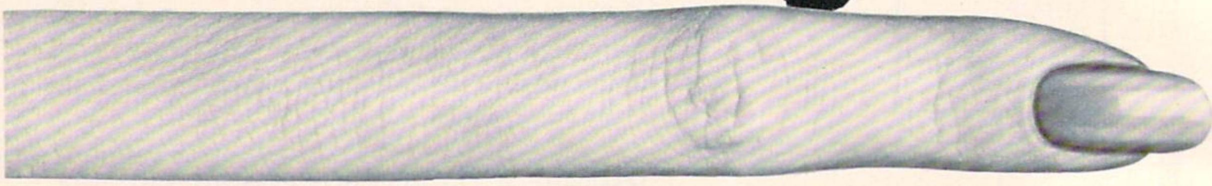


Besides these five locations, the program uses 11 other zero page locations, as working storage. These include three parameters that have already been discussed – byte count (BCOUNT), starting address (STARTP) and base address (BADDR) – and four additional parameters which have yet to be discussed. Of these four, the working byte count (WCOUNT) is a byte count that gets decremented during each cycle. In fact, WCOUNT gets decremented *twice* during each cycle – once during a cycle through the write loop, then (after being reloaded with the byte count, BCOUNT) once again during the subsequent cycle through the read/verify loop.

The parameter SPAT is a save location for the test pattern. During the read/verify and write loops, the test pattern is held into the accumulator; SPAT saves the pattern while the accumulator is being used for other operations. The last two parameters, PADDR and ROTLOC, are used to hold information that gets printed out if an error occurs. Specifically, PADDR holds the effective address of the error location and ROTLOC holds the "should be" pattern and the "is" pattern (i.e., the expected pattern and the actual pattern).

Next come equates that reference three subroutines in the AIM 65 monitor: NUMA prints the contents of the accumulator, as two ASCII charac-

Now you can do all accounting with...



without...

Program 1: Source Code for Initialization and Write Sequences

```
HDE ASSEMBLER REV 2.2

 LINE#    ADDR    OBJECT      LABEL  SOURCE

01-0010   2000                ; THIS PROGRAM TESTS MEMORY FOR BOTH HARD AND
01-0020   2000                ; SOFT ERRORS.
01-0030   2000                ; BEFORE EXECUTING, STORE THE FOLLOWING
01-0040   2000                ; PARAMETERS IN MEMORY:
01-0050   2000                ;  LOCS. 00 AND 01 = STARTING ADDRESS
01-0060   2000                ;  LOCS. 02 AND 03 = ENDING ADDRESS
01-0070   2000                ;  LOC. 04 = HARD ERROR (01) OR SOFT ERROR (00)
01-0080   2000                ; THE PROGRAM RETURNS TO THE MONITOR WHEN DONE.

01-0100   2000                ; USER-SUPPLIED PARAMETERS

01-0120   2000                       *=0
01-0130   0000                START  *=*+2            ; STARTING ADDR
01-0140   0002                END    *=*+2            ; ENDING ADDR
01-0150   0004                HARD   *=*+1            ; HARD/SOFT ERROR TEST SELECT

01-0170   0005                ; EQUATES FOR WORKING STORAGE IN ZERO PAGE

01-0190   0005                BCOUNT *=*+2            ; BYTE COUNT
01-0200   0007                WCOUNT *=*+2            ; WORKING BYTE COUNT
01-0210   0009                STARTP *=*+1            ; STARTING PATTERN
01-0220   000A                BADDR  *=*+2            ; BASE ADDRESS
01-0230   000C                SPAT   *=*+1            ; PATTERN IS SAVED HERE
01-0240   000D                PADDR  *=*+2            ; ERROR BYTE ADDRESS
01-0250   000F                ROTLOC *=*+1            ; WORKING BYTE FOR PRINT ROUTINE

01-0270   0010                ; AIM 65 MONITOR SUBROUTIINES

01-0290   0010                NUMA   =$EA46           ; PRINT A, AS TWO ASCII CHARS.
01-0300   0010                OUTPRI =$F000           ; OUTPUT A TO PRINT BUFFER
01-0310   0010                CRLOW  =$EA13           ; RESET DISPLAY & PRINTER

01-0330   0010                ; INITIALIZATION SEQUENCE

01-0350   0010                       *=$200
01-0360   0200  A2 FF         INIT   LDX #255         ; CYCLE COUNT = 255
01-0370   0202  A9 00                LDA #0           ; STARTING PATTERN = 0
01-0380   0204  85 09                STA STARTP
01-0390   0206  38                   SEC              ; BYTE COUNT = END ADDR. -
01-0400   0207  A5 02                LDA END          ;  START ADDR. + 1
01-0410   0209  E5 00                SBC START
01-0420   020B  85 05                STA BCOUNT
01-0430   020D  A5 03                LDA END+1
01-0440   020F  E5 01                SBC START+1
01-0450   0211  85 06                STA BCOUNT+1
01-0460   0213  E6 05                INC BCOUNT
01-0470   0215  D0 02                BNE MLOOP
01-0480   0217  E6 06                INC BCOUNT+1
01-0490   0219  20 8C 02      MLOOP  JSR SETUP        ; INITIALIZE COUNT, ADDR, INDEX
01-0510   021C                ; WRITE SEQUENCE

01-0530   021C  91 0A         WRITE  STA (BADDR),Y    ; WRITE PATTERN INTO NEXT BYTE
01-0540   021E  85 0C                STA SPAT         ;  AND SAVE PATTERN
01-0550   0220  38                   SEC              ; DECREMENT BYTE COUNT
01-0560   0221  A5 07                LDA WCOUNT
01-0570   0223  E9 01                SBC #1
01-0580   0225  85 07                STA WCOUNT
01-0590   0227  A5 08                LDA WCOUNT+1
01-0600   0229  E9 00                SBC #0
01-0610   022B  85 08                STA WCOUNT+1
01-0620   022D  D0 0A                BNE INCP         ; BYTE COUNT = 0?
01-0630   022F  C5 07                CMP WCOUNT
01-0640   0231  D0 06                BNE INCP
01-0650   0233  C5 04                CMP HARD         ;  YES.  GO READ/VERIFY
01-0660   0235  D0 23                BNE READH
01-0670   0237  F0 0C                BEQ READS
```

```
ADDR   OBJECT      LABEL   SOURCE

0239   A5 OC       INCP    LDA SPAT          ;  NO.  GET PATTERN
023B   18                  CLC               ;   AND INCREMENT IT
023C   69 01               ADC #1
023E   C8                  INY               ; INCREMENT INDEX
023F   DO DB               BNE WRITE         ; INDEX = 0?
0241   E6 OB               INC BADDR+1       ;  YES.  ADD 256 TO BASE ADDRESS
0243   DO D7               BNE WRITE         ;   AND GO WRITE TO NEXT BYTE
```

ters; OUTPRI sends one character to the print buffer; CRLOW initializes the display and printer to their "start" positions.

The rest of Program 1 shows the source code for the initialization sequence and the write loop. The programming is straightforward, so you should have no problem following it if you studied the flowchart in Figure 1. Note that the write loop is terminated when the byte count has been decremented to zero. At that time, we interrogate the contents of the user-specified parameter HARD (location $04), and branch to the read/verify sequence, at either READS (if HARD contains zero) or READH (if HARD contains a nonzero value).

The read/verify sequence, shown in Program 2, also follows its earlier flowchart (Figure 2), and

Program 2: Source Code for Read/Verify Sequence

```
ADDR   OBJECT      LABEL   SOURCE

0245               ; READ/VERIFY SEQUENCE.  ENTER HERE FOR SOFT
0245               ; ERROR TESTING

0245   8A          READS   TXA               ; SAVE X (CYCLE COUNT) ON STACK
0246   48                  PHA
0247               ; WAIT ONE MINUTE BEFORE BEGINNING TO READ
0247   A9 C8       DELAY1  LDA #200          ; EXECUTION COUNT = 200
0249   A2 A5       D300    LDX #$A5          ; LOAD X AND Y FOR A 300 MS. DELAY
024B   AO EA               LDY #$EA
024D   CA          WAIT    DEX
024E   DO FD               BNE WAIT          ; LOOP UNTIL X = 0
0250   88                  DEY
0251   DO FA               BNE WAIT          ; LOOP UNTIL Y = 0
0253   38                  SEC               ; DECREMENT TIMING BYTE
0254   E9 01               SBC #1
0256   DO F1               BNE D300          ; LOOP UNTIL A = 0
0258   68                  PLA
0259   AA                  TAX               ; RESTORE CYCLE COUNT
025A               ; ENTER HERE FOR HARD ERROR TESTING
025A   20 8C 02    READH   JSR SETUP         ; REINITIALIZE PARAMETERS
025D   D1 0A       COMP    CMP (BADDR),Y     ; BYTE = TEST PATTERN?
025F   85 OC               STA SPAT
0261   FO 03               BEQ DECBC
0263   20 A1 02            JSR ERROR         ;  NO. PRINT ERROR MESSAGE
0266   38          DECBC   SEC               ;  YES.  DECREMENT BYTE COUNT
0267   A5 07               LDA WCOUNT
0269   E9 01               SBC #1
026B   85 07               STA WCOUNT
026D   A5 08               LDA WCOUNT+1
026F   E9 00               SBC #0
0271   85 08               STA WCOUNT+1
0273   DO 04               BNE INCP1         ; BYTE COUNT = 0?
0275   C5 07               CMP WCOUNT
0277   FO OB               BEQ DECCC
0279   A5 OC       INCP1   LDA SPAT          ;  NO.  GET PATTERN
027B   69 01               ADC #1            ;   AND INCREMENT IT
027D   C8                  INY               ; INCREMENT INDEX, TOO
027E   DO DD               BNE COMP          ; INDEX = 0?
0280   E6 OB               INC BADDR+1       ;  YES.  ADD 256 TO BASE ADDRESS
0282   DO D9               BNE COMP          ;   AND GO COMPARE NEXT BYTE
0284   E6 09       DECCC   INC STARTP        ; INCREMENT START PATTERN
0286   CA                  DEX               ; DECREMENT CYCLE COUNT
0287   EO FF               CPX #$FF          ; COUNT CYCLE NEGATIVE?
0289   DO 8E               BNE MLOOP
028B   00                  BRK               ;  YES.  RETURN TO MONITOR
```

DOS FOR AIM-65®: $499.00 COMPLETE!

INSTANT EASY ACCESS TO YOUR AIM-65 PROGRAMS!

MANIPULATE OBJECT FILES AT THE TOUCH OF A KEY.

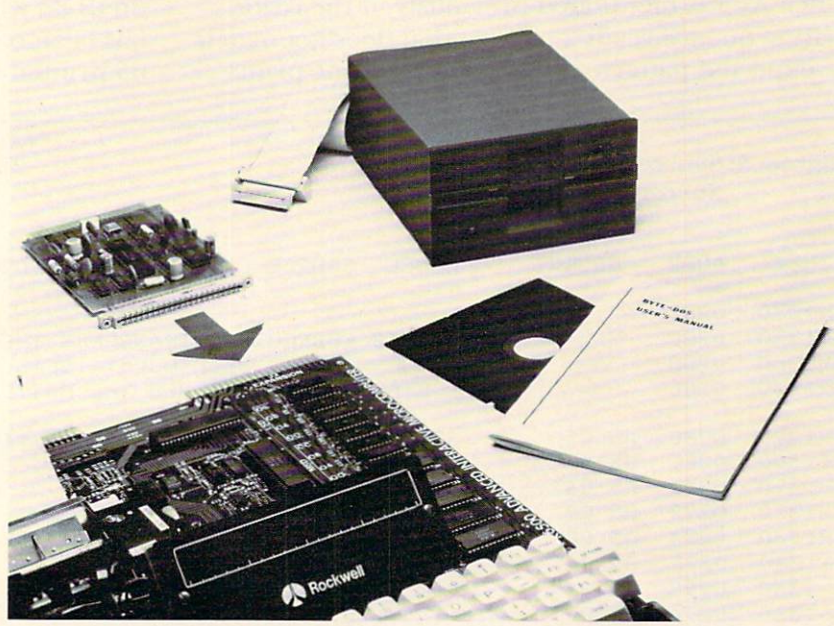
NEW BYTE-DOS DISC OPERATING SYSTEM.

153K BYTES ON-LINE. INEXPENSIVELY!

BYTE–DOS COMMANDS

- **FORMAT** - Prepares New Disc
- **BAD SECTOR SCAN** - QC's Disc
- **DUMP** - RAM File to Disc
- **LOAD** - Disc File to Ram
- **LIST** - Prints Disc File Names
- **ERASE** - Eliminate a File
- **UNERASE** - File Recovery
- **RENAME** - Rename a File
- **COPY** - Copy 1 File
- **TRANSFER** - Copy Complete Disc
- **KOMPRESS** - Optimizes Disc Space

AND MORE!



IT'S EASY AND FUN TO USE!

master charge THE INTERBANK CARD
VISA

BYTE MICROSYSTEMS
CORPORATION
1477 ELKA AVENUE, SAN JOSE, CA 95129 408/446/0559

- *BYTE-DOS OPERATES WITH THE AIM-65 MONITOR, BASIC, ASSEMBLER, EDITOR, PL/65 AND FORTH. ACCESS IS THROUGH PROGRAM I/O AND F1-F3 KEYS. BYTE-DOS SUPPORTS ONE OR TWO 5¼" OR 8" FLOPPYS!*
- *BYTE-DOS $499 SYSTEM INCLUDES DISC CONTROLLER CARD, TEAC FD-50A DISC DRIVE (153K FORMATTED BYTES), BYTE-DOS SOFTWARE SYSTEM, CABLES, CONNECTORS, MANUALS. COMPLETE! JUST PLUG IT TOGETHER AND USE!*

SEND ME THE FULL STORY!

Name ______________________

Company ______________________

Address ______________________

City ____________ State ______ Zip ______

Telephone ______________ ☐ home ☐ work

☐ I own an AIM-65.

☐ I'm thinking of buying an AIM-65.

☐ Send full BYTE-DOS Data.

needs no additional description. However, you may be interested in the one-minute time delay routine that gets executed if you are testing for soft errors. This routine, DELAY1, generates the one-minute delay by executing a 300-millisecond delay routine (D300) 200 times! Readers who are interested in the details of this and other delay routines are referred to my book *6502 Software Design* (Howard W. Sams & Co., 1980). Because the D300 uses the X Register – which holds our program's cycle count – the contents of that register must be saved on the stack while the time delay is being generated.

The final program, Program 3, shows the source code for the set-up subroutine (SETUP), followed by the error printout subroutine (ERROR). As was mentioned previously, if the read/verify sequence reads a pattern that does not match the expected pattern, the error subroutine prints out three items of information: the address of the offending location, the expected pattern (the "S/B" pattern) and the pattern that is actually read (the "IS" pattern). As an aid to identifying faulty *bits* within a memory location, the S/B and IS patterns are printed in a binary representation. To do this, ERROR calls a second subroutine (BINARY) that left-rotates the contents of a working zero-page location, ROTLOC, eight times. After each rotate operation, BINARY outputs an ASCII 1 or ASCII 0 to the printer.

Figure 3 shows the kind of listing that the ERROR subroutine produces. To generate this listing on my AIM 65, I selected three locations within the monitor ROM – LOCATIONS $E000, $E001, and $E002 – knowing that a read/verify test on ROM will always fail. As you can see, the printout has been retyped for publication, because the AIM 65 printer output does not reproduce very well.

Program 3: Source Code for Set-Up and Error Subroutines

```
 LINE#    ADDR   OBJECT      LABEL  SOURCE              PAGE 0004

01-1230  028C
01-1250  028C               ; THIS SUBROUTINE TRANSFERS THE BYTE COUNT TO
01-1260  028C               ; THE WORKING BYTE COUNT, SETS BASE ADDR =
01-1270  028C               ; START ADDR, FETCHES THE PATTERN, AND SETS INDEX = 0

01-1290  028C   A5 05       SETUP  LDA BCOUNT          ; INITIALIZE WORKING BYTE COUNT
01-1300  028E   85 07              STA WCOUNT
01-1310  0290   A5 06              LDA BCOUNT+1
01-1320  0292   85 08              STA WCOUNT+1
01-1330  0294   A5 00              LDA START           ; TO BEGIN, BASE ADDR = START ADDR
01-1340  0296   85 0A              STA BADDR
01-1350  0298   A5 01              LDA START+1
01-1360  029A   85 0B              STA BADDR+1
01-1370  029C   A5 09              LDA STARTP          ; INITIALIZE PATTERN
01-1380  029E   A0 00              LDY #0              ; INDEX = 0
01-1390  02A0   60                 RTS

01-1410  02A1               ; PRINT ERROR INFORMATION
01-1420  02A1               ;  BAD ADDRESS (HEX)
01-1430  02A1               ;  S/B PATTERN (BINARY)
01-1440  02A1               ;  IS PATTERN (BINARY)

01-1460  02A1   98          ERROR  TYA                 ; SAVE Y ON STACK
01-1470  02A2   48                 PHA
01-1480  02A3   18                 CLC                 ; CALCULATE PRINT ADDRESS
01-1490  02A4   65 0A              ADC BADDR
01-1500  02A6   85 0D              STA PADDR
01-1510  02A8   A9 00              LDA #0
01-1520  02AA   65 0B              ADC BADDR+1
01-1530  02AC   85 0E              STA PADDR+1
01-1540  02AE   20 13 EA           JSR CRLOW           ; RESET DISPLAY AND PRINTER
01-1550  02B1   A0 00              LDY #0              ; PRINT LOC LINE
01-1560  02B3   B9 0B 03    LOOP1  LDA LINE1,Y
01-1570  02B6   20 00 F0           JSR OUTPRI
01-1580  02B9   C8                 INY
01-1590  02BA   C0 07              CPY #7
01-1600  02BC   D0 F5              BNE LOOP1
01-1610  02BE   A5 0E              LDA PADDR+1
01-1620  02C0   20 46 EA           JSR NUMA
01-1630  02C3   A5 0D              LDA PADDR
```

SYSRES™

THE ULTIMATE RESIDENT PROGRAM MANIPULATION SYSTEM FOR PET™/CBM™ MICROCOMPUTERS

SYSRES™ EXTENDED DOS SUPPORT COMMANDS

@ (type "N" keyboard)
← (type "B" keyboard)
! (original keyboard)
> (for 'wedge' users)

These commands may be used interchangably, to perform the following dos support functions.

Disk	Printer	Tape	Directory	Modes	Command	Function
x				3	@	Display disk status / send command
x					@N	Format (header) a new diskette
x					@I	Force initialize diskette
x					@V	Validate diskette (collect)
x					@D	Duplicate diskette
x			x	4	@C	Copy or concatenate disk file(s)*
x					@R	Rename file
x			x	3	@S	Scratch file(s)*
x	x				@$	List directory**
x					@U:	Reset disk drive
x	x	x	x	6	@L	List disk file or BASIC program**

Note: Some of the disk utility command set may also be used, if an appropriate direct access channel has been opened.

* Standard command with added options.
** Added disk command.

SYSRES™ EXTENDED EDITOR COMMANDS

Disk	Printer	Tape	Directory	Modes	Command	Function
x			x	4	/	Quick load from disk
x			x	4	↑	Quick load from disk with auto run
x			x	2	APPEND	Append from disk to end of current program
				4	AUTO	Auto line number (allows header)
x			x	3	BLOAD	Load machine language (binary) file
x			x	3	BRUN	Load and execute machine language program
	x			776	CHANGE	Change pattern to another pattern
				2	CLOSE	Close one or all files
				1	CMD	Set output to file (does not send "READY.")
				4	DELETE	Delete a range of lines from program
	x			1	DUMP	Dump all scalar variables to screen or file
x			x	2	EXEC	Execute a file as keyboard commands
	x			240	FIND	Find occurances of a pattern
x		x	x	3	GET	Read a sequential file into editor
				7	KEY	Define a key as a special function
				1	KEYS	Turn key functions on
				1	KILL	Disable SYSRES™
				1	KILL*	Disable SYSRES™ and unreserve memory
	x			10	LIST	Improved BASIC LIST command
x		x	x	3	LOAD	Defaults to disk drive
x			x	2	MERGE	Merge from disk into current program
	x			1	MON	Break to current machine language monitor
				1	OLD	Restore program after "NEW"
x	x	x	x	24	PUT	Send program to disk as text file
				6	RENUMBER	Renumber all or part of program
				2	RUN	Run current program, ignores screen garbage
x		x	x	3	SAVE	Defaults to disk drive, allows replace
x		x		1	SETD	Set disk device #, allows multiple drives
	x			4	SETP	Set printer channel, format mode, paging
	x			4	TRACE	Select 1 of 3 trace/step modes and speed
x		x	x	3	VERIFY	Compare current program against disk/tape
				1	WHY	Print position of last error
				1	WHY?	List line of break or error
x	x				*	Send output to printer
	x			1	#	Display current version of SYSRES™

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!
* Diskette and Extensive Manual – only $75
Documented Source Code (YES!) – $150
Custom versions are available for unique systems!

Dealer enquiries are welcome.

VISA MasterCard

CANSOFT

CANSOFT DATA INC.

MICROCOMPUTER SERVICES

PROGRAMMING * CONSULTING * DESIGNING * EVALUATING

BOX 76752, STN.S, VANCOUVER, B.C. V5R 5S7 TEL:(604)437-6667

```
ADDR  OBJECT     LABEL  SOURCE
02C5  20 46 EA          JSR NUMA
02C8  20 13 EA          JSR CRLOW
02CB  B9 0B 03   LOOP2  LDA LINE1,Y        ; PRINT S/B LINE
02CE  20 00 F0          JSR OUTPRI
02D1  C8                INY
02D2  C0 0F             CPY #15
02D4  D0 F5             BNE LOOP2
02D6  A5 0C             LDA SPAT
02D8  20 F6 02          JSR BINARY
02DB  20 13 EA          JSR CRLOW
02DE  A0 0F             LDY #LINE3-LINE1
02E0  B9 0B 03   LOOP3  LDA LINE1,Y        ; PRINT IS LINE
02E3  20 00 F0          JSR OUTPRI
02E6  C8                INY
02E7  C0 17             CPY #23
02E9  D0 F5             BNE LOOP3
02EB  68                PLA
02EC  48                PHA
02ED  A8                TAY
02EE  B1 0A             LDA (BADDR),Y
02F0  20 F6 02          JSR BINARY
02F3  68                PLA
02F4  A8                TAY
02F5  60                RTS

02F6             ; PRINT PATTERN IN BINARY FORMAT

02F6  A0 08      BINARY LDY #8
02F8  85 0F             STA ROTLOC
02FA  26 0F      ROTATE ROL ROTLOC         ; IS BIT A 1 OR A 0?
02FC  90 04             BCC SBZERO
02FE  A9 31             LDA #'1'           ; IT'S A 1
0300  D0 02             BNE PBIT
0302  A9 30      SBZERO LDA #'0'           ; IT'S A 0
0304  20 00 F0   PBIT   JSR OUTPRI
0307  88                DEY
0308  D0 F0             BNE ROTATE
030A  60                RTS

030B             ; MESSAGES FOR ERROR SUBROUTINE

030B  20 20      LINE1  .BYT '  LOC.='
0312  20 20      LINE2  .BYT '     S/B='
031A  20 20      LINE3  .BYT '      IS='
0322                    .END

ERRORS = 0000    END OF ASSEMBLY = 0321
```

Execution Times For The Test Program

Having seen the source code listings, you now know that the test program occupies slightly more than a page of memory; to be exact, it occupies 290 bytes. However, you're probably more curious about how long the program takes to execute – which translates to how long you will have to stand around before you know whether or not the memory is "bug-free."

In testing for *hard errors*, the program takes about 25 seconds to test 1024 (or 1K) bytes, if your computer has a 1-MHz clock. This means that it will take six minutes and 45 seconds to test a 16K-byte board.

In testing for *soft errors*, the program introduces a one-minute time delay between each write and read/verify sequence. And since the program executes 256 cycles, soft error testing will always take two hours and 16 minutes more than hard error testing, *regardless* of how much memory is being tested! Therefore, it will take roughly two hours, 22 minutes and 45 seconds to test a 16K board for soft errors. Clearly, it's best to check for hard errors first, then re-check for soft errors if you're still having problems.

Figure 3: Sample Error Printout

```
LOC. = E000
  S/B = 00000000
   IS = 01000110
LOC. = E001
  S/B = 00000001
   IS = 01010010
LOC. = E002
  S/B = 00000010
   IS = 01001111
LOC. = E000
  S/B = 00000001
   IS = 01000110
LOC. = E001
  S/B = 00000010
   IS = 01010010
LOC. = E002
  S/B = 00000011
   IS = 01001111
LOC. = E000
  S/B = 00000010
   IS = 01000110
LOC. = E001
  S/B = 00000011
   IS = 01010010
LOC. = E002
  S/B = 00000100
   IS = 01001111
.
.          (253 more sets of listings follow.)
.
```

©

NEECO

WHY BUY FROM THE BEST?

Service... Support...
Software...

commodore

16K B (16K RAM-40 Column) - Lim. Qty	$ 995
32K B (32K RAM-40 Clm.) - Lim. Qty	$1295
4016 (16K RAM 4.0 Basic-40 Clm.)	$ 995
4032 (32K RAM 4.0 Basic-40 Clm.)	$1295
8032 (32K RAM 4.0 Basic-80 Clm.)	$1495
8050 Dual Disk (1 Meg Storage)	$1795
4040 Dual Disk (343K Storage)	$1295
8010 IEEE Modem	$ 280
C2N Cassette Drive	$ 75
CBM - IEEE Interface Cable	$ 40
IEEE - IEEE Interface Cable	$ 50
VIC 20 Home/Personal Computer	$ 295

**CALL NEECO
FOR ANY OF YOUR
COMMODORE COMPUTER NEEDS**

EPSON PRINTERS

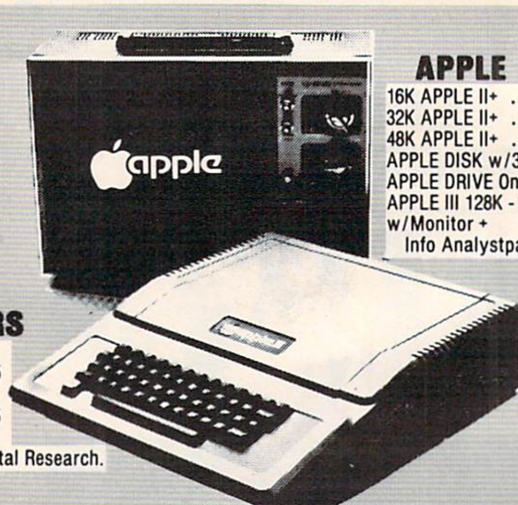
MX-80 PRINTER	$ 645
MX-80 FT	$ 745
MX-100	$ 945
MX-70	$ 459
INTERFACE CARDS	
8141 (RS-232)	$ 75
8150 (2K Buffered RS-232)	$ 150
8161 (IEEE 488)	$ 55
8131 (Apple Card)	$ 85
8230 (Apple Card)	$ 25
8220 (TRS-80 Cable)	$ 35

NEC SPINWRITER PRINTERS

5530 (Parallel)	$3055
5510 (Serial)	$3055
5520 (KSR-Serial)	$3415
Tractor Option	$ 225

DIABLO 630 PRINTER

DIABLO 630 - Serial - RS-232	$2710
Tractor Option	$ 250



APPLE

16K APPLE II+	$1330
32K APPLE II+	$1430
48K APPLE II+	$1530
APPLE DISK w/3.3 DOS	$ 650
APPLE DRIVE Only	$ 490
APPLE III 128K - In Stock! w/Monitor + Info Analystpak	$4740

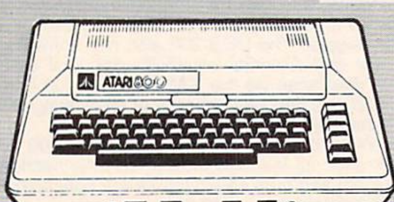
AMDEK MONITORS

Video 100 12" B+W	$ 179
Video 300 12" Green	$ 249
Color I 13" Low Res	$ 449
Color II 13" High Res	$ 999

INTERTEC COMPUTERS

64K Superbrain (360 Disk Storage), CP/M™	$3495
64K QD Superbrain (700K Disk Storage), CP/M™	$3995

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



ATARI COMPUTERS

Atari 400 (16K RAM)	$ 399
Atari 800 (32K RAM) - good thru 8/31	$1080
Atari 410 RECORDER	$ 89.95
Atari 810 DISK DRIVE	$ 599.95

NEECO carries all available ATARI Software and Peripherals.

PROFESSIONAL SOFTWARE

WordPro 1 8K	$ 29.95
WordPro 3 (40 Clm.)16K	$ 199.95
WordPro 3+	$ 295
WordPro 4 (80 Clm.) 32K	$ 375
WordPro 4+	$ 450

**JUST A SAMPLE OF THE MANY PRODUCTS WE CARRY, CALL US FOR OUR NEW 60-PAGE CATALOG.
WE WILL MATCH SOME ADVERTISED PRICES ON CERTAIN PRODUCTS LISTED UNDER SIMILAR "IN STOCK" CONDITIONS.**

NEECO

679 HIGHLAND AVE.
NEEDHAM, MA 02194

(617) 449-1760
Telex: 951021

MON-FRI 9:00 - 5:00

MasterCharge and VISA Accepted

Here's a game that's easy to program, short, and fun. There are versions of the game here for Atari, PET/CBM, Apple, and OSI.

Shooting Stars

Frank Cohen
Pacific Palisades, CA

"So what does your computer do?" I remember my parents asking me that question over and over again when I was a teenager, just getting into computers. I couldn't begin to answer a question like that. So, instead, I wrote a simple demo program to show them what was possible. After an hour of programming with my new O.S.I. Superboard II, I came up with Shooting Stars.

I originally saw a program like this on a friend's computer system and thought it would be easy to convert to my computer. And indeed it was.

The object of the game is to clear the screen of as many targets as possible using the fewest missiles. The number of targets is variable. When you first start the program it will clear the screen and print the welcome message. After a few seconds it will ask, "HOW MANY STARS?". Depending on how complex or how long a game you want to play, you can enter the number of targets you want to fire at. A suggested value is 30 to begin with. The program will then set up the star field and start the missile launcher.

To launch a missile, wait until the launcher passes under a star and hit the space bar. The number of missiles is unlimited. Your score is the ratio of missiles to targets destroyed. If you want to finish the game before you destroy all the missiles, press the A key. After you finish it will tell you how well you scored and ask if you want to play again. To play again leave another quarter under your computer and type, "Yes."

There are a number of areas where this program could be expanded. For example, add sound effects using the audio output port, make the number of missiles limited, or make the targets mobile. As the program is really very short when it has been condensed (using multiple instructions per line) it shouldn't take you a long time to add to Shooting Stars.

It's not a very complex program, but it does impress "non-computer" people. And it answers the question, "So, what does your computer do?" Well, at least for a while.

OSI Version

```
10 REM SHOOTING STARS V1.2
20 REM BY FRANK COHEN
30 REM BASED ON AN IDEA FROM ROB
   HOCKER
40 REM
100 REM INITIALIZE
110 X2=53989
120 X3=0
130 SC=0
140 S2=0
150 GOSUB1000
160 REM
170 REM PRINT GREETING
180 PRINT"WELCOME TO..."
190 PRINT:PRINT"          SHOOTING
    STARS"
200 FORI=1TO17:PRINT:NEXTI
210 REM
220 REM PLOT STARS
230 INPUT"HOW MANY STARS?";S
240 S1=S
250 GOSUB1000
260 X1=INT(RND(1)*480)
270 IF PEEK(X1+53381)=27 THEN260
280 POKEX1+53381,27
290 S=S-1
300 IFS>0THEN260
310 FORI=53405 TO 54141 STEP 32
320 FORJ=ITOI+8
330 IFPEEK(J)=27THENS1=S1-1
340 S5=S1
350 POKEI+J,32
360 NEXT J
370 NEXT I
375 REM
380 REM MOVE LAUNCHER
390 POKE 530,0
400 X3=0
410 POKEX2+32,248
420 POKEX2,32
430 X2=X2+1
440 IFX2>54012THEN POKEX2,32:X2=
    53989
450 REM
460 REM CHECK FOR WINN
470 IF S1<1THEN800
480 REM
490 REM READ KEYBOARD DIRECTLY
500 POKE530,1:POKE57088,253
510 IFPEEK(57088)=191THEN800
520 IFPEEK(57088)<>239THEN380
530 REM
540 REM FIRE A MISSILE
550 S2=S2+1
560 X4=X2-32
```

```
570 X4=X4-32
580 IFPEEK(X4)=27THEN630
590 POKEX4+32,32:POKEX4,139
600 IFX4>53381THEN570
610 POKEX4,32:GOTO 380
620 REM
630 REM SCORE ONE
640 POKEX4+32,32
650 POKEX4,32
660 SC=SC+1
670 S1=S1-1
680 REM
690 REM DESTROY A STAR
700 RESTORE
710 FORI=1TO15
720 READZ:POKEX4,Z
730 NEXTI
740 DATA233,232,233,232,226,187,184
    ,185
750 DATA184,186,232,186,27,185,32
760 GOTO 380
800 REM
810 REM PRINT FINAL SCORE
820 GOSUB1000
830 PRINT"GREAT..."
840 PRINT:PRINT"      YOU SCORED ";
    (S5/S2)*100;" %"
850 PRINT:PRINT:INPUT"DO YOU WANT
    TO PLAY AGAIN";A$
860 POKE530,0
870 IFA$="YES"THEN10
880 STOP
1000 REM CLEAR SCREEN SUBROUTINE
1010 FORI=1TO32
1020 PRINT
1030 NEXTI
1040 RETURN
9999 END
```

Note: If you added Super-Cursor (**COMPUTE!**, December, 1981, #19) to your Superboard II, then the clear screen subroutine at 1000 is unnecessary. You can replace it with: PRINT CHR$(1),CHR$(2). This works only with Supercursor.

Atari Version

This is not a line-by-line translation of Mr. Cohen's Shooting Stars program but rather, a game programmed especially for the Atari which plays a similar game.

The object of the game is to destroy a number of shooting stars (you select how many). Your score is the percentage of stars hit versus shots fired. Press SPACE to fire.

The game is fully commented, so you can trace the execution of the program and modify it as you please. Note that this game uses a little-known Atari graphics mode. More applications of these four color text/graphics modes would be welcome.

```
100 REM ************************
110 REM |  ATARI SHOOTING STARS  |
120 REM ************************
150 REM ------------------
160 REM Generate playfield
170 REM Using 3-color IRG mode 4
180 REM Set up GRAPHICS 1
190 REM Text line at top of
200 REM screen
210 GRAPHICS 0:POKE 752,1:REM Disable cu
rsor
220 DLIST=PEEK(560)+256*PEEK(561)+4:REM
Display list
230 SCREEN=PEEK(88)+256*PEEK(89):SCREEN=
SCREEN+40:REM Screen memory
240 POKE DLIST-1,6+64:POKE DLIST+2,6:REM
 IRG 6=Graphics mode 1
250 FOR I=3 TO 24:POKE DLIST+I,4:NEXT I:
REM change mode 0 lines to IRG 4
260 SETCOLOR 0,6,6:REM Set COLOR 1 to vi
olet
270 POSITION 3,0:? #6;"SHOOTING STARS"
280 DIM C(6):REM HOLDS STARS
290 C(0)=42+128:REM Reverse-field asteri
sk
300 C(1)=20+64:REM [CTRL-T], Ball
310 C(2)=20+128+64:REM Reverse-field bal
l
320 C(3)=16+64:REM [CTRL-P], Club
330 C(4)=16+128+64:REM Reverse field clu
b
340 C(5)=96:REM [CTRL-,] diamond
350 C(6)=96+128:REM Reverse-field diamon
d
360 FOR I=1 TO 30:FOR J=1 TO 10*RND(1):S
OUND 0,I+10*RND(1),10,8:NEXT J:NEXT I:SO
UND 0,0,0,0:REM Sound effect
370 ? "{CLEAR}":POSITION 1,0:? "HOW MANY
 STARS";
380 TRAP 370:INPUT STARS:TRAP 0
390 IF STARS=0 THEN GRAPHICS 0:END
400 IF STARS>50 THEN ? "{CLEAR}|too many
 stars{A}|":FOR W=1 TO 300:NEXT W:GOTO 3
70
410 ? "{CLEAR}":POSITION 3,0:? "SHOOTING
 STARS"
420 REM Draw stars
430 FOR I=1 TO STARS
440 STAR=C(INT(6*RND(0)+1))
450 REM START is column where the star w
ill fall from, length is distance
460 START=INT(40*RND(0)):LENGTH=INT(18*R
ND(0))
```


2 FROM HORIZON
Fathoms 40™
Time: June, 1943.
Place: North Atlantic.
German Wolf Packs have been driven from their hunting grounds by Allied escort vessels and massive air attacks. But unknown to the Allies is a Nazi super sub completed well ahead of schedule: The Mark XXI. Der Führer has demanded total destruction of enemy merchant shipping — a vital link for continued success of the Allied war effort. Will you, as one of the Kriegsmarine's remaining ace skippers, alter history and save the Third Reich from defeat?
Der Führer will not be disappointed!
FATHOMS 40: $39.95 ppd. Foreign, FPO and APO add 10% for shipping. Please specify Apple or Atari when ordering.
In the 1930's dramas were slowly unfolding in capital cities throughout the world. From dimly lit hotel rooms to highly sophisticated Intelligence Headquarters, men and women were playing a game of cat and mouse for scraps of information essential to the security of their country. CLOAK & DAGGER pits you against computer, or other players, as head of an Intelligence Agency trying desperately to stay one step ahead of its rivals. Moving agents from capital to capital, ferreting information, and offering bribes are fundamental parts of each player's turn . . . and your survival!
CLOAK & DAGGER™
CLOAK & DAGGER: $39.95 ppd. Foreign, FPO and APO add 10% for shipping. Please specify Apple or Atari when ordering.
For Apple II*, DOS 3.3, or Atari 800*, 48K with disk drive. Joysticks required. Games on Dysan** diskettes.
Visa & MasterCard accepted.
Dealer Inquiries Welcome, contact your local dealer for full details.
HORIZON SIMULATIONS
Horizon Simulations Inc
7561 Crater Lake Highway / White City / Oregon 97503
(503) 826-4640

```
470 R=RND(1):DIR=40-(R<0.3)+(R>0.6):REM
DIR is 39 (left diagonal, 40 (down), or
41 (right diagonal)
480 FOR FALL=0 TO LENGTH
490 P=PEEK(SCREEN+START+DIR*FALL):REM Sa
ve current character
500 SCR=SCREEN+START+DIR*FALL:POKE SCR,S
TAR:REM Replace it with a "star"
510 SOUND 0,10+FALL*2,8,8:REM Falling so
und
520 REM PEEK(53770) returns a random num
ber from 0-255
530 POKE 709,PEEK(53770):POKE 710,PEEK(5
3770):POKE 711,PEEK(53770):REM Change co
lors rapidly for "flash" effect
540 POKE SCR,P:REM Restore background ch
aracter
550 NEXT FALL
560 REM If final character already taken
 by another star, burn up star and try a
gain:
570 IF P<>0 THEN GOSUB 1010:POKE SCR,P:G
OTO 440
580 REM M is screen position modulo 40.
 Determines if star is at the left or ri
ght margin
590 M=((SCR-SCREEN)/40-INT((SCR-SCREEN)/
40))*40:IF M<3 OR M>36 THEN GOSUB 990:GO
TO 440
600 IF RND(1)>0.9 THEN GOSUB 990:GOTO 44
0:REM Occasionally burn up a star
610 POKE SCREEN+START+DIR*LENGTH,STAR:RE
M Place star finally
620 NEXT I
630 SOUND 0,0,0,0
640 FOR W=15 TO 0 STEP -0.05:SOUND 0,10,
8,W:NEXT W:REM Explosion sound
650 REM Set colors for good contrast
660 SETCOLOR 1,INT(16*RND(0)),10
670 SETCOLOR 2,INT(16*RND(0)),6
680 SETCOLOR 3,INT(16*RND(0)),10
690 REM Start game
700 S=1:F=37:REM Laser scans left to rig
ht, 1 to 37 and 37 to 1
710 FOR X=S TO F STEP SGN(F-S)
720 POSITION X,22:? " {D}|{X}|{O} {LEFT}
";:REM Draw laser
730 IF PEEK(764)=33 THEN MS=1:SHOTS=SHOT
S+1:MY=0:MX=X+1:POKE 764,255:GOSUB 970:R
EM Shoot missile if space is pres
740 IF MS=0 THEN 850
750 REM Update missile
760 MY=MY+1:IF MY=22 THEN MS=0:GOTO 850
770 SCR=SCREEN+(21-MY)*40+MX:P=PEEK(SCR)
780 POKE SCR,124:REM Vertical line
790 SOUND 0,MY,2,8
800 REM If missile hits something, blow
it up, make "boing" sound, give credit,
update score line
810 IF P THEN GOSUB 990:HITS=HITS+1:GOSU
B 970:MS=0:FOR W=15 TO 0 STEP -1:SOUND 0
,50,10,W:NEXT W:GOTO 840
820 POKE SCR,P:SOUND 0,0,0,0
830 REM Check if all stars hit
840 IF HITS=STARS THEN 870
850 NEXT X
860 S=37-S:F=37-F:GOTO 710
870 REM End of game
880 REM Flash score, wait for START
890 POSITION 0,0:? "NEW GAME:PRESS start
"
900 T=PEEK(20):REM Time in sixtieths of
a second
910 IF T>20 AND T<40 THEN POSITION 20,0:
? "sh ";SHOTS;" |SCR|    |hts| ";HITS;"
"
920 IF T>40 THEN GOSUB 970:POKE 20,0:REM
 Reset time
930 POKE 53279,8:REM Reset console switc
hes
940 IF PEEK(53279)<7 THEN RUN :REM START
=6
950 GOTO 900
960 REM Following subroutine updates sco
re line
970 POSITION 20,0:? "sh ";SHOTS;" |SCR|
";INT(HITS*100/SHOTS);" |hts| ";HITS;"{L
EFT}{DELETE}";
980 SOUND 0,0,0,0:RETURN
990 REM OBLITERATE STAR
1000 REM POKE 53279,0 generates click on
 internal speaker
1010 FOR L=1 TO 10:POKE SCR,PEEK(53770):
POKE 53279,0:NEXT L:POKE SCR,0
1020 RETURN
```

Apple Version

```
100 REM APPLE SHOOTING STARS
110 REM
120 GR :  HOME
130 PRINT "**** SHOOTING STARS ****
    " : PRINT
140 INPUT "HOW MANY STARS? ";NS
150 IF NS = Ø THEN TEXT : HOME : EN
    D
160 HOME
170 FOR I = 1 TO NS
180 COLOR = INT ( 15 * RND ( 1 ) + ~
    1 )
190 X = INT ( 34 * RND ( 1 ) + 3 ) ~
    : Y = INT( 3Ø * RND( 1 ) )
     : IF SCRN( X , Y ) THEN 1
```

```
    90
200 PLOT X , Y
210 NEXT I
220 REM MAIN LOOP
230 S = 1 : E = 36
240 FOR X = S TO E STEP SGN ( E - S
    )
250 COLOR = 1 : PLOT X , 39 : PLOT ~
    X + 2 , 39
260 COLOR = 2 : PLOT X + 1 , 39 : P
    LOT X + 1 , 38
270 COLOR = 0 : PLOT X - 1 , 39 :PL
    OT X + 3 , 39 :PLOT X , 38
     :PLOT X + 2 , 38
280 IF MS = 0 THEN 390
290 REM FIRE MISSILE
300 COLOR = 0 : PLOT MX , MY
310 Z = PEEK  ( - 16336 )
320 MY = MY - 1 : IF MY < 0 THEN MS
    = 0 : GOTO 390
330 P = SCRN ( MX , MY )
340 Z = PEEK  ( - 16636 )
350 IF P = 0 THEN COLOR = INT ( 15 ~
    * RND ( 1 ) + 1 ) : PLOT M
    X , MY : GOTO 390
360 H = H + 1 : GOSUB 490 : MS = 0 ~
    : COLOR = 0 : PLOT MX , MY

370 IF H = NS THEN 460
380 FOR J = 1 TO 10 : Z = PEEK (  -
     16636 ) : FOR W = 1 TO 2 ~
    :NEXT :Z = PEEK ( - 16636
     ) :NEXT
390 IF PEEK  ( - 16384 ) < 127 THE
    N 440
400 POKE  - 16368 , 0
410 IF MS THEN COLOR = 0 : PLOT MX ~
    , MY
420 MS = 1 : MX = X : MY = 37 : SH ~
    = SH + 1
430 GOSUB 490
440 NEXT
450 S = 37 - S : E = 37 - S : GOTO ~
    240
460 HOME : PRINT "SCORE : ";INT ( H
     * 100 /  SH )
470 PRINT "  PRESS "; : FLASH : PRI
    NT " RETURN"; : NORMAL
480 GET A$ : RUN
490 HOME
500 PRINT " SHOTS FIRED :  "; : INV
    ERSE : PRINT SH; : NORMAL
510 PRINT " SCORE :  "; : FLASH : P
    RINT INT ( H * 100 /  SH )
    ; : NORMAL
520 PRINT "  HITS :  "; : INVERSE :
     PRINT H : NORMAL
530 RETURN
```

PET Version

```
100 REM PET/CBM SHOOTING STARS
110 REM
120 GOTO 180
130 REM SOUND SUBROUTINE
140 POKE S1,K1:POKE S2,K2:POKE S3,S
    V
150 POKE S1,K0:RETURN
160 REM PET/CBM SHOOTING STARS
170 REM
180 PRINT "{CLEAR}**** SHOOTING STA
    RS ****{DOWN}"
190 S1=59467:S2=59466:S3=59464:K1=1
    6:K0=0:K2=51
200 INPUT "HOW MANY STARS";NS
210 IF NS=K0 THEN PRINT "{CLEAR}":E
    ND
220 PRINT "{CLEAR}":CRT=32768:LL=40
    :REM LL=80 FOR 8032
230 FOR I=1 TO NS
240 X=INT((LL-5)*RND(1)+3):Y=INT(20
    *RND(1))
250 IF PEEK(CRT+X+LL*Y)<>32 THEN 24
    0
260 POKE CRT+X+LL*Y,42
270 FOR W=100 TO K0 STEP-10:SV=W:GO
    SUB 130:NEXT W
280 NEXT I
290 S=1:E=LL-4:CH=E+1
300 SHIP$=CHR$(172)+CHR$(177)+CHR$(
    187)
310 FOR X=S TO E STEP SGN(E-S)
320 PRINT"{HOME}{23 DOWN}";
330 PRINT TAB(X);" "SH$" ";
340 IF MS=K0 THEN FOR W=1 TO 10:NEX
    T:GOTO440
350 REM FIRE MISSILE
360 POKE CRT+MX+LL*MY,32
370 SV=200:GOSUB130
380 MY=MY-1:IF MY<K0 THEN MS=K0:GOT
    O 440
390 P=PEEK(CRT+MX+LL*MY)
400 IF P=32 THEN POKE CRT+MX+LL*MY,
    93:GOTO440
410 H=H+1:GOSUB 540:MS=K0:V=K0
420 FOR J=1 TO 9:SV=50:GOSUB130:POK
    ECRT+MX+LL*MY,32+128*V:SV=
    80:GOSUB130:V=1-V:NEXT
430 IF H=NS THEN 500
440 GET K$:IF K$="" THEN 480
450 IF MS THEN POKE CRT+MX+LL*MY,32

460 MS=1:MX=X+1:MY=22:SH=SH+1
```

```
47Ø GOSUB 54Ø
48Ø NEXT
49Ø S=CH-S:E=CH-E:GOTO31Ø
5ØØ PRINT "{CLEAR}SCORE: {REV}";INT
    (H*1ØØ/SH)
51Ø PRINT "{DOWN}PRESS {REV}RETURN{
    OFF} TO PLAY AGAIN:";
52Ø GET A$:IF A$="" THEN 52Ø
53Ø RUN
54Ø PRINT"{HOME}{24 DOWN}";
55Ø PRINT " SHOTS FIRED:";SH;" SCOR
    E:{REV}";INT(H*1ØØ/SH);"{O
    OFF} HITS:";H;
56Ø RETURN
57Ø END
```

©

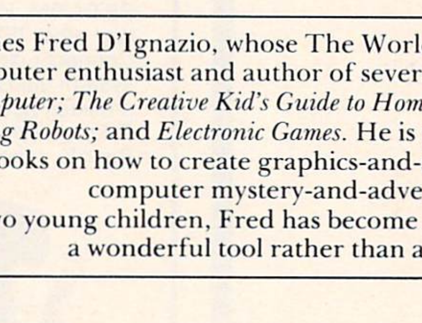
The World Inside The Computer

COMPUTE! welcomes Fred D'Ignazio, whose The World Inside The Computer column will appear each month. Fred is a computer enthusiast and author of several books on computers for young people, including: *Katie and the Computer; The Creative Kid's Guide to Home Computers; Small Computers: Exploring Their Technology and Future; Working Robots;* and *Electronic Games.* 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.

Santa Claus, Subways, And Penguins

Fantasy And The Four Bases

Close your eyes and imagine the baseball diamond at Yankee Stadium. The diamond has four bases. This column also has four bases. What do the four bases represent?



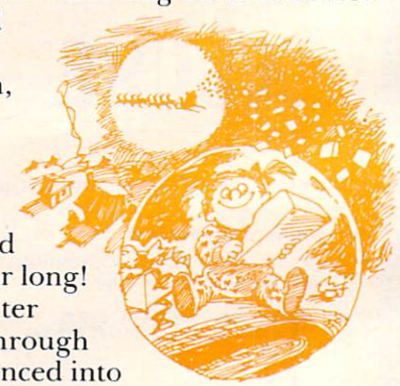
First base is Fun.
Second base is Learning.
Third base is Kids.
Home plate is Computers.

In the middle of the baseball diamond is the pitcher's mound. As author of this column, I am like the pitcher. My ball isn't made out of cork or cowhide. It's pure fantasy. It is the world inside the computer.

This column will explore the many ways kids can use computers to learn and have fun at the same time. And on their own. It will focus on ways computers can be used to foster self-directed learning for each kid's own benefit and enjoyment.

Santa And The Penguins

Our society is feeling the impact of a computer *implosion.* It's as if Santa Claus' bag burst as he flew across the world, and the presents are tumbling to the earth, ending magically under everyone's Christmas tree. And all the presents are small computers. And it is Christmas all year long!



Has your computer just dropped down through the chimney and bounced into your living room? If so, prepare for a knock on your door. Answer it, and you will find that your front lawn is overrun with experts frantically trying to attract your attention. "We are ready to advise you," they say. "We can introduce you to guide books, cook books, checklists, disks and cassettes – whatever you need to operate your home computer."

By all means, let the experts in. And listen to what they have to say.

But don't expect to find an expert here in this column. I'm no expert. I'm not an educator. I'm not a psychologist or a game whiz or a scientist.

Announcing the Printing Breakthrough of the Century: Smith-Corona® TP-1™ Text Printer

- **Low Cost Daisy Wheel Printer**

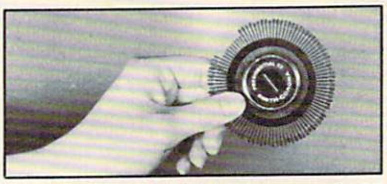
$799.00

- Microprocessor Electronics
- Serial or Parallel Interface
- Simple, Reliable Mechanism

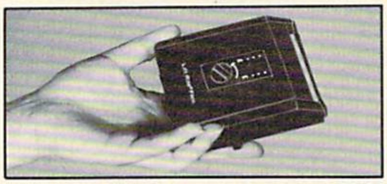
ACT NOW: Limited Supply, Low, Low Cost

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 **$799.**

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**

In PA Call Collect **215 / 433-3366**

MasterCard and Visa Accepted

MasterCard VISA

Who am I?

I'm a writer, and a storyteller. I'm also a parent – of a little boy (Eric) who's almost three, and a not-so-little girl (Catie) who's just turned six.

If I'm no expert, why do I think I can climb on my soapbox and tell you things about computers?

First, because of what I love. I love kids, fantasy, fun, learning, and computers. I think I can build a column around these five loves, a column that will be interesting to anyone who shares my affections.

Second, have you ever seen a penguin rookery? It's a small island or hunk of rock near the bottom of the world, on or near Antarctica. Its most distinguishing characteristic is that it is crowded, packed, jammed with penguins.

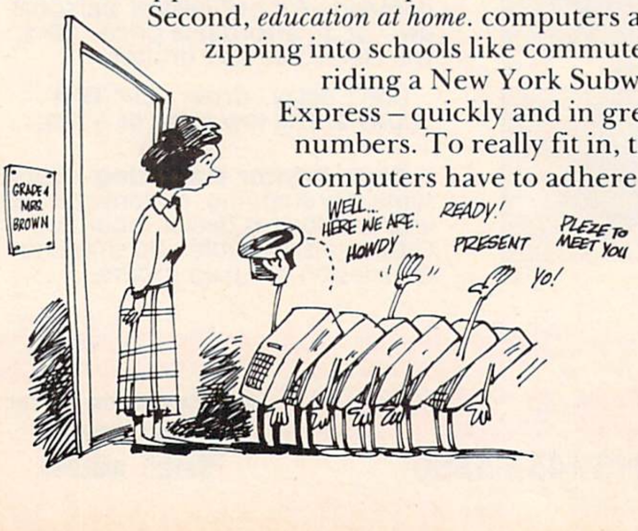
Certain applications of small computers resemble a penguin rookery. There are experts and so-called experts crawling all over, bumping into each other, stealing each other's rocks (for nests), and occupying all the free space.

A penguin rookery is a good place to read about, but I have no intention of visiting one and pretending to be just another penguin (as Jacques Cousteau once did). The same goes for this column. In this column, we (you and I) will search for islands that are less crowded. We will look for computer applications which are vital, but which have not yet received a great deal of attention.

Uncrowded Islands

What are some areas we might explore?

First, *computers for little kids*. Let's say, arbitrarily, from age two up to age eight. We have all seen articles, games, and programs aimed at this group. But not so many as at other groups. As a father and lover of small children, I'd like to explore some new applications for computers here.



Second, *education at home*. computers are zipping into schools like commuters riding a New York Subway Express – quickly and in great numbers. To really fit in, the computers have to adhere to the schools' curriculum. That means, programmed learning, by the experts.

But how about education at home? Self-motivated and self-directed education? Education without a formal curriculum. Education without a game plan or an expert peering over the learner's shoulder. Education that is not just alphabets, multiplication, memorization, and drill. This is another area that fascinates me. I'd like to focus on it in this column.

Third, *fantasy*. The computer playground. A place for kids to act like monkeys and develop bulging muscles of the imagination.

Fantasy is the world of kids. It remains their world until they have heard enough facts and enough drivel to drive fantasy back – back into their mind's back burners and dark corners.

Computers are an immensely powerful tool of directed, solo, or group-oriented fantasy. Just witness the enormous popularity of electronic games.

Fantasy is natural for the child. It has a galvanizing, emotive, and energizing effect on whatever the child does.

A personal computer is like a wizard's staff, a magician's wand. It is a powerful tool for fantasy. It is a tool for the gods, and the gods within us. It is creativity unbounded. According to computer philosopher Greg Yob:

> If you can program your computer, here is a tiny universe in which you can be God. Within the realms of expression that the computer can provide, you can build a world, define its laws, and watch the universe unfold. As your whim dictates, you can intervene at any time, and if you desire, the history of the universe can be changed and rewritten at will. Such a power this is!*

Yet, as they become increasingly popular, small computers are also becoming more focused, more regimented. They are becoming big business. They are becoming standardized and institutionalized.

But they are still ripe for fantasy. Safe in the confines of your own home, where you and your kids are the kings and queens of the realm.

A large dose of fantasy can make your child's use of the computer more playful, his or her learning more creative, more effective and long-lasting.

Fourth, and last, *games*. What kind of games? It doesn't matter, just so they're fun and produce a positive, constructive effect on the young game player.

Welcome to D'Ignazio's Game Arcade

I have eight programmable computers in my home. Running on those computers are over two hundred different games.

I have opened the doors of my home to neighborhood youngsters. As a result, my home has become the local videogame arcade.

It's an offer the kids can't refuse: they get to play games, hour after hour, and it doesn't cost them a single quarter. Needless to say, the parents are delighted with this arrangement. And I get lots of contact with young people, which I enjoy and which helps me with my books.

On any afternoon, there are usually three or more young people present, playing "canned" games or inventing new ones. The young people range in age from two years old (my son) to seventeen years old. Most, however, are between eleven and fourteen.

Afternoons are noisy in the D'Ignazio Arcade. There are squeaks, giggles, beeps, and booms. In the midst of all the silliness and fierce competition, though, two things are apparent. First, the kids are having fun. Second, they are learning.

During the course of this column, I'd like to write about the many ways kids can use computer games to learn and have fun. I think it will make interesting reading.

Adventure, Oracles, Picturebooks & Turtles

So now you have it. I've emptied my pockets and dumped the contents into your lap. As a result, you probably have some idea of this column's flavor and slant.

But where is it going?

In the next few months, here are some of the topics I'd like to explore:

If Your Teacher Were a Turtle

Using Turtle robots to teach young children reading, writing, programming, directionality, etc.

Alice in Computerland

A visit to the world inside the computer. How the youngest children can learn the basics of computer hardware and software. Computer literacy for toddlers. Computers as a second language. Computers as a new mythology.

Robots, Games and Learning

A special chapter on using robots and robot games to teach things to kids. Kids love robots. I'll bet you do, too.

Special Games for Kids with Special Needs

If your child has a physical, emotional, or learning disability, this chapter is for you.

Toddler Adventure

How to wean your toddler from her blanket or bottle and turn her loose on a computer. How to launch your young children on their first adventure – an exciting, educational experience.

The Computer Picturebook

The electronic book is on its way. It will come in the form of a microchip, ready to plug into your *book player*. But until it arrives, you can create your own books on your home computer. This column will show how you and your kids can create electronic picturebooks.

The Computer Oracle

What are your kids' favorite questions? Mine are: Why? What? How? Who? Where? This chapter shows you how you can turn your kids' questions – and your answers – into a game and a growing data base of information pertinent to your children's blossoming interests and knowledge of the world.

Building Models

Kids can fire questions endlessly at you. You try to answer them, but you suspect that your answers flit like butterflies into their ears, ricochet around a bit, then flit back out, only a moment later.

We all learn things best by doing them. How did you first learn about people and their bodies? Did you have a dolly or a teddy bear? How did you first learn about automobiles, monsters, trains, airplanes, and spaceships? The way I learned was by building models.

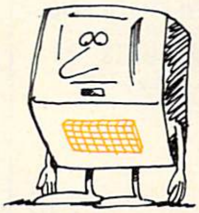
You can answer your kids' questions with models – *computer* models. You don't build them using paper, plastic, or glue, but by creating simulations – miniature replicas of creatures, things, processes, or events pulled from the real world.

But why stick to the real world? Why not copy something directly from your child's dreams or imagination? With a little ingenuity, you can probably build a model of it on your computer. Then your child can run the model, change it, or add to it. Or replace it with something else.

A New Member of Your Family

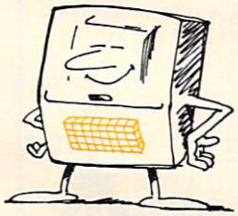
Our family computer (the oldest one) is named 'Ged,' after the wizard hero of Ursula LeGuin's Wonderful Earthsea Trilogy.

When Ged first arrived on our doorstep, he was a dull and simple-minded character. He knew how to edit text, save and copy files, things like that. But that was about all.



So my kids began to teach him. They imagined what kind of personality he ought to have (wise but mischievous and tricky), and we gradually breathed life into what was once a dry and pedestrian computer.

Now we treat Ged like a member of the family. He has his own jokes, his own riddles, tricks, favorite expressions, and peculiarities. He is very much like a real person. That means he is constantly learning – and my children are his teachers.



This column will offer suggestions on how to turn your computer into a member of your family. Or several members, After all, just like a real person, he or she can have many faces.

On the way, I guarantee you, your children will learn many things about computers, programming, intelligence, and personality. And about themselves. In a sense the computer becomes a mirror – it reflects the kids, you, and your entire family.

Making Your Computer More Real

Have you ever read the *Velveteen Rabbit*? It's a story about a nursery toy – a little stuffed bunny – who becomes real to the child who loves him.

Kids tend to anthropomorphize everything. They see a person, a spirit, a gremlin, or creature inside or behind everything that exists. Ironically, this fantasy image of the world seems to make the things in it more real.

Computers can easily become more human-like and more real. We can program a personality into them. We can add a voice synthesizer. We can attach a speech-recognition device. There are many other options.

This column continues the discussion of the previous column: how to make computers more human-like and real. This process can be educational and a lot of fun.

Software And Reviews

Each month, I hope to come up with some original software, usually written in BASIC, and written so you can use it on one of the popular, low-cost computers, such as the Commodore VIC, the Atari 400, or the TI 99/4A.

In addition, I will often review books, magazines, and software that are relevant to that month's column. I want to awaken your curiosity and spark your imagination. I want to startle you and surprise you. But I also want to inform you.

Feedback

Well, that's it. Welcome aboard the column. And while you're reading it each month, I'd like you to ask yourself one question, over and over:

WHAT WOULD *I* BE WRITING ABOUT IF *I* WERE DOING THIS COLUMN?

Then, if answering that question gets you all fired up, drop me a line:

Fred D'Ignazio
c/o **COMPUTE!**
P.O. Box 5406
Greensboro, NC 27403

I promise to write back, and I promise to listen carefully to your suggestions. There's a good chance I can use them in an upcoming column. And, of course, I'll give *you* the credit.

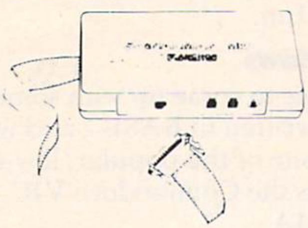
**Gregory Yob, "The Computer as a Gun: Personal Computers and Personal Autonomy," NCC'79* Personal Computing *Proceedings, NY: American Federation of Information Processing Societies (AFIPS), p. 9, 1979.*

The author, his daughter Catie and "Ged" the home computer. Like his namesake in Ursula LeGuin's *Earthsea Trilogy*, Ged is wise. Unlike his namesake, Ged is mischeivous and tricky. Catie is Ged's teacher and is responsible for his personality. ©

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		3.75	25/3.50	100/3.25
2114-L300		3.15	25/2.90	100/2.65
2716 EPROM		7.00	5/6.45	10/5.90
2532 EPROM				14.50
6116 Hitachi 2K x 8 CMOS RAM				**14.50**
4116				**8 for 17**
Zero Insertion Force 24 pin Socket				2.00
6550 RAM (PET 8K)				12.70
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

STAR MODEM Prentice/Livermore Data Systems

RS232 MODEM	**SALE $128**
IEEE 488 MODEM	**SALE $199**
RS232 CCITT	**$170**
IEEE 488 CCITT	**$270**

We carry Apple II+ from Bell & Howell

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
Energy Miser - PET, Apple, Zenith	24.50
Histo-Graph (Boyd) 48K Apple	24.50
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
Basic FORTRAN (Coan)	7.25
CP/M Handbook (with MP/M) Zaks	11.85
Programming the 6502 - Zaks	10.00

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)	1650
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

VIC 20 Color Computer	259
VIC 1515 Graphics Printer	335
VIC 3K RAM	36
VIC 8K RAM	54
VIC 16K RAM	112
VIC 1540 Disk Drive	525
VIC Avengers (Invaders)	24
VIC Jupiter Lander	24
VIC Superslot	24
VIC Super Alien	24
VIC Maze (Victory Software)	12
VIC Cosmic Debris (Victory Software)	9
VIC Amok (UMI)	17
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
OZZ Data Base System for CBM 8032	335
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
Verbatim 8″ Doubled Dens	10/3.65	50/3.55	100/3.40
Verbatim 5″Datalife	10/2.65	50/2.60	100/2.55
(Add 1.00 for Verbatim 5″ plastic storage box)			
BASF 5″ soft	10/2.10	20/2.00	100/1.90
Wabash 5″	10/2.30	50/2.25	100/2.15
Wabash 8″	10/2.40	50/2.35	100/2.25

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
EPSON MX-80 F/T Printer
EPSON MX-70 Printer
EPSON MX-100 Printer

Centronics 739 Printer with dot graphics	575
STARWRITER Daisy Wheel Printer	**1445**
Zenith ZVM-121 Green Phosphor Monitor	119
Amdek Color Monitor	355
DC Hayes Smartmodem	240
Watanabe Intelligent Plotter	1195

ALL BOOK and SOFTWARE PRICES DISCOUNTED
OSBORNE/McGraw-Hill, HAYDEN, SYBEX,etc.

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	695	Microsoft BASIC	72
400 - 16K	339	**MISSILE COMMAND**	**32**
810 Disk Drive	449	**ASTEROIDS**	**32**
825 Printer	629	**STAR RAIDERS**	**32**
850 Interface	175	Space Invaders	36
822 Printer	359	Music Composer	45
Paddle Pair	19	Chess	30
Joystick Pair	19	Super Breakout	30
16K RAM	85	**PAC-MAN**	**36**
Assembler/Editor	46	**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.

The Microsoft version of this program contains REM statements showing the necessary changes for Apple, CBM/PET where specific commands are machine specific. Program 2 is for the Atari.

Grading Exams On A Microcomputer

Kenneth J. Freese
East Meadow, NY

There is certainly nothing new about grading a short answer test with a computer. With the first programs I wrote for this purpose, however, I was frustrated by two problems. I found the usual serial input prompt method for entries much too slow and was unable to look back at previous entries to keep my place and check for accuracy. I developed this program primarily to correct these shortcomings. It is designed for exams with one-character answers. Multiple choice and true-false questions are ideally suited since all input (i.e. A B C D E, T F) can be done with the left hand when touch typing and full advantage of the speed of the program can be taken.

First an overview. The correct answers are placed in the string KEYANS$. Each student's answers are then entered into the string STUDANS$ and compared to KEYANS$ one element at a time. Errors for each student are then displayed and a score is generated. A tally is kept of the number of students getting each question wrong and then displayed after all exams are graded. A "#" is entered to end the answer sequence and, if you get things all fouled up and find yourself at the wrong place on an answer sheet, entering a "&" allows you to start that entry sequence all over again.

The following explains the important routines in the order they are implemented.

LINE 100 – Opens the Keyboard (K:) for an input operation.

SUBROUTINE 3000 – Formats the screen.

SUBROUTINE 4000 and LINE 220 – Reads the keyboard without requiring depressing the RETURN key (see **COMPUTE!**, September-October, 1980, #6 "Reading the ATARI Keyboard on the Fly") and creates the answer string ANS$.

SUBROUTINE 5000 – Places the answers entered onto the screen in the correct position.

LINES 340-380 – Allows correction of erroneous entries one at a time.

LINES 390-580 – Creates STUDANS$ and allows for corrections just as done for KEYANS$ above.

LINES 630-680 – Compares STUDANS$ to KEYANS$ one element at a time.

SUBROUTINE 7000 – Called each time an incorrect answer is encountered. Prints the student's incorrect answer and the correct answer on the screen.

LINE 640 – When the total answer display will cause scrolling of the screen SUBROUTINE 8000 is called and allows the user to view blocks of data on command.

LINE 690 – Calculates and displays student's grade.

SUBROUTINE 6000 – Utilized when "END" is entered for student's name. Displays on the screen the number of students getting each question wrong and prevents scrolling of the screen as above.

Program 1. Atari Version

```
100 REM *** GRADING EXAMS WITH THE ATARI

110 REM *** by KENNETH FREESE
120 OPEN #2,4,0,"K:"
130 POKE 752,1
140 DIM ANS$(101)
150 DIM A$(1),NAME$(10)
160 ? CHR$(125):? :? :? :? "FIRST ENTER
ANSWER KEY"
170 ? :? "WHEN FINISHED ENTER '#'"
180 ? :? "ENTER '&' TO START ENTRY ANEW"

190 ? :? :? :? "PRESS RETURN TO CONTINUE
";:INPUT A$
200 GOSUB 3000
210 GOSUB 4000
220 IF D=0 THEN GOTO 210
230 GOSUB 5000
240 ? ANS$(B,B)
250 IF ANS$(B,B)="#" THEN GOTO 280
260 IF ANS$(B,B)="&" THEN 200
270 B=B+1:GOTO 210
280 NO=B-1
290 POSITION 2,22:? "PRESS RETURN TO CON
TINUE";:INPUT A$
300 DIM KEYANS$(NO),STUDANS$(NO)
```

COLLEGE BOARD SAT* PREP SERIES

TRS-80, APPLE, PET, OSI, ATARI, CP/M, PDP-11

Each program confronts the user with a virtually limitless series of questions and answers. Each is based on past exams and presents material on the same level of difficulty and in the same form used in the S.A.T. Scoring is provided in accordance with the formula used by College Boards.

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. **Independent tests of S.A.T. series performance show a mean total increase of 70 points in students' scores.** Price **$229.95**

GRADUATE RECORD EXAM Series - Educator Edition includes 29 programs covering Vocabulary, Word Relationships, Reading Comprehension, Sentence Completion, Mathematics, Analytical Reasoning and Logical Diagrams. Price **$289.95**

COMPETENCY PROFICIENCY EXAM PREP SERIES

This comprehensive set of programs consists of simulated exam modules, a thorough diagnostic package, and a complete set of instructional programs. It is designed to teach concepts and operations, provide drill and practice and assess achievement levels through pre and post testing. The Competency Exam Preparation Series provides a structured, sequential, curriculum encompassing mathematical, reading and writing instruction.

This program is designed for individual student use or use in a classroom setting. Programs provide optional printer capability covering worksheet generation and performance monitoring. C.E.P.S. are available in three software formats.

Special editions available for California Proficiency Assesment Test and New York Regents Competency Tests. Call for Prices.

Krell's M.I.T. Logo for Apple* $179.95

Complete 4 disk set includes Library Disk, 20 program Tutorial Disk, **Alice In Logoland** and **2 COPIES OF M.I.T. LOGO FOR APPLE***

Odyssey In Time

This spectacular adventure game adds a new dimension of excitement and complexity to **Time Traveler.**

Odyssey In Time includes all the challenges of **Time Traveler** plus 10 additional eras. Each game is different and may be interrupted and saved at any point for later play. **$39.95**

Time Traveler

The best of the adventure games. Confronts the player with complex decision situations and the demand for real time action. Using the Time Machine, players face a challenging series of historical environments. To succeed you must build alliances and struggle with the ruling power. Each game is unique. **$24.95**

Isaac Newton + F.G. Newton

Perhaps the most fascinating and valuable educational game ever devised - **ISAAC NEWTON** challenges the players (1-4) to assemble evidence and discern the underlying "Laws of Nature" that have produced this evidence. **ISAAC NEWTON** is an inductive game that allows players to intervene actively by proposing experiments to determine if new data conform to the "Laws of Nature" in question. Players may set the level of difficulty from simple to fiendishly complex.

Full Graphics Newton. This version of Isaac Newton presents all data in graphic form. Because data is graphic rather than symbolic, this game is suitable for very young children. Players may select difficulty levels challenging to the most skilled adults. **$49.95**

☆NEW☆

☆MICRO-DEUTSCH☆

Micro-Deutsch set includes 24 grammar lessons, covering all material 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. Available for Apple II and PET/CBM. (PET version includes a special foreign language character chip.) Also available soon: MICRO-FRANCAIS, MICRO-ESPANOL, MICRO-IVRIT, MICRO-YIDDISH, MICRO-CHINESE, MICRO-JAPANESE. **$179.95**

☆ NEW ☆

☆Pythagoras and The Dragon☆

Mathematics in a fantasy game context. Based on **The Sword of Zedek, Pythagoras and The Dragon** 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 the player response, confers secret information. With Pythagoras as an ally, the quest to overthrow Ra, The Master of Evil, assumes a new dimension of complexity. Depending on the level chosen, problems range from arithmetic through plane geometry.

32K **$39.95**

Free Bonus with purchase of $300.00 or more: Applesoft Tutor Series

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.

PROGRAMS AVAILABLE FOR TRS-80, APPLE II, PET & ATARI

N.Y.S. residents add sales tax.

All programs require 16K • TRS-80 programs require LEVEL II BASIC • APPLE programs require Applesoft BASIC

*Trademark Apple Computer Corp.

KRELL SOFTWARE CORP.

"The State of the Art in Educational Computing"

21 Millbrook Drive, Stony Brook, N.Y. 11790 (516) 751-5139

```
310 DIM TOTAL(NO)
320 FOR X=1 TO NO:TOTAL(X)=0:NEXT X
330 KEYANS$=ANS$
340 ? CHR$(125):? :? :? "ARE THERE ANY C
ORRECTIONS (Y OR N)";:INPUT A$:IF A$<>"Y
" THEN 390
350 ? CHR$(125):? :? :? "QUESTION #";:IN
PUT Q
360 ? :? "CORRECTED ANSWER  ";:INPUT A$
370 KEYANS$(Q,Q)=A$
380 GOTO 340
390 ? CHR$(125):? :? :? "THE STUDENT'S A
NSWERS WILL NOW"
400 ? "BE ENTERED"
410 ? :? :? "ENTER THE STUDENT'S LAST NA
ME"
420 ? "('END' IF ALL ENTERED)":? :? "
     ";:INPUT NAME$
430 IF NAME$="END" THEN 6000
440 GOSUB 3000
450 GOSUB 4000
460 IF D=0 THEN 450
470 GOSUB 5000
480 ? ANS$(B,B)
490 IF ANS$(B,B)="#" THEN 520
500 IF ANS$(B,B)="&" THEN 440
510 B=B+1:GOTO 450
520 POSITION 2,22:? "PRESS RETURN TO CON
TINUE";:INPUT A$
530 STUDANS$=ANS$
540 ? CHR$(125):? :? :? "ARE THERE ANY C
ORRECTIONS (Y OR N)";:INPUT A$:IF A$<>"Y
" THEN 590
550 ? CHR$(125):? :? :? "QUESTION #";:IN
PUT Q
560 ? :? "CORRECTED ANSWER  ";:INPUT A$
570 STUDANS$(Q,Q)=A$
580 GOTO 540
590 C=0
600 ? CHR$(125):? NAME$:?
610 ? "QUESTION #  STUDENT ANSWER  ANS
WER"
620 ?
630 FOR X=1 TO NO
640 IF PEEK(84)=20 THEN GOSUB 8000:? "QU
ESTION #  STUDENT ANSWER  ANSWER":?
650 IF KEYANS$(X,X)<>STUDANS$(X,X) THEN
670
660 C=C+1:GOTO 680
670 GOSUB 7000
680 NEXT X
690 ? :? ,;"GRADE:";,;INT(100*C/NO+0.5);
"%"
700 ? " PRESS RETURN TO CONTINUE";:INPUT
 A$
710 GOTO 390
3000 B=1:? CHR$(125):POKE 82,1
3010 Z=1
3020 FOR X=1 TO 33 STEP 8:FOR Y=0 TO 19
3025 IF Z=100 THEN POSITION X-1,Y:GOTO 3
035
3030 POSITION X,Y
3035 ? Z:Z=Z+1
3040 NEXT Y:NEXT X
3050 FOR X=3 TO 35 STEP 8:FOR Y=0 TO 19
3060 POSITION X,Y:? ":"
3070 NEXT Y:NEXT X
3080 FOR X=7 TO 31 STEP 8:FOR Y=0 TO 19
3090 POSITION X,Y:? CHR$(2)
3100 NEXT Y:NEXT X
3110 RETURN
4000 D=0
4010 IF PEEK(764)<>255 THEN GET #2,D:ANS
$(B,B)=CHR$(D):POKE 764,255
4020 RETURN
5000 IF B<21 THEN POSITION 5,B-1:RETURN

5010 IF B<41 THEN POSITION 13,B-21:RETUR
N
5020 IF B<61 THEN POSITION 21,B-41:RETUR
N
5030 IF B<81 THEN POSITION 29,B-61:RETUR
N
5040 IF B<101 THEN POSITION 37,B-81:RETU
RN
5050 RETURN
6000 ? CHR$(125):? :? :? "DO YOU WANT A
"
6010 ? "SUMMARY OF ANSWERS (Y OR N)";:IN
PUT A$:IF A$<>"Y" THEN END
6040 ? CHR$(125):? :? "         ANSWER SUM
MARY":?
6050 ? "    QUESTION #";,;"# WRONG":?
6060 FOR X=1 TO NO
6065 IF PEEK(84)=20 THEN GOSUB 8000:? "
   QUESTION #";,;"# WRONG":?
6070 ? ,X;,;TOTAL(X)
6080 NEXT X
6090 END
7000 ? "    ";X;,,;STUDANS$(X,X);,;:? "
";KEYANS$(X,X)
7010 TOTAL(X)=TOTAL(X)+1
7020 RETURN
8000 POKE 84,22:? "PRESS RETURN TO CONTI
NUE";:INPUT A$:? CHR$(125)
8010 RETURN
```

Program 2. Microsoft Version

```
100 REM GRADING EXAMS: MICROSOFT VE
    RSION
145 POKE 59468,12:REM THIS LINE ONL
    Y FOR PET/CBM
```

```
150 GOSUB 9000:PRINT:PRINT:PRINT:PR
    INT "FIRST ENTER ANSWER KE
    Y"
160 PRINT:PRINT"WHEN FINISHED, ENTE
    R '#'"
180 PRINT "ENTER '&' TO START ENTRY
     ANEW"
190 PRINT:PRINT "PRESS RETURN TO CO
    NTINUE"
195 REM FOLLOWING LINE WILL WORK ON
     PET OR APPLE
197 GET A$:IF A$="" THEN 197
200 GOSUB 3000
210 GOSUB 4000
220 IF D=0 THEN 210
230 GOSUB 5000
240 PRINT MID$(AN$,B,1)
250 IF MID$(AN$,B,1)="#" THEN 280
260 IF MID$(AN$,B,1)="&" THEN 200
270 B=B+1:GOTO 210
280 NO=B-1
290 PRINT "PRESS RETURN TO CONTINUE
    ..."
295 GET A$:IF A$="" THEN 295
300 DIM TTL(NO)
330 KEY$=AN$
340 GOSUB 9000:INPUT"ARE THERE ANY ~
    CORRECTIONS (Y OR N) ";A$:
    IFA$<>"Y"THEN390
350 GOSUB 9000:INPUT "QUESTION # ";
    Q
360 PRINT:INPUT "CORRECTED ANSWER "
    ;A$
370 KEY$=LEFT$(KEY$,Q-1)+A$+MID$(KE
    Y$,Q+1)
380 GOTO 340
390 GOSUB 9000:PRINT "THE STUDENT'S
     ANSWERS WILL NOW"
400 PRINT "BE ENTERED."
410 PRINT:PRINT "ENTER THE STUDENT'
    S LAST NAME"
420 PRINT"('END' IF ALL ENTERED)":P
    RINT:INPUT "       ";NAME$
430 IF NAME$="END" THEN 6000
440 GOSUB 3000
450 GOSUB 4000
460 IF D=0 THEN 450
470 GOSUB 5000
480 PRINT MID$(AN$,B,1)
490 IF MID$(AN$,B,1)="#" THEN 520

500 IF MID$(AN$,B,1)="&" THEN 440
510 B=B+1:GOTO 450
520 PRINT "PRESS RETURN TO CONTINUE
    ";
525 GET A$:IF A$="" THEN 525
530 SA$=AN$
540 GOSUB 9000:INPUT"ARE THERE ANY ~
    CORRECTIONS (Y OR N) ";A$:
    IFA$<>"Y"THEN590
550 GOSUB 9000:INPUT "QUESTION # ";
    Q
560 PRINT:INPUT "CORRECTED ANSWER "
    ;A$
570 SA$=LEFT$(SA$,Q-1)+A$+MID$(SA$,
    Q+1)
580 GOTO 450
590 C=0
600 GOSUB 9000:PRINT NAME$:PRINT
610 PRINT"QUESTION #  STUDENT ANSWE
    R  ANSWER"
620 PRINT:L=0
630 FOR X=1 TO NO
640 IF L=20 THEN GOSUB 8000:L=0:PRI
    NT"QUESTION #  STUDENT ANS
    WER  ANSWER"
650 IF MID$(KEY$,X,1)<>MID$(SA$,X,1
    ) THEN L=L+1:GOTO 670
660 C=C+1:GOTO 680
670 GOSUB 7000
680 NEXT X
690 PRINT:PRINT,"GRADE:",INT(100*C/
    NO+.5);"%"
700 PRINT "PRESS RETURN TO CONTINUE
    "
705 GET A$:IF A$="" THEN 705
710 GOTO 390
3000 B=1:GOSUB 9000:PRINT
3010 FOR X=0 TO 19
3020 FOR J=0 TO 4
3030 PRINT TAB(J*8);J*20+X+1;
3040 NEXT J
3050 PRINT:NEXT X
3060 RETURN
4000 GET D$:IF D$="" THEN 4000
4010 D=ASC(D$):AN$=LEFT$(AN$,B-1)+D$
     +MID$(AN$,B+1)
4020 RETURN
5000 REM THIS SUBROUTINE POSITIONS A
     N ABSOLUTE CURSOR
5005 REM IT WILL NEED TO BE CHANGED ~
     FOR YOUR COMPUTER
5007 REM

5010 MY=B-INT((B-1)/20)*20
5020 MX=INT((B-1)/20)*8+5
5030 PRINTLEFT$("{HOME}{24 DOWN}",MY
     +1);TAB(MX);:REM FOR PET
5040 REM POKE 37,MY+1:PRINT TAB(MX);
     :REM FOR APPLE
5050 RETURN
6000 GOSUB 9000:PRINT:PRINT:PRINT"DO
      YOU WANT A"
6010 INPUT "SUMMARY OF ANSWERS (Y OR
      N) ";A$:IF A$<>"Y" THEN E
     ND
6040 GOSUB 9000:PRINT:PRINT "ANSWER ~
     SUMMARY":PRINT
```

```
6050 PRINT "QUESTION #","# WRONG":L=
    0
6060 FOR X=1 TO NO
6065 L=L+1:IF L=20 THEN GOSUB 8000:L
    =0:PRINT"QUESTION #","# WR
    ONG"
6070 PRINT X,TTL(X)
6080 NEXT X
6090 END
7000 PRINT"     ";X,,MID$(SA$,X,1),MI
    D$(KEY$,X,1)
7010 TTL(X)=TTL(X)+1
7020 RETURN
8000 PRINT "PRESS RETURN TO CONTINUE
    "
8005 GETA$:IF A$="" THEN 8005
8010 RETURN
8999 END
9000 REM CLEAR SCREEN SUBROUTINE
9010 REM USE "HOME" OR "CALL-936" FO
    R APPLE
9020 REM USE "FOR I=1 TO 32:PRINT:NE
    XT" FOR OSI
9030 PRINT CHR$(147);:REM FOR PET
9040 RETURN
```

©

COMPUTE! The Resource.

Friends Of The Turtle

P.O. BOX 1317
LOS ALTOS, CALIFORNIA 94022

David D. Thornburg
Los Altos, CA

I want to thank all of you who have taken the time to write me letters in support of Friends of the Turtle. I am busy sending answers to your questions. If you haven't heard from me, please be patient – I answer all my letters personally, so it takes me a while to catch up.

Many of the letters expressed the hope that turtles really are for everyone – expert and neophyte alike.

They are.

My interest in turtle graphics (and in the computer languages LOGO and PILOT) comes from the success I have had teaching programming to audiences as diverse as second graders and professional artists.

Friends of the Turtle is becoming international in scope, with members writing from places as far away as England and Argentina. Horacio Reggini wrote from Argentina to tell me of his work with LOGO and children. As a sign of their level of interest, he sent me a brochure describing the Spanish translation of Papert's Mindstorms (called "Desafio a la mente").

If you don't have Atari PILOT, TI LOGO, or one of the other languages which supports turtle graphics, then watch this column for announcements of new languages – there is a lot of activity in this area.

Several owners of the new IBM computer wrote me for information. While I don't know of any commercial turtle graphics packages for that machine yet, several programmers wrote to tell me they are working on the problem. Any computer with a graphics display should be able to support this graphics environment as long as someone is willing to create the language. I remember back in 1978 when WSFN was released as a turtle graphics language for the PET by the Peninsula School Computer Project (Peninsula Way, Menlo Park, CA 94025). It shouldn't be too hard to get this language to run on the VIC, or on the forthcoming $150 Commodore Ultimax. If you enjoy writing languages, keep me informed of any turtle languages you create.

Visible Vs. Invisible Turtles ...

Some of the more sophisticated languages give the user the option of a visible turtle whose orientation can be seen as it moves around the screen. While experienced programmers usually don't use this feature, it can be quite valuable as a training tool.

Those of you using Atari PILOT know that this language does not contain a visible turtle. Since there wasn't room to incorporate this feature into the language, I decided to create one of my own called VISITURT.

VISITURT is an example of a PILOT-based turtle graphics program which illustrates some important ideas. If you have Atari PILOT, you may want to use this program to introduce turtles to your friends. If you use other languages, you might want to try converting this program to work on your system. In addition to generating a visible turtle, this program automatically alternates between DRAW and TURN commands to allow first-time users to create pictures by entering the distance or angle for each command. While the program itself is designed to be used by beginners, I will assume that those of you who are going to enter this program already understand enough about PILOT to follow the listing.

VISITURT uses three procedures: *ERASE, *ACCEPT, and (of course) *TURTLE. Since these procedures are needed by the main program, we will describe them first.

The ERASE procedure clears the screen, moves the turtle to the origin, makes it turn straight up and draws a fresh copy of the turtle's picture. Here is its listing:

```
*ERASE
GR: GOTO 0,0; TURNTO 0; CLEAR
U: *TURTLE
E:
```

The ACCEPT procedure does two things. It accepts input from the keyboard and places the numeric value of the input in the variable #A. If

the user enters the letter E, this procedure will use the ERASE module to reset the screen. Here is its listing:

```
*ACCEPT
A: #A
M: E
UY: *ERASE
E:
```

The TURTLE procedure has one task – to draw a picture of the turtle on the screen. As you can see from its listing, this procedure is made from a combination of commands which TURN, which GO (move without drawing), and which DRAW. Except for small movements to insure proper centering of the figure, this procedure works this way. First the turtle is moved straight ahead by four units to the edge of a 30-sided body. The body is then drawn by the command 30(DRAW 1; TURN 12). This is the Atari PILOT short hand for "thirty times draw 1 unit and turn 12 degrees." Next, the turtle is repositioned and the head is drawn with a 10-sided polygon. Finally, the turtle is returned to its starting position and orientation. If we didn't make this last step, our turtle wouldn't be very useful. Our goal is to draw a picture of a turtle around the turtle's starting location and to leave the turtle in the same place it was when we started. The listing for this procedure is shown below:

```
*TURTLE
GR: GO 4; TURN -90; GO 1; TURN 180
GR: 30(DRAW 1; TURN 12)
GR: GO 1; TURN 180
GR: 10(DRAW 1; TURN 36;
GR: TURN 90; GO -4
E:
```

Finally we are ready for the main procedure – *VISITURT. Here is its listing:

```
*VISITURT
U: *ERASE
T:  WELCOME TO THE VISIBLE TURTLE
T:
J: *STARTHERE
*MASTERLOOP
T: TURN\
U: *ACCEPT
GR: PEN ERASE
U: *TURTLE
GR: TURN #A
*STARTHERE
GR: PEN YELLOW
U: *TURTLE
T: DRAW\
U: *ACCEPT
GR: PEN ERASE
U: *TURTLE
GR: PEN RED
GR: DRAW #A
GR: PEN YELLOW
U: *TURTLE
J: *MASTERLOOP
E:
```

The idea behind this program is that the turtle is first drawn on the screen. Once the user indicates how much to draw or turn (since these commands appear alternately), the turtle is then erased (by drawing it again with the ERASE pen), a line is drawn, or the turtle is turned, and the picture of the turtle is drawn in its new location with the yellow pen.

If you use Atari PILOT, you might want to enter this program by keying in the *VISITURT procedure first. This way, when you type RUN, the correct procedure will be started first. If you key the procedures in any other order, you will have to *use* the VISITURT procedure (by typing U: *VISITURT) in order to start the program properly.

Those of you who are familiar with PILOT should recognize most of the commands used in this program. The following figures show

Figure 1.

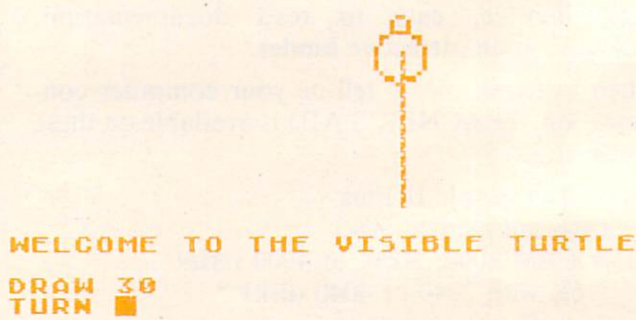


Figure 2.

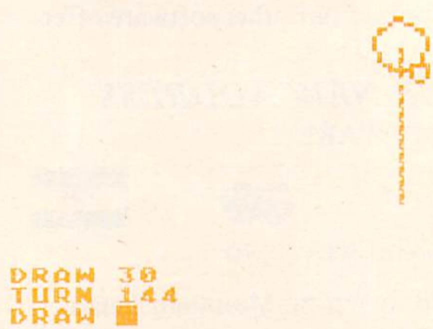


Figure 3.

the VISITURT display during the creation of a 5-pointed star.

While this program was written in Atari PILOT, those of you familiar with other turtle languages should be able to convert it for your system.

Next month we will cover some basic ideas about turtle paths. That topic applies to all turtle implementations and should be of interest to turtle users of all ages.

Until then, please feel free to write me with your programs and pictures. I hope to be able to share your efforts with your fellow readers.

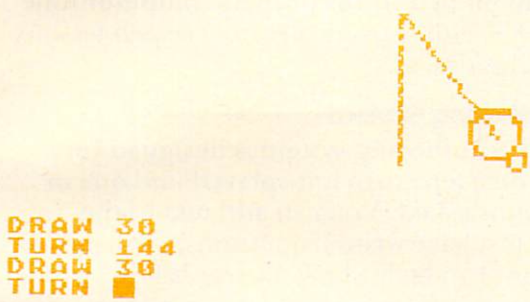


Figure 4.

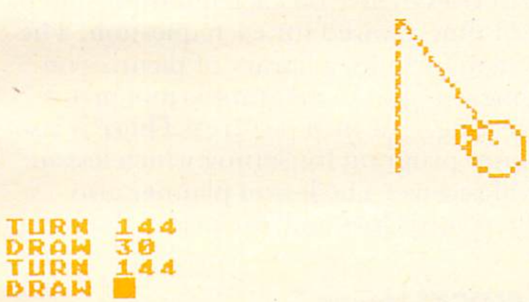


Figure 5.

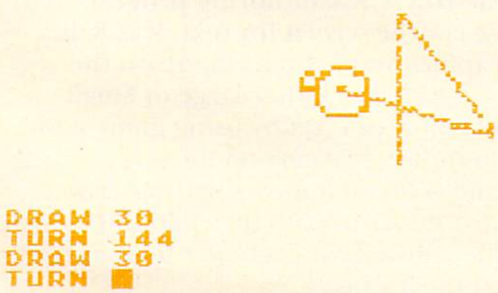


Figure 6.

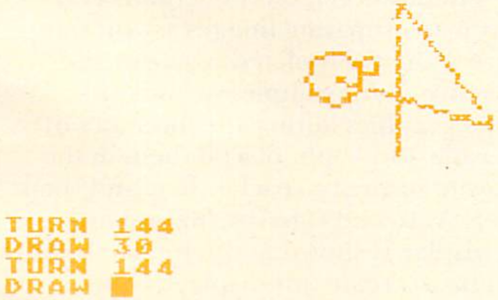


Figure 7.

Figure 8.

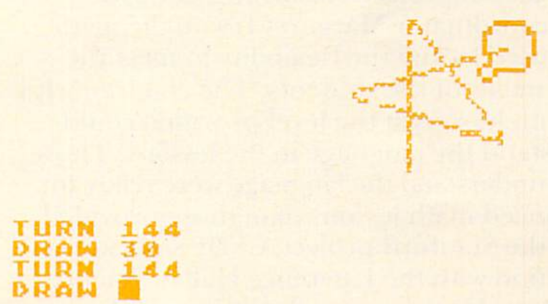


Figure 9.

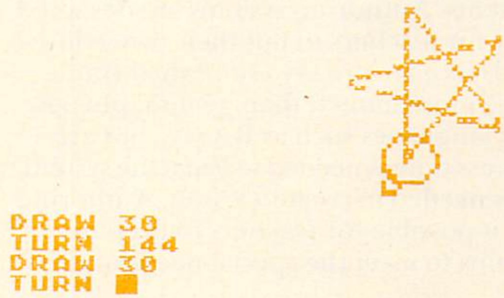


Figure 10.

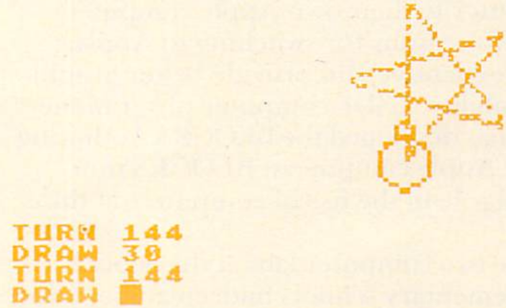


Figure 11.

©

Learning With Computers

Glenn Kleiman and Mary Humphrey
Teaching Tools: Microcomputer Services
Palo Alto, CA

The California School For The Deaf

We recently visited the California School for the Deaf at Fremont, California (CSDF). The School serves 500 hearing impaired students, ranging in age from 4 to 21 years. It uses a total communication approach to teaching deaf children, encouraging both sign language and oral language.

CSDF has been using computers to teach students for 12 years. Their first experience was as part of a Stanford University math computer assisted instruction project. According to CSDF computer coordinator Margaret Irwin, the math programs used lacked the flexibility to meet the individual needs of the students. The children who needed math lessons at the level provided could not understand the language in the lessons. Those who could understand the language were ready for more advanced math lessons than those provided.

After the Stanford project, CSDF staff worked in conjunction with the Lawrence Hall of Science in Berkeley to develop more flexible ways of using time-sharing computers for education. The main emphasis of the project was the development of authoring systems. Authoring systems are designed to make it easy for teachers to put their own educational materials into an already established framework. They are more limited than general purpose programming languages such as BASIC, but are easier to use. Less time is needed to learn the system, and less time is needed to create a lesson. Authoring systems make it possible for teachers to prepare computer lessons to meet the special needs of their students.

In the last few years CSDF has changed from using terminals connected to the Lawrence Hall of Science computer to their own Apple computers. There were two reasons for switching to Apples: They are less expensive and provide better graphic and color capability. CSDF computer programmer Linda Slovick has developed the BLOCKS Authoring Language for Apple computers. BLOCKS now plays a central role in the use of computers at the School.

There are two computer labs at the School, one for the elementary school children and one for the junior high and high school students. About half of the teachers actively use these labs with their classes. There is a much greater demand for computer time than can be met by the available facilities.

In the elementary lab, about half of students' computer time is spent on lessons teachers have authored using the BLOCKS language. The rest of the time is spent using math and problem solving activity programs. In the other lab, the older children spend about two-thirds of their computer time with BLOCKS, either using already created lessons or creating their own.

BLOCKS Authoring System

The BLOCKS authoring system is designed for lessons in which a picture is displayed and one or more questions asked. You can add text to the pictures, or just have written questions if you prefer. The questions can be in short answer, fill-in, multiple choice or true-false formats. BLOCKS lets you design many aspects of a lesson, such as the feedback to be given for correct and incorrect answers, the number of tries to give on each question, and the amount of time allowed for each question. The BLOCKS system includes a library of pictures on disks and some very powerful graphic tool programs for creating your own pictures. There is also a lesson planner program for setting which lesson each child will receive. The lesson planner also automatically records how well each child does on the lessons.

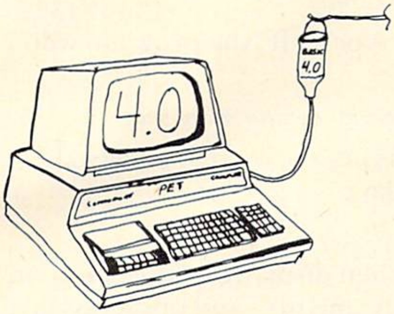
Creating a BLOCKS Lesson

We will describe what is involved in writing a single question of a lesson on states and maps. The first step in using the BLOCKS authoring program is to set a certain area of the screen for text. BLOCKS lets you put graphics and text anywhere on the screen and lets you choose either large or small type fonts. The text area is set by using game paddles to move boundary markers on the screen. Once this is done, a menu appears, letting you easily select the number of tries the child is to receive, the time allowed for each question and other aspects of the lesson.

Next you create a picture on the screen. You select an image from the graphics library disks and use the paddles to position it on the screen. You can combine as many separate images as you want into one picture. When completed, you can add labels to the picture. For example, we took the images from the graphics library for the states of California, Nevada and Utah, placed them in the appropriate positions next to each other, and then added the labels A, B, and C to the three states. The resulting display is shown in the figure. It took about one minute to create this display.

Once the picture is formed, BLOCKS prompts you for a question. Our first question was (prompts

TRANSFORM GRANDPA "PET™" INTO A YOUNG VIBRANT "PET™" BY UPGRADING TO 4.0 BASIC



Socketed P.C. Board that will plug into your ROM Sockets and accepts Commodore's 24 Pin ROMS 4.0 BASIC Interpreters (ROMS not included). When used with I.C. Clip, requires no soldering. Please check number of pins on your ROMS before ordering.

RB-4 (for 28 Pin ROM "PET™")	**$49.95**
RB-4 with I.C. Clip	**69.95**
RB-4A (for 24 Pin ROM "PET™")	**49.95**
RB-4A with I.C. Clip	**69.95**

Please include $1.50 each for shipping and handling.
California residents, please include 6% Sales Tax.

10% User Group Discount for Purchase of 5 or More Units.
Send S.A.S.E. For Free Catalog.

BETTER SOLUTIONS

1163 PETERLYNN DRIVE • SAN DIEGO, CALIFORNIA 92154

"PET" is a trademark of Commodore Business Machines, Inc.

EXPAND YOUR ATARI 400* TO 32K!

FOR ONLY $119.95

Now with the Mosaic Expander and your own Atari 16K RAM board you'll have 32K RAM. Plus all the famous Mosaic quality features: **Four year warranty, bus-compatability, highest quality components, complete instructions. All for** $119.95. Call toll free for your nearest Mosaic dealer.

FOR NOW AND INTO THE FUTURE.

MOSAIC
ELECTRONICS
1-800-547-2807
P.O. Box 748, Oregon City, Oregon 97045 503-655-9574

*Trade Mark of Atari Inc.

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. **Haunted House Adventure**
2. **Income Averaging (Schedule G)**
3. **Hangman with graphics**
4. **Cryptoquote Game**

Plus two of our teaching programs:

5. **Writing Utility Programs**
6. **More on Graphics**

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! Remember MAGATARI is for the ATARI 400/800; APPLETREE is for the APPLE II +; and TRC is for the MODEL I, III, and the TRS-80 COLOR, so be sure to order the right one. Subscription price for cassette is $50 per year, $30 per ½ year and $10 for a trial issue; for diskette $62 per year, $36 per ½ year, and $11 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.
Included in this set of programs are:

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

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 for the TRS-80 COLOR, Model I, III, and Apple II +. However the ATARI 400/800 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, $87 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

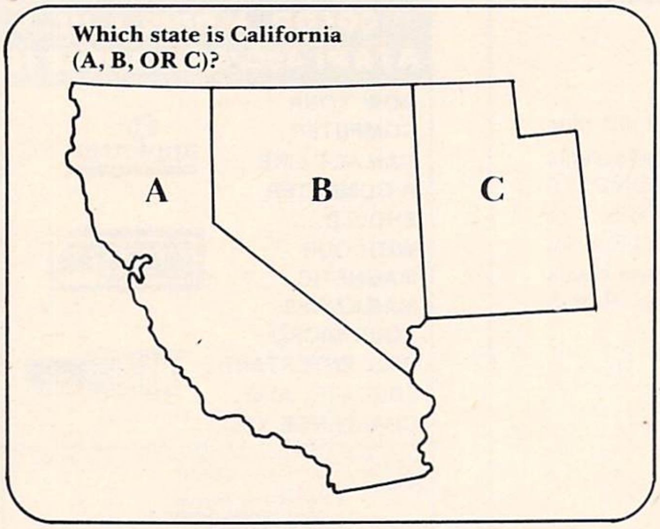
A Futurehouse Company
P.O. Box 3191 Dept. C
Chapel Hill, N.C. 27514
(919) 489-2198 MC & Visa Welcome

Name ____________________
Address ____________________
City/St. ____________________

☐ Programmer's Program ☐ Year Subscription
☐ ½ Year Subscription
☐ Color Accountant ☐ Trial Issue
Type of Computer ____________________
☐ Cassette ☐ Diskette

ATARI, TRS-80, and APPLE are trademarks of the Atari, Tandy, and Apple Corp.
Programmer's Program, TRC, Appletree, and Magatari 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 the APPLE II +, MODEL III, ATARI 400/800.



displayed by the computer are shown in italics):

Question #1: Which state is California (A, B, or C)?

BLOCKS now prompts you to specify the correct answers and the feedback to give. You can easily set the programs to accept as correct one particular word, any one of a list of words, or any answer which contains a given word. For our example lesson we entered:

Correct answer: A
Correct feedback: That's right.

BLOCKS also lets you check for expected incorrect answers so you can give useful feedback. It prompts you for as many expected wrong answers as you would like to check and the appropriate feedback for each. For example, we entered:

Wrong answer #1: B
Wrong feedback #1: No, that's the state known for gambling.
Wrong answer #2: C
Wrong feedback #2: No, that's the state with the Great Salt Lake.

You can also provide a default feedback to appear if none of the expected answers are given. For our example, we used:

Wrong answer #3: (space signifies any other answer)
Wrong feedback #3: Answer A, B or C only.

You can then add more questions about this picture or go on to another picture. When you have finished the lesson, you save it on disk.

A student begins using a lesson by typing his name, his teacher's name, and the lesson name (if not set with the lesson planner). If our lesson was chosen, the display shown in the figure would appear on the screen and the program would wait for the student's answer. If a student types: "I think it is the one on the left," the program will display:

Answer A, B, or C only.

If the student now types: "B" the program will show the message:

No, that's the state known for gambling.

When the student types the correct answer the computer will display:

That's right.

The program will then display the next question or picture. When all the pictures and questions in a lesson are completed, the student is told the number of correct answers and the total time working. This information is automatically recorded on disk by the lesson planner program.

Graphic Tools

Several programs for creating your own pictures are included in the BLOCKS system. These programs are also useful on their own to explore the graphic capabilities of the Apple or to create graphics for use in other programs. Edu-Paint, created by Steve Dompier, is one of the graphic programs which provides a number of options. With the draw option, you use the game paddles like an Etch-A-Sketch to draw on the screen. With the circle option, you use paddles to move a marker to where you want the center of the circle, and to mark any point on the circumference. The program then completes drawing the circle. Similar options let you easily draw lines and boxes. A Fill option lets you fill any shape with any color.

Shaper, created by Linda Slovick, is another graphic tool program. You create shapes from the keyboard using four keys, one to move in each direction. With Shaper, you can almost trace a shape onto the screen point-by-point. Shaper also lets you change the shape's size, rotate it, change colors and relocate it on the screen. The third graphic tool program, Paint Chip, was created by Pete Rowe to let you combine images into a single picture and add text in either large or small type fonts.

BLOCKS Is Available

BLOCKS is a well designed authoring system which makes good use of the capabilities of Apple computers. Like all authoring systems, it is limited in what it can do, but it is an excellent tool for creating picture-question-answer lessons. The graphic tools are fun, whether or not the pictures are to be used in lessons. Since BLOCKS makes it easy to get the computer to do interesting things, it can be used to introduce students to controlling a computer and to some programming concepts. Students can learn to create lessons for their classmates or for younger children. Even the youngest students can work with the graphic tool programs with only a small amount of supervision.

The current version of BLOCKS requires an Apple with Integer BASIC, 48K RAM, two disk drives, and game paddles. Another version is being developed at CSDF which will be compiled so it will run with either Integer or Applesoft BASIC. The new version will require two disk drives for creating lessons, but only one for using them.

The BLOCKS system, with sample lessons, the classroom management system, some useful utility programs, a sample of the graphics library and the three graphic tool programs, is available for $100.00. Not a bad price for 10 disks with comprehensive and clear documentation. The three graphic tool disks are available alone for $30.00 – definitely a best buy. (Edu-Paint, which can be used by itself, is written in Applesoft BASIC.) Groups of disks from the graphics library (which contains 24 disks) can be ordered separately.

Softswap

Where can I get BLOCKS? For the current version, Softswap is the answer. And what is Softswap? We're glad you asked.

Softswap is a distribution center for public domain educational software. In addition to the BLOCKS system they have a large number of programs developed by educators and donated for distribution. They have many disks available for PET, Apple, TRS-80, and Atari computers. Each disk costs $10.00. Two Apple disks of math and logic programs used at the California School for the Deaf are also available.

Softswap is a service of the Microcomputer Center run by the San Mateo County Office of Education and a California group called Computer-Using Educators. For those in Northern California, there is also a center for on-site previewing of commercial software and a collection of books, magazines and other useful materials. This is a very successful resource center: It can serve as a model for others interested in starting one.

For more information about the Microcomputer Center and Softswap please send $1.00 to:

Ann Lathrop, Coordinator
Microcomputer Center
San Mateo County Office of Education
333 Main Street
Redwood City, CA 94063

For more information about the California School for the Deaf Computer Project and future versions of BLOCKS contact:

Margaret Irwin
Computer Project Coordinator
California School for the Deaf
39350 Gallaudet Drive
Fremont, CA 94538

©

Using The VIC Game Paddles

David Malmberg
Fremont, CA

In addition to being able to use the Atari joystick (as described in my article in the Fall, 1981 issue of *Home and Educational COMPUTING!*), the Commodore VIC can use the Atari game paddles. This article provides a tutorial on how these paddles work with the VIC by giving a detailed discussion of a Pong game. This version of Pong can have two human players against one another, or one player against the VIC which has nine skill levels. A game paddle version of the classic game Breakout with three skill levels is also presented. After studying these two programs the reader should be a game paddle "expert" and be capable of easily incorporating game paddles into his own programs.

How Game Paddles Work

For those readers who are unfamiliar with the Atari game paddles, a brief description is in order. These game paddles are included when you buy an Atari home video computer system (the game machine), or may be purchased separately as a peripheral device for the Atari personal computer. The price for a pair of paddles varies between $15 and $20. There are two separate paddle units which attach to a single connector that plugs directly into the VIC game port. Each paddle unit consists of a red "fire" button and a knob that may be turned freely in either direction. The knob is attached to a potentiometer which varies a voltage fed into the VIC's game port. After converting this voltage to a digital value, the VIC is able to know the exact position to which the knob has been turned.

To see how the VIC can read the paddles, plug your paddles into the game port and enter the following short program:

```
10 DD=37154:P1=37151:P2=37152
20 PX=36872:PY=36873
30 POKE DD,127:P=PEEK(P2)AND128
40 FR=-(P=0):POKE DD,255
50 P=PEEK(P1):FL=-((PAND16)=0)
60 VL=PEEK(PX):VR=PEEK(PY)
70 PRINT"CLEAR"FL;VL;FR;VR
200 GOTO30
```

When you RUN this program you should see four numbers in the first row of the screen. The first two numbers correspond to the left paddle and the last two values to the right paddle. FL and FR will be either one or zero depending upon whether the left or right "fire" button is pushed or not. VL and VR will correspond to the knob settings for the left and right paddles respectively. Both VL and VR will vary between 255 and zero as the knobs are turned clockwise from their leftmost position to their rightmost position. You will notice that the knobs are more sensitive than their wide arc would imply, i.e. there is a large band of arc at either extreme of the knob's movement where the values stay at 255 or zero. The actual arc where the values change linearly from zero to 255 is only about one-quarter of a full turn. Although this is a little more sensitive than you might wish, you will find that this will still be enough arc to produce some exciting games.

The reason why this short program actually works is beyond both the scope of this article and the interest level of most readers. I will leave it to Commodore to explain more fully when they issue their documentation on the game paddles. Suffice it to say, it does work!

Controlling Screen Motion

To see how the game paddles can be used to control motion on the VIC's screen, *add* the following lines of code to the above program:

```
2 POKE 36879,27:PRINT"CLEAR":C=4
4 S=256*PEEK(648):A=30720:LL=S
6 IF PEEK(648)=16 THEN A=33792
70 X=21-INT(VL*21/255):Y=22-INT(VR*22/255)
80 NL=S+X+22*Y:CL=NL+A
90 IF FR OR FL THEN C=C+1
100 IF C=8 THEN C=2
110 IF NL=LL THEN 30
120 POKE LL,32:POKE LL+A,1
130 POKE NL,81:POKE CL,C
140 LL=NL
```

You will notice that line 70 above replaces line 70 in the previous program. All of the other lines are new additions.

When you run this program, you will find that the game paddles will move a ball graphic character around the screen at a very rapid pace. Specifically, the left paddle will move the ball from left to right horizontally, and the right paddle will cause vertical movement from top to bottom. Pushing either one of the fire buttons will change the ball's color.

Let's look at this short program in more detail. It not only demonstrates how the game paddles can control motion (and will make following the logic of Pong and Breakout easier) but also contains several useful techniques that will help improve any "action" game you may write. Line 2 sets the border color to cyan and the background color to white, and clears the screen. The variable "C" contains the color of the ball and is initialized to purple.

Lines 4 and 6 will be a useful addition to any program that POKEs characters to the screen. As

VIC-20 SOFTWARE from MIDWEST MICRO

GRAFIX MENAGERIE $11.95

Demonstrate what your $300 miracle can do! Two-program set unleashes VIC's graphics. **SHOWOFF** contains Color Kaleidoscope, Arcade Critters, Custom Fonts, Electronics Schematic, and Music Notation. **PLOTTING** uses dot-plot and line-plot routines to make equations perform computer video-art on your screen. Change equation values and create your own interesting patterns. Plot routines may be easily included in your own programs.

GRAFIX DESIGNER $14.95

Two-program set helps you design custom graphics characters. **GEN/EDIT** displays an enlarged 8x8 square; move the cursor around in it and turn dots on or off to form a character (holds 100). Erase, edit or recall at random. Load **DATAMAKER** when finished designing. Characters automatically become numbered data statements. Save them on tape just like a program. Instructions included for appending to any new or old program. Build libraries of graphics . . . throw away the graph paper!

VIC-PICS .... digitized pictures! $18.95

Nineteen fascinating high-resolution pictures to display on your VIC screen. Created by digitizing video camera images. Includes portraits, models, scenery, and much more. Over 16K points analyzed in each picture. Three styles: hi-contrast, dithered, and colorized. Compatible with both color and B/W sets.

UN-WORD PROCESSOR ($12.95). What else do you call a program which does what a word processor does, yet it is just too simple and easy to use? Enter text . . . Edit . . . List . . . Save . . . just like a program. Send to printer for finished copy. Save paragraphs separately, then merge in any order for printing. Requires only two lines of BASIC to access the M/L routines; over 3K available for text. For VIC-1515 or RS-232 printers.*

BANNER/HEADLINER ($14.95). Two-program set makes GIANT headlines and banners on your printer. **HEADLINER** prints large characters across the page in three sizes. **BANNER** turns the characters sideways, printing continuously down the paper roll. Up to three lines of text, nearly unlimited in length. (How about a ten-foot long "WELCOME HOME"?) For VIC-1515 or RS-232 printers.*

TICKERTAPE ($14.95). Interrupt-driven! Watch your message glide smoothly across the screen in color. Adds motion and interest to any message display. Position on any line, even mix with normal printing. Two built-in character sets: standard and BOLD (or use custom sets from our **LIBRARY VOL. 1**). Message capacity: @ 2K bytes.

LIBRARY VOL. 1 ($12.95). Add style to displays with six full sets of custom character fonts; UPPERCASE, lowercase, numerals, punctuation, etc. Simple to fancy styles. Upper and lower case stored separately; load upper alone to save space . . . load both for a full set. May be used with **TICKERTAPE.**

TERMINAL ($14.95). All machine language! FAST! Enables VIC to emulate a standard terminal. Add a BIZCOMP modem directly (or RS-232 modem with interface*) and access SOURCE, TELENET, or any of the free Bulletin Boards around the country (list included). Special features include reformatting screen data for easier viewing and screen-dump of data to VIC-1515 printer.

DISASSEMBLER ($14.95). Improve your M/L skills as you study the VIC's ROMs. Fast disassembly to screen or printer, with handy hex/dec and dec/hex conversion. Includes key VIC addresses to study. For VIC-1515 or RS-232 printers.* This is the one we use.

RS-232 INTERFACE ($49.95). Get more OUT of your VIC. Plug-in interface communicates with most standard serial printers and modems. Simply plug into User Port; needs no external power. Bi-directional operation. 90 day warranty. Full instructions for use. Includes M/L handshake "wedge."

*RS-232 printers require an interface. See ours above.

MIDWEST MICRO Associates PO Box 6148 K.C. Mo. 64110

Include $1.25 for postage and handling. Missouri residents add 4.6% sales tax.

VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20

NEW VIC-20 SOFTWARE

General Purpose

• Character Generator
Design your own special symbols for any mathematical or engineering application with your own unique figures, create excellent low resolution graphics for your games. The program comes on a cassette with a sample run that includes many strange game creatures.
Price $12.50

• Mailing List
Create files, enter your mailing lists, names, addresses and phone numbers with this user friendly program. Use one keystroke commands to sort by name or zip code, modify, delete or add new entries, and store them on tape. Recall entries by name or phone number. Print out mailing lists. It's simple! Requires at least 10k bytes of memory to be used effectively.
Price $16.00

• Bold Face
This program lets you create titles, banners, and headlines in almost any shape or size. Create big banners, headlines, feetlines, all kinds of lines. Imagine a ten foot long "Happy Birthday", for instance.
Price $9.00

• Text Editor
At last! The indespensable - An easy to use single load text editor for your VIC. Don't wait any longer.
Price $16.00

Mathematics

MathPak

• Complex Function Calculator: Use your VIC as a handy complex number calculator with eight registers to store complex numbers, recall them, operate on them, and restore them.

• Matrix Operations: Add, subtract, multiply, find the inverse and determinant of any matrices with this menu driven program.

• Systems of Linear Equations: Solve any system of linear equations with this menu driven program.

• Function Analysis: Finds zeros, min., max., derivative, or integral of any user supplied polynomial function.

• Statistics I and II: Computes mean and standard deviation. Performs linear and nonlinear regression on a user supplied data set as well as histograms.

• Special Functions: Use trigonometric functions not found in VIC's instruction set with this easy to use program.

• Data and Function Plotting: Plots any user supplied functions or data set with labels on the VIC printer.

Each program $9.50
Complete MathPak $49.50

Educational

- U.S. Quiz Set
- States of the Union
- Revolutionary War Battles
- Civil War Battles

This series of four tutorial programs helps teach important facts about U.S. History. Learn the who, when, where, and how long of two centuries of American Presidents. Learn when each state entered the Union, where it is, what its capital and population are. Learn about the battles of the Revolutionary and Civil wars that changed the course of history, when and where they were fought, who the commanding generals were, and who won!
Single $8.00
Pack $25.00

• Quiz Kit
Create your own quizes with this handy quiz programming kit. Compile the questions and answers and the computer program does the rest. This flexible program allows you to design custom quiz sets for home or classroom use. A great way to teach or to study.
Price $14.00

• Spelling Bee
I. You are the spelling bee, and though that flower you nestle in looks beautiful, if you miss the word, it's deadly. The program contains a dictionary of often misspelled words for you to practice on.
II. Included on the same tape is an advanced version of the above program that allows teachers and parents to make their own lists of hard to spell words to use with this program.
Price $12.50

Games & Entertainment

- Road Rally
- Wipeout
- Hidden Code
- Alien Descent
- Labyrinth
- Pods
- Blackjack
- Kaliedescope

Color Sound Graphics

Each $7.00
Any three $17.00

Order today or ask for our free catalogue;
MDS
47 Dearborn Pl. #18
Goleta, CA 93117

Please add $1.50 for shipping. California residents add 6% sales tax.
VIC-20 is a Registered Trademark of Commodore Business Machines.

VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 you may have read or even experienced, when memory modules (8K or more) are plugged into the VIC, the location of the screen memory and the color matrix move. When you run a program which assumes the screen is in its "normal" position of 7680 and the screen has actually moved (to 4096), the results are frequently a disaster. Line 4 and 6 determine the correct locations for the screen and the color matrix — regardless of the amount of memory that has been added to the VIC. In line 4, the variable "S" will be the starting location of the screen and will be either 7680 or 4096. The variable "A" is the value that must be added to a particular screen location to get its corresponding color matrix location. For example, if we POKE location S with 81 (a ball character), and we POKE location S+A with 4, we would get a purple ball in the "home" corner of the screen. For a "normal" 3.5K VIC, "A" is 30720, i.e., the start of the color matrix is 7680 + 30720 or 38400. *If* the screen moves to 4096, the color matrix will move to 37888 and the value of "A" changes to 33792 — this is the condition given in line 6.

The logic of the Pong program actually starts on line 280 which defines a group of variables that will be used repeatedly later. R is the number of rows. C is the number of columns. NA$(1 and 2) contain the two players names. SP$(1 and 2) contain strings with the cursor control characters needed to position the cursor where each player's name is printed. SC(1 and 2) contain the scores for the two players. In lines 290 and 300 more useful variables are defined. E and F are values used in the calculation of the paddle location. By making these calculations once at the start of the program, and just referencing their variable names later, the speed of the game is increased. The other variables in these lines are either identical to those used in the previous example or their purpose will be obvious when you see how they are used. Line 300 determines the starting locations for the screen, S, and color matrix, S+A.

Line 320 defines three very useful functions. FNA will return the current screen location corresponding to row Y and column X. This will normally be the ball's location. FNB will return the color matrix location corresponding to FNA. FNR(Z) will return a random integer between 0 and Z. Lines 330 to 470 ask the player(s) to specify the options for the play of the game. The variable N is the number of players. If N=1 then the VIC will play the part of the player on the right side of the screen. The variable D is the skill level for the VIC. A value of 1 will play a very poor game. A value of 9 will never miss a shot.

Lines 480 to 520 begin the game by zeroing both scores, randomly deciding who serves first, clearing the screen, drawing the border in row 1, and printing the names and scores in row 0. Line 530 tests the variable SV (which will either be 1 for the left player's serve or 2 for the right player's serve) and branches accordingly.

Lines 540 to 580 handle the left player's serve. Specifically, line 550 reads the status of the left fire button and allows the left paddle and the right paddles to move by the GOSUB's to 120 and 190 respectively. Line 560 jumps back to 550 unless the fire button has been pushed and the serve has been made. Line 570 randomly causes the serve to go upward (DY=-1) or downward (DY= +1). Line 580 puts the ball in front of the current position of the left paddle, sets its forward direction toward the right (DX= +1), sounds a "hit", and branches to line 670 which is the actual play loop.

Lines 590 to 650 are the right player's serve routines. Line 600 tests N for the number of players. If N=1 and the VIC is playing the right side, line 610 waits a random amount of time and then serves. Lines 620 to 650 handle a human player on the right side almost identically to the way that lines 550 to 580 handle the left side player.

Line 670 is the start of the play loop and moves the ball one position in its current direction, via the GOSUB70. Line 690 allows the left paddle to move. Line 700 tests whether it is possible that the ball is about to hit a paddle because it is either in column 2 or column 19, i.e., one column from the paddles. If the answer is no, the ball is allowed to move one more position. This "extra" move in the playing loop makes the game considerably faster – to see how much faster try the game with line 700 deleted.

Line 710 to 790 handle the right player's paddle move. Lines 730 to 760 are for the VIC playing the right paddle. Based on whether a random integer from zero to 9 is greater than the skill level, the VIC will move. If the VIC moves, it moves so its paddle is even vertically with the current position of the ball. Obviously, if the skill level is 9, then the VIC will always move and will never miss the ball. Line 780 makes the paddle move for a human player on the right side.

Line 790 again tests whether the ball is possibly going to hit a paddle because it is in a column next to a paddle. If the answer is no, then the program branches back to the start of the play loop at line 670. If the answer is yes, the ball is either just about to hit the paddle or to miss it. Lines 810 and 820 determine the values for ZZ, the index for the player who will win the point if there is a mix, and ZC, the screen PEEK/POKE character for the paddle. Line 830 slows any "fast" ball down to normal speed.

Lines 840 and 850 test if the ball would hit the paddle if it moved one more position. The variable

Q in line 840 is the screen location of the position next (in the same row) to the ball's current position. If the screen character at location Q is equal to ZC, the appropriate paddle character, the ball is about to hit the middle of the paddle and the program branches to 960. Line 850 performs the same type of test, but for a possible "corner" hit.

If the ball failed both of these tests, it is just about to miss the paddle. In this case lines 870 to 940 move the ball off the field, sound a "miss," update the score, determine the next server (the loser), and branch back to serve again or asks about a new game if that was the winning point.

If the ball is hit by the paddle, line 960 will change its direction, sound a "hit," and move it one position on its flight path. Line 970 will randomly make it a "fast" ball by doubling the ball's X movement, DX. This not only doubles the speed, but also changes the angle of flight. These "fast" balls make the game much more exciting.

After having gone through the previous example, the subroutines used in Pong should be very straightforward. The subroutine at line 70, checks if the ball is about to hit a wall and, if the answer is yes, it causes the ball to bounce. Then this subroutine erases the current ball location and draws it at its new location. The subroutines at 120 and 190 move the left and right paddles respectively. They first check to see if the paddle has moved since the last reading. If the answer is no, they RETURN. If yes, these subroutines erase the last paddle, and draw a paddle at the new location. The subroutine at 260 sounds a "hit" and the subroutine at 270 a "miss."

Breakout

The version of Breakout here also uses the game paddles. You will find it much faster and more exciting than versions which use a joystick or the keyboard and can only move the paddle a column at a time. It has three skill levels and the highest will challenge even the most seasoned arcade malingerers.

The overall program flow and even the variable names are almost identical to Pong. The program is well commented and self-documenting.

Breakout

```
10 REM VIC BREAKOUT USING GAME PAD
   DLE
20 REM BY DAVID MALMBERG
30 REM 43064 VIA MORAGA
40 REM FREMONT, CALIFORNIA 94538
50 GOTO200
60 REM SUB TO MOVE BALL ONE POSITI
   ON
70 IFX=0ORX=21THENDX=-DX:GOSUB190:
   REM BOUNCE IF WALL
80 POKEFNA(0),32:POKEFNC(0),1:REM
   ERASE CURRENT BALL LOCATIO
   N
90 X=X+DX:Y=Y+DY:POKEFNA(0),81:POK
   EFNC(0),4:REM DRAW NEXT BA
   LL LOCATION
100 RETURN
110 REM SUB TO UPDATE PADDLE
120 VL=PEEK(PX):REM READ PADDLE
130 NL=E-INT(VL*F):IFNL=LLTHENRETUR
    N:REM SAME AS LAST LOCATIO
    N
140 Z=S+461+LL:FORI=1TOD:POKEZ+I,32
    :POKEZ+I+A,1:NEXT:REM ERAS
    E OLD PADDLE
150 Z=S+461+NL:FORI=1TOD:POKEZ+I,22
    6:POKEZ+I+A,6:NEXT:REM DRA
    W NEW PADDLE
160 LL=NL:REM UPDATE PADDLE LOCATIO
    N
170 RETURN
180 REM SUB TO SOUND HIT
190 POKEV,15:POKES1,220-3*Y:FORI=1T
    O50:NEXT:POKEV,0:POKES1,0:
    RETURN
200 POKE36879,27:C=22:HI=0:X=RND(-T
    I)
210 REM INPUT PADDLE WIDTH (DIFFICU
    LTY)
220 PRINT"{CLEAR}  WELCOME TO BREAK
    OUT"
230 PRINT"{DOWN}DO YOU WANT A"
240 PRINT"{DOWN}   {REV}1{OFF} - DI
    FFICULT"
250 PRINT"    {REV}2{OFF} - AVERAGE,
     OR"
260 PRINT"    {REV}3{OFF} - EASY GAM
    E ?"
270 A$="":GETA$:IFA$=""THEN270
280 D=VAL(A$):IFD<1ORD>3THEN270
290 E=C-D:F=E/255
300 V=36878:S1=36876:PX=36872:LL=8
310 S=256*PEEK(648):A=30720:IFPEEK(
    648)=16THENA=33792
320 DEF FNA(Z)=S+X+C*Y : DEF FNC(Z)
    =FNA(Z)+A : DEF FNR(Z)=INT
    (Z*RND(1))
330 SC=0:BA=9
340 REM DRAW BRICKS
350 NN=0:PRINT"{CLEAR}":Y=1:FORX=0T
    O21:POKEFNA(0),160:POKEFNC
    (0),3:NEXT
360 FORY=5TO8:FORX=0TO21:POKEFNA(0)
    ,160:POKEFNC(0),Y-1:NEXTX,Y
```

```
370 PRINTCHR$(144)"{HOME}BA";BA;" H
    I";HI;" SC";SC
380 TT=FNR(300):ZZ=TI
390 IFTI-ZZ<TTTHENGOSUB120:GOTO390:
    REM RANDOM START
400 X=5+FNR(10):Y=9:DY=1:DX=-1:IFFN
    R(9)>4THENDX=1:REM NEW BAL
    L DIRECTION
410 POKEFNA(Ø),81:POKEFNC(Ø),4:REM ~
    DRAW NEW BALL
420 REM START OF MAIN LOOP
430 GOSUB70:REM MOVE BALL
440 GOSUB120:REM ERASE OLD PADDLE A
    ND DRAW NEW ONE
450 REM TEST FOR PADDLE HIT OR MISS
460 IFY<>20THEN610:REM NOT NEAR PAD
    DLE
470 Z=FNA(Ø)+C:IFPEEK(Z)=226THEN590
    :REM STRAIGHT HIT
480 IFPEEK(Z+DX)=226THENDX=-DX:GOTO
    590:REM CORNER HIT
490 REM MISS PADDLE
500 GOSUB70:GOSUB70:POKEFNA(Ø),32:P
    OKEFNC(Ø),1
510 POKEV,15:FORI=1TO30:POKES1,200-
    2*I:NEXT:POKEV,Ø:POKES1,Ø
520 BA=BA-1:PRINT"{HOME}BA";BA;" HI
    ";HI;" SC";SC
530 IFBA<>ØTHEN380:REM NEXT BALL
540 PRINT"{HOME}{10 DOWN}      PLAY ~
    AGAIN ?"
550 A$="":GETA$:IFA$=""THEN550
560 IFA$<>"Y"THENEND
570 IFSC>HITHENHI=SC
580 GOTO330
590 GOSUB190:DY=-DY:GOSUB70:GOTO430
    :REM SOUND HIT AND MOVE
600 REM TEST FOR BRICK AREA
610 IFY<4ORY>9THEN710
620 REM IN BRICK AREA
630 IFPEEK(FNA(Ø)+C*DY+DX)<>160THEN
    430:REM NO BRICK NEXT - SO
    NORMAL MOVE
640 REM HIT BRICK NEXT
650 GOSUB70:GOSUB190:SC=SC+9-Y
660 NN=NN+1:IFNN>70THEN350:REM DRAW
    NEW BRICKS
670 PRINT"{HOME}BA";BA;" HI";HI;" S
    C";SC
680 IFFNR(Y+10)>4+3*DYTHENDY=-DY:GO
    TO430:REM BOUNCE BACK
690 GOTO610
700 REM TEST FOR TOP
710 IFY=2THENDY=-DY:GOSUB190:REM BO
    UNCE OFF TOP
720 GOTO430:REM END OF MAIN LOOP
```

Pong

```
10 REM VIC PONG USING GAME PADDLES
20 REM BY DAVID MALMBERG
30 REM 43064 VIA MORAGA
40 REM FREMONT, CALIFORNIA 94538
50 GOTO280
60 REM SUB TO MOVE BALL ONE POSITI
   ON
70 IFY=2ORY=22THENDY=-DY:GOSUB240:
   REM BOUNCE IF WALL
80 POKEFNA(Ø),32:POKEFNC(Ø),1:REM ~
   ERASE CURRENT BALL LOCATIO
   N
90 X=X+DX:Y=Y+DY:POKEFNA(Ø),81:POK
   EFNC(Ø),4:REM DRAW NEXT BA
   LL LOCATION
100 RETURN
110 REM SUB TO UPDATE LEFT PADDLE
120 VL=PEEK(PX):REM READ PADDLE
130 NL=E-INT(VL*F):IFNL=LLTHENRETUR
    N:REM SAME AS LAST LOCATIO
    N
140 Z=S+45+LL*C:FORI=ØTO2:POKEZ+I*C
    ,32:POKEZ+I*C+A,1:NEXT:REM
     ERASE OLD
150 Z=S+45+NL*C:FORI=ØTO2:POKEZ+I*C
    ,225:POKEZ+I*C+A,6:NEXT:RE
    M DRAW NEW
160 LL=NL:REM UPDATE PADDLE LOCATIO
    N
170 RETURN
180 REM SUB TO UPDATE RIGHT PADDLE
190 VR=PEEK(PY):REM READ PADDLE
200 NR=INT(VR*F):IFNR=LRTHENRETURN:
    REM SAME AS LAST LOCATION
210 Z=S+64+LR*C:FORI=ØTO2:POKEZ+I*C
    ,32:POKEZ+I*C+A,1:NEXT:REM
     ERASE OLD
220 Z=S+64+NR*C:FORI=ØTO2:POKEZ+I*C
    ,97:POKEZ+I*C+A,6:NEXT:REM
     DRAW NEW
230 LR=NR:REM UPDATE PADDLE LOCATIO
    N
240 RETURN
250 REM SUB TO SOUND HIT
260 POKEV,15:POKES1,220-3*Y:FORI=1T
    O50:NEXTI:POKEV,Ø:POKES1,Ø
    :RETURN
270 POKEV,15:FORI=1TO30:POKES1,200-
    2*I:NEXTI:POKEV,Ø:POKES1,Ø
    :RETURN
280 POKE36879,27:R=23:C=22:X=RND(-T
    I):DIMNA$(2),SP$(2),SC(2)
290 E=R-5:F=E/255:DD=37154:P1=37151
```

```
    :P2=37152:SP$(1)="{HOME}":
    SP$(2)="{HOME}{12 RIGHT}"
300 V=36878:S1=36876:PX=36872:PY=36
    873:LL=8:LR=8
310 S=256*PEEK(648):A=30720:IFPEEK(
    648)=16THENA=33792
320 DEF FNA(Z)=S+X+C*Y : DEF FNC(Z)
    =FNA(Z)+A : DEF FNR(Z)=INT
    (Z*RND(1))
330 PRINT"{CLEAR} WELCOME TO VIC PO
    NG"
340 PRINT"{DOWN} DO YOU WISH TO PLA
    Y"
350 PRINT"{DOWN} {REV}1{OFF} - THE ~
    VIC, OR"
360 PRINT"{DOWN} {REV}2{OFF} - ANOT
    HER PLAYER ?"
370 A$="":GETA$:IFA$=""THEN370
380 N=VAL(A$):IFN<1ORN>2THEN370
390 PRINT"{DOWN}ENTER NAME(S)"
400 FORI=1TON:PRINT"{DOWN}PLAYER";I
    ;:INPUTNA$(I):NEXTI
410 IFN<>1THEN490
420 NA$(2)="VIC":PRINT"{CLEAR}HOW H
    ARD SHOULD I PLAY"
430 PRINT"{DOWN}       {REV}1{OFF} - E
    ASY"
440 PRINT"{DOWN}          TO"
450 PRINT"{DOWN}       {REV}9{OFF} - I
    MPOSSIBLE"
460 A$="":GETA$:IFA$=""THEN460
470 H=VAL(A$):IFH<1ORH>9THEN460
480 REM BEGINNING SERVE AND GAME
490 B$="              ":NL=9:NR=9:SC(1)
    =0:SC(2)=0:SV=1:IFFNR(9)>4
    THENSV=2
500 PRINT"{CLEAR}":Y=1:FORX=0TO21:P
    OKEFNA(0),160:POKEFNC(0),3
    :NEXTX
510 FORJ=1TO3:PRINTSP$(SV)"SERVICE"
    :GOSUB270:PRINTSP$(SV)B$:G
    OSUB260:NEXTJ
520 PRINT"{HOME}";NA$(1);SC(1);SP$(
    2);NA$(2);SC(2)
530 ON SV GOTO550,600
540 REM LEFT SERVE
550 POKEDD,255:P=PEEK(P1):JL=-((PAN
    D16)=0):GOSUB120:GOSUB190
560 IFJL<>1THEN550
570 DY=1:IFFNR(9)>4THENDY=-1
580 Y=NL+3:X=2:DX=1:GOSUB260:GOTO67
    0
590 REM RIGHT SERVE
600 ON N GOTO610,620
610 NR=FNR(19):GOSUB210:GOTO640:REM
     VIC'S SERVE
620 POKEDD,127:P=PEEK(P2)AND128:JR=
    -(P=0):GOSUB190:GOSUB120:R
    EM HUMAN SERVE
630 IFJR<>1THEN620
640 DY=1:IFFNR(9)>4THENDY=-1
650 POKEDD,255:Y=NR+3:X=19:DX=-1:GO
    SUB260
660 REM START OF PLAY LOOP
670 GOSUB70:REM MOVE BALL
680 REM LEFT MOVE
690 GOSUB120
700 IFX<>2ANDX<>19THENGOSUB70
710 REM RIGHT MOVE
720 ON N GOTO730,780
730 IFH<FNR(9)THEN790:REM COMPUTER ~
    MOVE
740 NR=Y-3:IFNR<0THENNR=0
750 IFNR>18THENNR=18
760 GOSUB210:GOTO790
770 REM HUMAN RIGHT MOVE
780 GOSUB190
790 IFX<>2ANDX<>19THEN670
800 REM TEST FOR PADDLE HIT OR MISS

810 IFX=2THENZZ=2:ZC=225
820 IFX=19THENZZ=1:ZC=97
830 IFABS(DX)=2THENDX=DX/2
840 Q=FNA(0)+DX:IFPEEK(Q)=ZCTHEN960
    :REM STRAIGHT HIT
850 IFPEEK(Q+C*DY)=ZCTHENDY=-DY:GOT
    O960:REM CORNER HIT
860 REM MISS PADDLE
870 GOSUB70:GOSUB70:POKEFNA(0),32:P
    OKEFNC(0),1
880 SC(ZZ)=SC(ZZ)+1:SV=2:IFZZ=2THEN
    SV=1:REM UPDATE SCORE, LOS
    ER SERVES
890 IFSC(ZZ)<>15THEN510:REM NEXT SE
    RVE
900 FORJ=1TO10:PRINTSP$(ZZ)"WINNER!
    ":GOSUB270:PRINTSP$(ZZ)B$:
    GOSUB260:NEXTJ
910 PRINT"{HOME}{11 DOWN}       PLAY ~
    AGAIN ?"
920 A$="":GETA$:IFA$=""THEN920
930 IFA$<>"Y"THENEND
940 GOTO490:REM NEW GAME
950 REM HIT PADDLE
960 DX=-DX:GOSUB260:GOSUB70:REM SOU
    ND HIT AND MOVE
970 IFFNR(9)>4THENDX=2*DX:REM DOUBL
    E X-SPEED
980 GOTO670
```

©

This program is an expansion of a popular Atari, 3-D drawing program which first appeared in **COMPUTE!**, *April, 1981, #11. Here, a number of new commands are added to the original (you don't need to have the earlier version to use this one). Type it in and create some beautiful screen displays.*

Ultracube: Supercube Revisited

David N. Benson
Auburn, IN

If there is one area in which Atari computers shine above all others it is in the field of graphics. Star Raiders is a good example of this. Another good example of this is Steve Steinberg's Supercube (**COMPUTE!**, April '81, #11). For the newer readers who missed it, Supercube enabled the user to make three-dimensional drawings with the joystick. The following program, Ultracube, also enables the user to make three-dimensional drawings, but many new commands have been added and some minor improvements have been made along the way.

When the "new and improved" version is run, a menu appears explaining the commands that are uses in the program. Skip this as the commands will be explained later. After the menu, the computer asks for a dimension for the cube. The larger the number the larger the cube. For now use a number between four and twelve. The computer then asks for the color of the cube. This is two numbers: the first is the hue and the second is the luminance. A table of hue values can be found on page fifty of the Atari Basic Reference Manual.

You are now ready to draw. A cursor of the color you chose will appear on the upper left of the screen and can be moved by the joystick. If you push the joystick button a cube will appear. Notice that if you let go of the button the cursor does not return to the upper left of the screen, or draw an erasing line as it does in Steve's program. However, when the cursor moves into a cube the cursor changes to the background color and back to the cube color when it leaves the cube. So unless there is very little contrast between the color of the cube and the color of the background, the cursor should always be seen.

All of this is fine and good, but eventually you will want to change cube size, colors, correct a mistake or see your creation full screen. If you want to change the size of the cube, type "R" and input the new dimension. To change the color of the cube hit the C and input the new hue and luminance values. Hitting the B will have the same effect on the background. If you wish to clear the screen, hit the W and all graphics shall be banished from the screen. The size and color of the cube will remain the same. If you just want to erase a portion or a mistake in your creation, move the cursor to the undesired part and hit E. The computer will ask you for the size of the eraser. The eraser is a square the size of the number inputted, that, when moved by the joystick, will obliterate all trace of anything that had the misfortune of being in its path. The square is the color of the background; it will not be seen. When finished with the eraser, hit any key and you may begin drawing again. All of this should bring many hours of pleasure. When you eventually finish drawing, hit Q and your masterpiece shall be displayed full-screen.

The magic of the commands is the OPEN statement in line 50 which opens the keyboard for input. The first number is the device code and may be any number from one to five. The next number tells what kind of operation is to be performed; here, the four means input. The third number is required but means absolutely nothing here. The fourth character which must be enclosed in quotation marks tells what is being opened; K means keyboard.

Working hand-in-hand with the OPEN statement is the GET statement in line 74. Whenever a key is hit its ATASCII number is stored in the variable at right. The device code at left may also be any number from one to five but it must be the same number used in the OPEN statement. The next line 75 checks to see if the key pressed is one of the command letters, in which case the computer goes to subroutine 2000.

There are three memory locations used in this program: 77, 752, and 764. The number stored in location 77 increases by one every four seconds a key is not pressed. Pressing a key will reset the value to 0, but if the value reaches 128 the computer goes into the random color switching that we all know and love. Location 72 monitors the status of the cursor: POKEing a one will turn the cursor off and a zero back on again. Location 764 monitors whether a key has been pressed; 255 means that no key has been pressed and anything else means that

- Optimized Systems Software, the company that brought you BASIC A+ and OS/A+, proudly presents another *"PLUS"*.
- ***SpeedRead+*** is the *world's first* speed reading tutor designed for use on your personal computer.
- ***SpeedRead+*** *begins* with training your eyes and mind to function as the incredible precision machine they were meant to be.
- ***SpeedRead+*** goes beyond mere words — it trains you to recognize phrases and columns instantly — it exercises your peripheral vision — it increases your comprehension.
- ***SpeedRead+*** matches *your* pace — now and in the future.

AVAILABLE NOW!

For 16K ATARI® computers with disk, Introductory Price **$59.95**
Coming soon for APPLE II®. Inquire about availability of TRS-80® and cassette versions.

SEE YOUR DEALER TODAY!

Call or write for a descriptive brochure of ***SpeedRead+***™ and other fine OSS products, such as OS/A+, BASIC A+, and TINY C®.

Optimized Systems Software, Inc., 10379 Lansdale Ave., Cupertino, CA 95014, (408) 446-3099

a key has been pressed.

Here is the line-by-line explanation:

8 Sets graphics mode to zero; sets the colors of both the foreground and the background to blue and turns off the cursor.
10-28 Prints menu; anything underlined should be in inverse video.
30 Waits for user to hit a key to continue.
40 Clears character so that it will not appear later.
50 Goes into graphics mode 7; opens keyboard for input.
55 Turns off cursor.
60 Asks user for cube size.
70 Asks user for cube color.
72 Clears screen (to create this character hit ESC and hold CTRL and push CLEAR).
73-75 Checks if user entered a command.
73 Checks if a key has been pressed.
74 The ATASCII value is stored in R.
75 Checks to see if R is one of the command letters; if so, then it goes to subroutine 2000 which processes the command.
80 Checks to see if the joystick button has not been pressed – in which case it goes first to see whether the cursor is in a space occupied by a cube or the background, and then to move the cursor; and second, to a subroutine that checks if the joystick has been moved, and which way.
110 Clears the screen.
120 Resets location 77 to zero. If this location ever reaches 128, the Atari goes into the random color switching (see text).
130 If the joystick button has been pressed, control passes into the cube drawing sequence.
140 Goes to a subroutine that figures out if the joystick has been moved, and which way.
150 Updates x-y coordinates.
200-210 Makes sure that the new x-y coordinates are not out of range.
400 Starts entire procedure over again.
500-520 This subroutine plots the cursor and sets the color of the cursor.
500 The LOCATE statement stores the contents of plot position X,Y in L. This will be a number from 1 to 4 equal to the color register. If this number is 0 or a 4, the cursor is in the background as opposed to being in a cube; the color of the cursor will be the color of the cube.
510 If the cursor is not in the background, then it is in a cube; the color of the cursor will be set to the color of the background.
520 This line plots the cursor on the screen.
600-630 This subroutine is called whenever the color of the cube is changed.
600 If the luminance is below 4, then it is set to 10. This is done so that the front face of the cube is always the darkest.
610 This line calls the subroutine that updates the color of the screen.
700-830 This subroutine draws the cube.
700 The TRAP statement will send control to line 80 if the cube goes off the screen.
710-740 These lines draw the first face of the cube.
750-780 These lines draw the second face of the cube.
790-830 These lines draw the third face of the cube.
1000-1180 This subroutine checks which way the joystick has been moved. For more information on the numbers returned by the joystick, see pages 59-60 in the *Basic Reference Manual.*
2000-2060 These lines process any commands that have been entered by the user.
2000 This line clears the text window.
2010 This line checks to see if the R has been pressed – in which case, the computer asks for the new cube size. The TRAP 2000 sends control back to line 2000 to start the whole procedure over again if the user enters illegal input, and the TRAP 40000 resets the TRAP statement.
2020 If the B has been pressed, the computer asks for the new hue and luminance values for the background color.
2030 If the C has been pressed, the computer asks for the new color of the cube.
2040 If the Q has been pressed, the computer goes into graphics mode 7+16+32. The 7 is self-explanatory, the 16 means full-screen (no text window), and the 32 keeps the screen from going blank. The next instruction, GOSUB 4100, resets the colors to the ones the user chose instead of the default colors. Line 3000 is an endless loop that prevents the computer from going into graphics 0 and wiping out all graphics.
2050 If the W has been pressed, the computer reenters graphics 7, which clears the screen at the same time, surpresses the cursor, and resets the colors.
2060 This is the eraser routine. After the user inputs the size of the eraser, the computer goes to subroutine 4000 which plots the eraser on the screen.
3000 See line 2040 for explanation.
4000-4040 This subroutine plots the eraser on the screen, moves it, and looks for the user to hit a key to disengage it.
4000 This line plots the eraser on the screen and goes to subroutine 1000 to see if the joystick has moved.
4010-4020 See lines 200-210 for explanation.
4030 If a key has not been pressed, the procedure

is done over again.

4040 If a key has been pressed, it is then cleared along with the text window.

4100-4140 This subroutine updates the colors of the cube and background.

The Bug That Got Away

The variable E is used to represent two different things in Ultracube: the luminance value of the background, and the size of the eraser. So, if the user uses the eraser and then changes the color of the cube, the color of the background will change. The solution to this problem is to change the variable E to ERASER in lines 2060 and 4000. The author apologizes for this inconvenience.

The Cube, The Whole Cube, And Nothing But The Cube

For the most part, whenever the joystick button is pressed, a cube will appear. However, there are exceptions, such as when part of the cube runs off the screen and leaves only two faces completed, and the third to the viewer's imagination. This is due to the TRAP 80 in line 700 which sends control to line 80 if an error occurs. If the TRAP 80 statement was not there and an error occurred, the program would come to a grinding halt and a beautiful ERROR 3 (value error) or ERROR 141 (cursor out of range) would appear at the bottom of the screen. To see a full cube everytime, make the following changes:

```
Change 700 TRAP 745
Add    745 TRAP 790
Change 810 TRAP 820:PLOT X+SQ+I,Y-I:
       DRAWTO X+SQ+I,Y+SQ-I+1:TRAP 40000
```

I hope that you have enjoyed this program as much as I have enjoyed improving it. However, there were some things that I would have liked to have added but couldn't, because of lack of expertise. An eraser that can be seen would be nice and so would a machine language subroutine to help speed up the joystick. Any help that anyone could give would be appreciated.

```
8 GRAPHICS 0:SETCOLOR 2,8,0:SETCOLOR 4,8
,0:POKE 752,1
10 POSITION 10,0:PRINT "| ULTRA CUBE MEN
U |":POSITION 5,3:PRINT "|JOYSTICK|":POS
ITION 7,4:PRINT "JOYSTICK:"
12 POSITION 17,4:PRINT "USE TO MOVE CURS
OR":POSITION 7,5:PRINT "BUTTON   : USE TO
 DRAW CUBE":POSITION 5,8
14 PRINT "|KEYBOARD|":POSITION 7,9:PRINT
 "<B> |BACKGROUND COLOR| INPUT HUE":POSI
TION 11,10:PRINT "AND LUMINANCE"
16 POSITION 25,10:PRINT "TO CHANGE COL-"
:POSITION 11,11:PRINT "OR OF BACKGROUND"
:POSITION 7,12:PRINT "<C> |CUBE COLO|"
18 POSITION 20,12:PRINT "|R| INPUT HUE A
ND LU-":POSITION 11,13:PRINT "MANINCE TO
 CHANGE CUBE COLOR":POSITION 7,14
20 PRINT "<E> |ERASE| INPUT SIZE OF ERAS
ER":POSITION 11,15:PRINT "TO ERASE UNDES
IRED PORTIONS":POSITION 7,16
22 PRINT "<R> |REDEFINE| INPUT NEW DIMEN
SION":POSITION 11,17:PRINT "TO CHANGE SI
ZE OF CUBE":POSITION 7,18
24 PRINT "<Q> |QUIT| WHEN PRESSED CREATI
ON":POSITION 11,19:PRINT "WILL BE SEEN F
ULL SCREEN":POSITION 7,20
26 PRINT "<W> |WIPE OUT| USE TO ERASE EN
-":POSITION 11,21:PRINT "TIRE SCREEN":PO
SITION 8,23:PRINT "HIT ANY KEY";
28 POSITION 20,23:PRINT "TO CONTINUE";:P
OKE 764,255
30 IF PEEK(764)=255 THEN 30
40 POKE 764,255
50 GRAPHICS 7:OPEN #1,4,0,"K"
55 POKE 752,1
60 LET R=82:GOSUB 2000
70 LET R=67:GOSUB 2000
72 PRINT "{CLEAR}"
73 IF PEEK(764)=255 THEN 80
74 GET #1,R
75 IF R=82 OR R=66 OR R=67 OR R=81 OR R=
87 OR R=69 THEN GOSUB 2000
80 IF STRIG(0)<>0 THEN GOSUB 500:GOTO 14
0
110 PRINT "{CLEAR}"
120 POKE 77,0
130 IF STRIG(0)=0 THEN GOSUB 700
140 GOSUB 1000
150 X=X+XDIF:Y=Y+YDIF
200 LET X=X+(X<0)-(X>159)
210 LET Y=Y+(Y<0)-(Y>95)
400 GOTO 73
500 LOCATE X,Y,L:LET L1=L:IF L=0 OR L=4
THEN LET L=1:GOTO 520
510 LET L=4
520 COLOR L:PLOT X,Y:FOR I=1 TO 5:NEXT I
:COLOR L1:PLOT X,Y:RETURN
600 IF B<4 THEN B=10
610 GOSUB 4100
620 IF X<10 THEN 120
630 RETURN
700 TRAP 745
710 COLOR 1
720 FOR I=0 TO SQ
730 PLOT X,Y+I:DRAWTO X+SQ,Y+I
740 NEXT I
745 TRAP 790
```

```
750 COLOR 2
760 FOR I=1 TO INT(3*SQ)/5
770 PLOT X+I,Y-I:DRAWTO X+I+SQ,Y-I
780 NEXT I
790 COLOR 3
800 FOR I=1 TO INT(3*SQ)/5
810 TRAP 820:PLOT X+SQ+I,Y-I:DRAWTO X+SQ
+I,Y+SQ-I+1:TRAP 40000
820 NEXT I
830 RETURN
1000 LET WHAT=STICK(0):XDIF=0:YDIF=0
1100 IF WHAT=15 THEN RETURN
1110 IF WHAT=14 THEN YDIF=-1:RETURN
1120 IF WHAT=13 THEN YDIF=1:RETURN
1130 IF WHAT=11 THEN XDIF=-1:RETURN
1140 IF WHAT=10 THEN XDIF=-1:YDIF=-1:RET
URN
1150 IF WHAT=9 THEN YDIF=1:XDIF=-1:RETUR
N
1160 IF WHAT=7 THEN XDIF=1:RETURN
1170 IF WHAT=6 THEN YDIF=-1:XDIF=1:RETUR
N
1180 IF WHAT=5 THEN XDIF=1:YDIF=1:RETURN

2000 PRINT "{CLEAR}":PRINT
2010 IF R=82 THEN PRINT "ENTER NEW DIMEN
SION FOR CUBE";:TRAP 2000:INPUT SQ:TRAP
40000:RETURN
2020 IF R=66 THEN PRINT "INPUT BACKGOUND
 COLOR, LUM.";:TRAP 2000:INPUT D,E:TRAP
40000:SETCOLOR 4,D,E:RETURN
2030 IF R=67 THEN PRINT "INPUT NEW COLOR
 FOR CUBE";:TRAP 2000:INPUT A,B:TRAP 400
00:GOSUB 600:RETURN
2040 IF R=81 THEN GRAPHICS 7+16+32:GOSUB
 4100:GOTO 3000
2050 IF R=87 THEN GRAPHICS 7:POKE 752,1:
GOSUB 4100:RETURN
2060 IF R=69 THEN PRINT "INPUT SIZE OF E
RASER";:TRAP 2000:INPUT ER:TRAP 40000:PO
KE 764,255:GOSUB 4000:RETURN
3000 GOTO 3000
4000 COLOR 4:FOR P=0 TO ER-1:FOR P1=0 TO
 ER-1:PLOT X+P,Y+P1:NEXT P1:NEXT P:GOSUB
 1000
4010 LET X=X+XDIF:LET X=X+(X<0)-(X>159)
4020 LET Y=Y+YDIF:LET Y=Y+(Y<0)-(Y>80)
4030 IF PEEK(764)=255 THEN GOTO 4000
4040 POKE 764,255:PRINT "{CLEAR}":RETURN

4100 SETCOLOR 1,A,B
4110 SETCOLOR 2,A,B-2
4120 SETCOLOR 0,A,B-4
4130 SETCOLOR 4,D,E
4140 RETURN
```

©

Micros With The Handicapped

Marshall Curtis and Susan Semancik
The Delmarva Computer Club

Inability to communicate means isolation from people and experiences which otherwise would help a person develop intellectually, socially, personally, and vocationally. Most nonverbal people without motor-impairment can communicate through handwritten or typewritten messages, or by finger spelling and hand signs. Those with some motor skills can typically communicate by pointing at letters or symbols until a word or concept is understood by the person with whom communication is desired. The letters or symbols are generally arranged on a *communications board,* and pointing can be achieved by the hand or other part of the body, by a mouth-stick or similar device held in the mouth, or by using a flashlight or pointer attached to the user's head.

Besides being a time consuming means of communication, the use of a communications board also requires concentration by both the sender and receiver in order to remember the previous letters and symbols indicated in the message, since no record is automatically kept of the message unless the receiver writes it down as it is formed. These communications boards can be expensive and are not easily modified for individual needs. This is a natural area in which to use a microcomputer since menus of words, statements, or characters can be user-determined and placed in an arrangement determined by individual preference or frequency of use. Also, alternative input devices can be used to take advantage of whatever motor skills the person has. For example, eye-switches, sip-and-puff switches, tongue-switches, cheek-switches, and light sensors can all be similarly connected through the input port of a microcomputer.

The Delmarva Computer Club has developed a communications program for the PET/CBM computer that uses both the keyboard and an alternative input device. The device is a light sensor that can be interfaced to the PET's parallel user port with items purchased from a local electronics store. This allows someone with limited motor control, such as someone with cerebral palsy, to move some part of his/her body over the sensor, which creates a shadow on the sensor. This shadow changes the voltage input to the computer, thus signaling a response which the computer will interpret just as if someone had hit the computer's keyboard.

The Figure shows the current words and characters list used in our program. Note that the characters list includes the alphabet, so that words can be formed even though they do not appear in the words list. Also note that a message has been formed and placed in reverse-field within the bottom four lines of the screen. The computer scans the menu by reverse-fielding or highlighting each column for a period of time determined by the user's initial response time. When an input signal is received on either the keyboard or an alternative device attached to the user port, the computer scans the choices within the current column being highlighted. An input received when a word or character of that column is highlighted causes that word or character to become part of the message being formed at the bottom of the screen.

Three other options besides the selection option are implemented in this current version of the program: one is used to erase the last entry in the message formed at the bottom of the screen; a second one is used to switch between the words list and the characters list so that time isn't wasted waiting for the computer to reach the desired part of the menu by the normal scanning sequence; and a third one is used to interactively increase or decrease the response time in which a selection can be made.

Developing An Individual Program

Certainly we could just publish this communications program and tell how to use it. But, because each potential user of the program has individual needs and requirements for communication, we feel that a single program could not have enough flexibility for widespread use. We will be exploring in a sequence of related articles how such a computer communications program can be developed. We've identified five fundamental areas in this communications program:

1) How to pick a words list, statements list, and/or characters list, from which selections will be made to allow the level of communication desired. Also, how to place this list or lists on the computer's screen.

2) How to indicate which element of the list is selected, including consideration of flexible response times.

3) How to pick an alternative method of input, either using the keyboard differently, or selecting a device to match the motor skills of

the user and interfacing the device to the computer.

4) How to form a message as selections are made, and how to get attention paid to special messages.

5) Besides selecting from a list, how to decide on other options in order to improve the speed of communication, as well as the ease of computer use.

By dedicating a separate article to each of these five areas, we can concentrate on identifying alternatives to be considered in each area. For those readers who are in need of such a communications program, this will provide the means to create an individualized, tailor-made version. We will be allowing time in between articles for reader feedback. Tell us what requirements you, your friends, relations, or neighbors have for such a program. We will then pick the most requested option for that section to show how to incorporate it into the communications program. By carefully analyzing this process in the articles, it should make it possible for readers to do similar adjustments to the program for any other alternatives that meet their needs, as well as imitate the process for other computers. We would also like to have readers share their knowledge and opinions with us about any communications devices or programs so that other readers can be informed of their respective pros and cons.

The next article will be in two months, and will analyze the possibilities involved in picking the words, statements, or characters that become the menu from which communication is derived. The most requested arrangement will be implemented in the first part of the communications program. In exchange for $9.95, we would be glad to send our current version of the communications program for the PET/CBM computer that uses either the keyboard or user port input on four levels to anyone desiring it. This will help some readers who may want a completed program immediately and do not want to participate in an anticipated 12-month genesis of a tailor-made program. It may also be desirable by those readers who want a finished version of the program with which to follow along while we develop alternatives to the five major areas of the program. This way different options can be tried out with the whole program to evaluate what works best for individual circumstances.

But, this is not required. For the end result of the sequence of articles will be a communications program individualized by the greatest needs identified in each area. We hope this approach will accomplish what teaching someone to farm does in comparison to just providing that same person with produce. The teaching method is definitely more time-consuming and difficult, but we feel the end result of helping people with the creation of their own solutions is a desirable and worthwhile endeavor.

The Delmarva Computer Club
P.O. Box 36
Wallops Island, VA 23337

```
YOU ARE  BAD SLEEP WORK  HOSPITAL

I   AND  THE COLD  CANDY BATHROOM ETHROA

MOM HAVE MY  SICK  GAME  PLAYROOM .ISDLH

DAD DID  AT  BOOK  PLAY  SCHOOL   ;CFPMY

WHY IS   HOT FOOD  HOME  INSIDE   ?GWVBX

HOW LOVE BIG DR.   DRUG  OUTSIDE  ,QUKJZ

HI  EAT  OF  WHAT  STORE DOWN     -12345

HE  GET  OUR TIME  BOY   KITCHEN  67890$

WE  GOTO IT  DRINK GIRL  BEDROOM  +%!()

I AM SICK. CALL MY DR. AT 824-3400!
```

©

Here is a program which will let you measure frequencies, or time intervals in microseconds, using only an inexpensive phototransistor or two and your CBM/PET.

Using The PET/CBM In The High School Physics Lab

Peter Spencer
West Hill, Ontario

You are lucky. You are a Science teacher with a microcomputer. Its word processing ability alone justifies its cost to you. But, "what do you use it for in the classroom?" Hmmm...., at present, nothing that would make an administrator or a school trustee want to ensure that all Science departments should have one.

But there are hopeful rumors. You hear of magical digital to analog, and analog to digital (D/A and A/D) happenings in far-off TERC land (1) where someone has been tinkering with a beautiful printed circuit board that will interface to the PET, APPLE, AIM, SYM, or KIM. But, when you phone, the cleverly-synthesized voice says "Microinstrument II is not available yet." Then the budget troll descends on you in a nightmare, caresses your hopes with a trash compactor, and whispers lovingly in your ear "Even if it or any other A/D-D/A board were available, you'd first have to fight three dragons and then wait one and a half years in order to get one." You make an appointment for next February 29th.

In the meantime, what *can* you do with your microcomputer, your yearly issue of two metres of bell wire from Central Stores, and the $3.50 left in your budget?

Surprisingly, quite a bit. You can use your PET/CBM to time, in microseconds, how long a photocell is darkened or illuminated, to time how long a switch is open or closed, to time an object accelerating down an incline, or to measure the frequency of a stroboscope or other flashing light. In other words, you can use it as a photocell or switch-operated interval timer in either pulse or gated modes, and as a frequency meter. The only catch is that to do any of these well requires that you program in machine language. A week of fasting, flagellation, and Rodnay Zaks (2) later, you set to work.

You use a Fairchild FPT-100 phototransistor hooked across pin PA0 and ground of the PET's parallel user port. The FPT-100 is so inexpensive you can buy two of them on your budget, and Fairchild claims a rise time of 2.8 microseconds, which is really good (3).

Figure 1 shows a program that lets you use the phototransistor and computer as a microsecond timer. A disassembly of the machine language timing routine is included in lines 500 to 820. (4) Note that the first thing that MUST be done in any machine language timing routine is to set the interrupt disable (via the SEI instruction) so that the hardware-generated interrupt to refresh the screen every sixtieth of a second is shut off. There is nothing more embarassing than to time ten seconds, and get an answer of eight.

Where did the 43 in line 300 come from, you ask? That is the length of time in microseconds that it takes to traverse the machine language counting loop shown in lines 670 to 790. Multiplying the number of loop traversals (stored in bytes 823, 824, and 825) by the time for one traversal gives the elapsed time.

The machine language routine waits until pin PA0 of the PET's parallel user port is disconnected from ground, that is, until the light beam to the phototransistor is cut. Once into the timing loop, it increments byte $0337 (decimal 823) by one on each run around the loop, putting any carry into byte $0338 and eventually byte $0339. The machine language loop ends when PA0 is grounded again, meaning when the phototransistor is re-illuminated. The interrupt is re-enabled (via the CLI instruction) and control returned to BASIC. Hence, with only a $1.50 phototransistor, some wire, and a printed-circuit board connector, you have a timer accurate to a tenth of a millisecond! Do you remember what you paid for the last millisecond timer you bought for your lab, or what you would have paid had the funds been approved? (5)

The program in Figure 1 can be easily modified to time how long the phototransistor is illuminated rather than darkened. Simply change the 255 in line 210 to a 254, and change the 254 in line 490 to a 255.

If you don't need high accuracy, or if you are timing events longer than the approximately twelve second maximum of the program in Figure 1, you can use the BASIC wait statement, as shown in Figure 2. This second program is accurate to no better than a tenth of a second, but it *will* run for a full twenty four hours. It, too, times how long the phototransistor is darkened.

Want to time a cart rolling down an incline? That is, do you want a timer that is started by something interrupting the light to one phototransistor,

80 COLUMN GRAPHICS

The Integrated Visible Memory for the PET has now been redesigned for the new 12" screen 80 column and forthcoming 40 column PET computers from Commodore. Like earlier MTU units, the new K-1008-43 package mounts inside the PET case for total protection. To make the power and flexibility of the 320 by 200 bit mapped pixel graphics display easily accessible, we have designed the Keyword Graphic Program. This adds 45 graphics commands to Commodore BASIC. If you have been waiting for easy to use, high resolution graphics for your PET, isn't it time you called MTU?

The image on the screen was created by the program below.

```
 10 VISMEM: CLEAR
 20 P=160: Q=100
 30 XP=144: XR=1.5*3.1415927
 40 YP=56: YR=1: ZP=64
 50 XF=XR/XP: YF=YP/YR: ZF=XR/ZP
 60 FOR ZI=-Q TO Q-1
 70 IF ZI<-ZP OR ZI>ZP GOTO 150
 80 ZT=ZI*XP/ZP: ZZ=ZI
 90 XL=INT(.5+SQR(XP*XP-ZT*ZT))
100 FOR XI=-XL TO XL
110 XT=SQR(XI*XI+ZT*ZT)*XF: XX=XI
120 YY=(SIN(XT)+.4*SIN(3*XT))*YF
130 GOSUB 170
140 NEXT XI
150 NEXT ZI
160 STOP
170 X1=XX+ZZ+P
180 Y1=YY-ZZ+Q
190 GMODE 1: MOVE X1,Y1: WRPIX
200 IF Y1=0 GOTO 220
210 GMODE 2: LINE X1,Y1-1,X1,0
220 RETURN
```

K-1008-43M Manual only $10 (credited toward purchase)
k-1008-43 Complete ready to install package $495

Mastercharge and Visa accepted

Write or call today for our full line catalog describing all MTU 6502 products, including our high speed 8" Floppy Disk Controller for up to 4 megabytes of PET storage.

MTU
Micro Technology Unlimited
2806 Hillsborough Street
P.O. Box 12106
Raleigh, NC 27605, U.S.A.
(919) 833-1458

NOW 80 COLUMN PETS CAN HAVE MTU HIGH RESOLUTION GRAPHICS

and then stopped when the something passes by a second phototransistor? My, you have extravagant tastes – now you need *two* illuminated phototransistors wired in series between pin PA0 and ground, plus the program in Figure 3. Now that Figures 1, 2, and 3 have shown you examples of how to go about it in machine language, you may even dream of seven or eight photocells along a track, and automatic display of a distance-time graph.

Can't afford two phototransistors? Never mind, you can still have a frequency meter, a really useful instrument that is usually quite expensive and thus rarely found in the high school lab. You hook your one phototransistor across pin PA0 and ground as before, and run the program in Figure 4.

The machine language part of Figure 4 is slightly more tricky than the first three programs. You must maintain two sets of accumulators – one to count pulses, and one to count loop traversals. When you count a pulse, that is, a bright light on the phototransistor, you must wait until it is over until you count again. To do this, the program stores the value of PA0 in location $188. On the next run around the loop, it compares the new value of PA0 with what is in $188. If there is no change, nothing is added to the count accumulators CLOW and CMID. If there *is* a change and it is from bright to dark (i.e., from PA0 equals 0 to PA0 equals 1), one count is generated by knocking the PA0 bit off (ROR A instruction) into the carry flag, and then dumping the carry flag into register CLOW (ADC #0, STA CLOW instructions).

All this activity can be very time consuming. In order to get the time down to fifty-seven microseconds per loop traversal, the program uses zero-page instructions, with zero-page locations chosen (see lines 1 to 5 in Figure 4B) that are unlikely to be clobbered by BASIC 2.0 in the rest of the program.

You cannot use the program of Figure 4 for pulses with a half cycle shorter than fifty-seven microseconds. That is, not for pulses with a period less than 114 microseconds, which means not for frequencies greater than 8.7 kiloHertz. However, since most of the things – such as calibrating your xenon strobe (which only goes to 300 Hz), or measuring the frequency of a pendulum as a function of amplitude – that you want to do have frequencies nowhere near 8 kHz, you are more than safe. For even higher frequencies, you should investigate the PET's built-in timers $E848 and $E849 (6), but you will probably need a phototransistor with an even shorter rise time than the FPT-100 has.

Now, about February 29th....

References:

(1) *Technical Education Research Center, 8 Eliot Street, Cambridge, MA 02138. Phone (617)547-3890. See the Spring 1980 edition of their publication "Hands On!" for a description of Microinstruments I and II.*

(2) Programming the 6502, *by Rodnay Zaks, Sybex Incorporated, 1978. Also see* MCS6500 Microcomputer Family Programming Manual *by MOS Technology, Inc., 1976.*

(3) *Sometimes such unusual conditions are used to get "good" specifications that realistic behavior is much worse. However, even if the FPT-100's rise time is as long as 28 microseconds, that is still good.*

(4) *This program was originally inspired by the pioneering "Gravity Timer" program of Don Whitewood, Oakwood Collegiate Institute, Toronto, Ontario.*

(5) *Also, the last commercial photogate that I purchased, made by a popular educational supplier, has a rise time of 1000 microseconds! Try using that to time a 3 cm wide object going at 10 m/s.*

(6) PET Machine Language Guide, *by Arnie Lee, Abacus Software, 1979.*

Figure 1.

```
10 REM:  SET UP MACHINE LANGUAGE T
    IMER
20 FOR I=826 TO 888
30 READ A
40 POKE I,A
50 NEXT I
60 :
70 REM  DARK TIMER
80 POKE59468,14
90 PRINT"{CLEAR}{REV}DARK TIMER{OF
    OFF}          PETER SPENCER
    "
100 REM  WRITTEN FOR BASIC 2.Ø
110 REM  USES THE SECOND CASSETTE B
     UFFER
120 PRINT"{DOWN}TIMING STARTS WHEN ~
     THE CONNECTION BETWEEN PAØ
      AND GROUND IS OPEN
130 PRINT"{DOWN}(THAT IS, WHEN THE ~
     PHOTOTRANSISTOR IS IN THE ~
     DARK)"
140 PRINT"{DOWN}TIMING STOPS WHEN T
     HE PAØ-GROUND CONNECTION I
     S SHORTED"
150 PRINT"{DOWN}(THAT IS, WHEN THE ~
     PHOTOTRANSISTOR IS ILLUMIN
     ATED)"
160 :
170 PRINT"{DOWN}TIMER RESETS TO ZER
     O AFTER ABOUT 12 MINUTES"
180 REM:  43*(255+255*256+255*256*2
     56)/1000000 IS LARGEST MIC
     ROSECOND TIME
190 :
200 POKE59459,Ø:REM  SETS PORT FOR ~
     INPUT
210 POKE822,255:REM  SETS MASK FOR ~
     INPUT LINE PAØ
220 :
230 PRINT"{DOWN}{REV}PUSH ANY KEY W
     HEN THE PHOTOTRANSISTOR IS
      PROPERLY CONNECTED"
```

```
240 GET AN$ :IF AN$="" THEN 240
250 :
260 PRINT"{DOWN}{REV}YOU MAY START ~
    NOW
270 :
280 SYS826
290 :
300 MS=43*(PEEK(823)+PEEK(824)*256+
    PEEK(825)*256*256)
310 S=MS/1000000
320 PRINT"{DOWN}{REV}ELAPSED TIME"
330 PRINT MS;"MICROSECONDS "
340 PRINT MS/1000;"MILLISECONDS"
350 PRINT S;"SECONDS"
360 PRINT"{DOWN}NOT ALL DIGITS ARE ~
    SIGNIFICANT!
370 PRINT"TIME THE SAME EVENT SEVER
    AL TIMES TO GET AN ESTIMAT
    E
380 PRINT"OF THE RANGE OF UNCERTAIN
    TY"
390 :
400 PRINT"{DOWN}TYPE {REV}R{OFF} TO
     RESET TIMER
410 GETA$:IFA$<>"R"GOTO410
420 GOTO260
430 :
440 DATA 24,162,1,142,53,3,169,0,14
    1,55,3
450 DATA 141,56,3,141,57,3,120,234,
    234,174,79,232
460 DATA 236,54,3,208,245,24,173,55
    ,3
470 DATA 109,53,3,141,55,3,173,56,3
    ,105,00
480 DATA 141,56,3,173,57,3,105,00,1
    41,57
490 DATA 3,174,79,232,224,254,208,2
    23,88,96
500 :
510 REM:  INITIALIZATION
520 REM:  033A 18          CLC
530 REM:  033B A2 01       LDX #$01
540 REM:  033D 8E 35 03    STX $033
    5
550 REM:  0340 A9 00       LDA #$00
560 REM:  0342 8D 37 03    STA $033
    7
570 REM:  0345 8D 38 03    STA $033
    8
580 REM:  0348 8D 39 03    STA $033
    9
590 :
600 REM:  WAIT UNTIL PA0 GOES HIGH
610 REM:  034B 78 EA EA    SEI
620 REM:  034E AE 4F E8    LDX $E84
    F
630 REM:  0351 EC 36 03    CPX $033
    6
640 REM:  0354 D0 F5       BNE $034
    B
650 :
660 REM:  TIMING LOOP
670 REM:  0356 18          CLC
680 REM:  0357 AD 37 03    LDA $033
    7
690 REM:  035A 6D 35 03    ADC $033
    5
700 REM:  035D 8D 37 03    STA $033
    7
710 REM:  0360 AD 38 03    LDA $033
    8
720 REM:  0363 69 00       ADC #$00
730 REM:  0365 8D 38 03    STA $033
    8
740 REM:  0368 AD 39 03    LDA $033
    9
750 REM:  036B 69 00       ADC #$00
760 REM:  036D 8D 39 03    STA $033
    9
770 REM:  0370 AE 4F E8    LDX $E84
    F
780 REM:  0373 E0 FF       CPX #$FE
790 REM:  0375 D0 DF       BNE $035
    6
800 :
810 REM:  0377 58          CLI
820 REM:  0378 60          RTS
```

Figure 2.

```
10 REM  DARK STOPWATCH
20 PRINT"{CLEAR}{REV}STOPWATCH{OFF
   OFF}                  PETER
    SPENCER
30 REM  TIMES HOW LONG PHOTOCELL I
   S DARKENED
40 REM  INTERCHANGE LINES 100 AND ~
    130 TO TIME PHOTOCELL ILLU
   MINATION
50 PRINT"{DOWN}DISCONNECT PA0 FROM
    GROUND TO START TIMER
60 PRINT"{DOWN}SHORT PA0 TO GROUND
    TO STOP"
70 POKE59459,0 :REM  SET PORT FOR ~
   INPUT
80 PRINT"{DOWN}{REV}PUSH ANY KEY W
   HEN APPARATUS IS PROPERLY ~
```

NOW COMMODORE TALKS. AND LISTENS.

COGNIVOX VIO-1002 is a speech recognition and voice output peripheral for Commodore computers. It offers state-of-the-art capabilities with pleasant sounding natural voice output and recognition performance equal to that of units costing many times more. Its capabilities, the very affordable price and its variety of uses makes it a "must have" peripheral.

Many uses

With COGNIVOX your imagination is not the limit, as the saying goes. It is the starting point. Use if for data entry when the hands and/or eyes are busy. As an educational tool. As an aid to the handicapped. Or as a foreign language translator, a sound effects generator, a telephone dialing device, an answering machine, a talking calculator or clock. Use it in conjunction with the IEEE 488 port to control by voice instruments, plotters, test systems. And all these devices can talk back to you telling you their readings, alarm conditions, even their names. Or use it with a BSR controller interface (see Compute, Oct. 81) to control by voice lights and appliances in the house.

Some specifications.

COGNIVOX can be trained to recognize words or short phrases drawn from a vocabulary of up to 32 entries chosen by the user. To train COGNIVOX to your vocabulary, all you have to do is repeat the entries three times at the prompting of the computer. The voice output vocabulary can also have up to 32 words or phrases of your choice. Data rate is approximately 700 bytes per word. Vocabularies can be stored and recalled from disk, giving in effect unlimited selection of entries.

COGNIVOX VIO-1002 will work with all Commodore computers (old, new and newer ROMs) with at least 16K of RAM. It comes complete with assembled and tested hardware in a quality instrument case, speaker/amplifier, power supply, microphone, cassette with software and detailed user manual.

Easy to use

All you need to get COGNIVOX up and running is to plug it in the user port and load one of the programs supplied. Load the demon program and start talking to your computer right away. Or load one of the games and discover the magic of voice control.

It is easy to write your own talking and listening programs too. A single statement in BASIC is all that you need to say or to recognize a word. Full instructions on how to do it are given in the manual.

Order your COGNIVOX now.

To order by mail send us a check or money order for $249 plus $5 shipping and handling (CA residents add 6% tax). You may also order by phone and charge it to your VISA or MASTERCARD. Call us at (805) 685-1854, 9AM to 5PM, PST, M-F. Foreign orders are welcome, please add 10% for air mail shipping and handling.

VOICETEK

P.O. Box 388, Goleta, CA 93116

What does **COMMODORE** have that Apple, Radio Shack and IBM don't?

backpack™

BATTERY BACKUP SYSTEM

For CBM/PET 2000, 4000, 8000, and 9000 series computers and CBM 4040/8050 dual disk drives. Installs within the cabinets of the computer and disk drive. Recharges continually from the machine's own power supply and automatically supplies 30 minutes (max.) of reserve power during outages. Also eliminates surges and spikes. User installable.

OFFICIALLY APPROVED BY COMMODORE

In Canada call: Van-Hoy Group (604) 542-1138 or (604) 545-0794
In United Kingdom call: Wego Computers (0883) 49235

ETCETERA OF CSC CORPORATION
APEX, NORTH CAROLINA, U.S.A.
(919) 362-4200

SOLD ONLY BY INTELLIGENT COMPUTER DEALERS • DEALER INQUIRIES WELCOME

```
    CONNECTED TO USER PORT"
90 GET IN$ :IF IN$="" GOTO 90
100 WAIT59471,1,Ø :WAITS UNTIL Ø GO
    ES HIGH (IE, PHOTOCELL DAR
    K)
110 TIME$="ØØØØØØ"
120 PRINT"{REV}TIMER ON"
130 WAIT59471,1,1 :WAITS UNTIL Ø IS
     CONNECTED TO GROUND (IE,
    PHOTOCELL BRIGHT)
140 T2=TI
150 PRINT T2;"SIXTIETHS OF A SECOND

160 PRINT T2/60;"SECONDS
170 PRINT"{DOWN}PUSH {REV}R{OFF} TO
     RESET
180 GETA$:IFA$<>"R"GOTO18Ø
190 GOTO8Ø
```

Figure 3-A.

```
10 REM  PULSE-GATED MICROSECOND TI
    MER
20 FOR I= 826 TO 922  :READ M :POK
    E I,M :NEXT I
30 DATA 24 , 169 , Ø , 141 , 151
40 DATA 3 , 141 , 152 , 3 , 141
50 DATA 153 , 3 , 12Ø , 174 , 79
60 DATA 232 , 224 , 254 , 2Ø8 , 24
    9
70 DATA 174 , 79 , 232 , 224 , 255

80 DATA 2Ø8 , 249 , 24 , 173 , 151

90 DATA 3 , 1Ø5 , 1 , 141 , 151
100 DATA 3 , 173 , 152 , 3 , 1Ø5
110 DATA Ø , 141 , 152 , 3 , 173
120 DATA 153 , 3 , 1Ø5 , Ø , 141
130 DATA 153 , 3 , 174 , 79 , 232
140 DATA 224 , 254 , 2Ø8 , 224 , 24

150 DATA 173 , 151 , 3 , 1Ø5 , 1
160 DATA 141 , 151 , 3 , 173 , 152
170 DATA 3 , 1Ø5 , Ø , 141 , 152
180 DATA 3 , 173 , 153 , 3 , 1Ø5
190 DATA Ø , 141 , 153 , 3 , 174
200 DATA 79 , 232 , 224 , 255 , 2Ø8

210 DATA 224 , 88 , 96 , 1Ø7 , 95
220 DATA Ø , 237 , 237 , 237 , 237
230 PRINT"{CLEAR}{REV}PULSE-GATED M
    ICROSECOND TIMER"
240 PRINT"{DOWN}
    PETER SPENCER"
250 REM:  WAIT UNTIL PAØ IS SHORTED
     (IE, PT ILLUMINATED)
260 REM:  TIMING STARTS WHEN PAØ GO
    ES HIGH (IE, PT DARK)
270 REM:  KEEP TIMING AS YOU WAIT F
    OR PAØ TO GO LOW AGAIN
280 REM:  THEN TIME UNTIL PAØ GOES
    HIGH (DARK) AGAIN
290 POKE59459,Ø
300 PRINT"{DOWN}{REV}PUSH ANY KEY W
    HEN THE APPARATUS IS PROPE
    RLY CONNECTED TO THE PORT"

310 GET IN$ :IF IN$="" THEN 31Ø
320 SYS826
330 SE=41*(PEEK(919)+256*PEEK(92Ø)+
    256*256*PEEK(921))
340 PRINT"{DOWN}{REV}ELAPSED TIME
350 PRINTSE;"MICROSECONDS"
360 PRINTSE/1ØØØØØØ;"SECONDS"
370 PRINT"{DOWN}{REV}DIGITS ARE NOT
     ALL SIGNIFICANT"
380 PRINT"TIME THE SAME EVENT SEVER
    AL TIMES TO GET AN ESTIMAT
    E OF THE RANGE OF"
390 PRINT"UNCERTAINTY"
400 PRINT"{DOWN}TYPE {REV}R{OFF} TO
     RESET TIMER
410 GETA$:IFA$<>"R"GOTO41Ø
420 GOTO3ØØ
```

Figure 3-B: Source code for machine language part of Pulse-Gated Microsecond Timer Program.

```
*=826   !pulsed timer
clc
lda #Ø  !initialize registers"
sta lowt
sta midt
sta hight
sei
loop1 ldx 59471   !pa"
      cpx #254    !paØ shorted?"
      bne loop1
start ldx 59471
      cpx #255    !is pao high"
      bne start
loop2 clc !timing loop"
      lda lowt
      adc #1
      sta lowt
      lda midt
      adc #Ø
      sta midt
      lda hight
      adc #Ø
      sta hight
      ldx 59471
      cpx #254     !paØ low again?
      bne loop2
loop3 clc
      lda lowt
      adc #1
      sta lowt
      lda midt
      adc #Ø
      sta midt
```

```
        lda hight
        adc #0
        sta hight
        ldx 59471
        cpx #255      !pa0 high again?
        bne loop3
cli
rts
lowt=*
nop
midt=*
nop
hight=*
nop
.end
```

Figure 4-A.

```
10 REM  FREQUENCY METER PROGRAM
20 FOR I= 826 TO 906  :READ M :POK
   E I,M :NEXT I
30 DATA 120 , 24 , 169 , 0 , 133
40 DATA 138 , 133 , 139 , 133 , 13
   6
50 DATA 133 , 137 , 169 , 254 , 14
   1
60 DATA 67 , 232 , 169 , 1 , 133
70 DATA 188 , 173 , 79 , 232 , 197

80 DATA 188 , 133 , 188 , 240 , 33

90 DATA 234 , 234 , 234 , 106 , 16
   5
100 DATA 136 , 105 , 0 , 133 , 136
110 DATA 165 , 137 , 105 , 0 , 133
120 DATA 137 , 24 , 165 , 138 , 105

130 DATA 1 , 133 , 138 , 165 , 139
140 DATA 105 , 0 , 133 , 139 , 144
150 DATA 216 , 88 , 96 , 169 , 0
160 DATA 76 , 91 , 3 , 16 , 65
170 DATA 128 , 0 , 27 , 126 , 0
180 DATA 0 , 67 , 79 , 0 , 0
190 DATA 0 , 0 , 0 , 70 , 0
200 PRINT"{CLEAR}{DOWN}{REV}FREQUEN
    CY METER{OFF}     PETER SPE
    NCER"
210 PRINT"{DOWN}TYPE {REV}G{OFF}O T
    O START SAMPLING PA0 INPUT
    "
220 GETA$:IFA$<>"G"GOTO220
230 PRINT"RUNNING...
240 SYS826
250 COUNT=PEEK(136)+256*PEEK(137)
260 F=COUNT/((256+256*256)*(57E-6))

270 PRINT"{DOWN}{REV}COUNT
280 PRINTCOUNT
290 PRINT"{DOWN}{REV}FREQUENCY
300 PRINT F;"HERTZ"
310 PRINT"{DOWN}{REV}WARNING:  DIGI
    TS ARE NOT ALL SIGNIFICANT
    "
320 PRINT"MEASURE THE FREQUENCY SEV
    ERAL TIMES"
330 PRINT"TO GET AN ESTIMATE OF THE
     RANGE OF"
340 PRINT"UNCERTAINTY"
350 GOTO 210
```

Figure 4-B: Source code for machine language part of frequency meter program.

```
clow=136"   !first count register
cmid=137"   !second count register
ltime=138"  !first time register
mtime=139"  !second time register
pa0v=188"   !previous PA0 value
*=$033a"
sei"
clc"
lda #0 `!initialization"
  sta ltime"
  sta mtime"
  sta clow"
  sta cmid"
  lda #254"
  sta 59459  !pa0=input"
  lda #1"
  sta pa0v"
loop1=*" !check if PA0 has changed
  lda 59471"
  cmp pa0v"
  sta pa0v"
  beq stall !if not,add 0 to count
  nop"
  nop"
  nop"
else ror a !if yes,kick value into carry
  lda clow"
  adc #0"
  sta clow"
  lda cmid"
  adc #0"
  sta cmid"
  clc  !count loop traversals
  lda ltime"
  adc #1"
  sta ltime"
  lda mtime"
  adc #0"
  sta mtime"
  bcc loop1  !end of count loop traversals
  cli"
  rts"
stall lda #0"
     jmp else"
.end"
```

©

This technique is described for the Apple, PET/CBM, and Atari computers. It could prove quite useful for educational, user-friendly databases, and other programming.

Intelligent Input Routines

R. M. Smythe
Burlington, Canada

First there was the INPUT statement. You used it and put up with error messages. "The user shouldn't have typed a comma anyhow," you said. "Served him right." Then all the magazines came up with their Input Anything Subroutines using the GET statement. You've been using that and found that your programs are remarkably well protected against bad input. You made the input routine ignore inappropriate keystrokes. Now it's time for the next advance: Intelligent Input Subroutines (IIS). Rather than simply ignore faulty data input, the IIS attempts to figure out what the user meant and correct what was typed. Often users won't even know they mistyped an entry!

IIS: PET Examples

There are numerous uses for such a routine. Here is a simple one. The PET's "programmer's keyboard" differs from the standard keyboard in that the top row characters (from exclamation point to brackets) are obtained without shifting. On a typewriter you must shift for these characters. If the touch typist using your program shifts to hit the brackets, a weird graphics symbol appears, which will cause an error or, if caught at the time, have to be deleted.

A sophisticated input subroutine could monitor keystrokes, detect and intercept the shifted bracket, and replace it with the intended symbol before it appears on the screen.

Or suppose you have transferred to the lowercase alternate character set. You are inputting people's names. The IIS can capitalize the first character, and every character after a space, whether the user shifted or not.

But the real power of an IIS is revealed when it's used for specialized formatting of input. To illustrate, I will show examples that apply to education, that could be used to improve programs which have appeared in **COMPUTE!**.

Have you seen copper (II) sulfate printed by a computer this way: CUS04 ? Or students are told to input a quadratic as X2+2X-3 ; or, worse, as X"2+2*x-3. Programs displaying chemical formulae without lower case and subscripts, and mathematical expressions that do not display exponents are not examples of good user-oriented programs. Just by typing these characters: X,2+,2,X,-, and 3 a student *should see on the screen* x^2+2x-3, and the program should understand what was input. Anything else is pandering to the convenience of the computer (and the programmer) in the worst way.

Let's see how to devise an intelligent subroutine to accept algebraic expressions. The first step is to have an *Input Anything Routine* as your foundation. In brief, such a subroutine can use a GET statement to detect keystrokes. Each character is checked against allowed characters. Undesirable characters (e.g. cursor control characters, graphics symbols, etc.) are rejected, and the user sees nothing happen on the screen in response to such a keypress.

Suitable characters are added to the growing collection string (RESP$ in the accompanying examples) and printed to the screen. In the event the key pressed was DELETE, appropriate action must be taken. The routine keeps looping back to the GET statement until a CHR$(13) (RETURN) arrives, whereupon the subroutine returns to the main program. (This whole routine replaces INPUT RESP$ (and any accompanying error checking), but the final product is worth the extra coding!)

We will modify the Input Anything Subroutine in PET Program 1 to accept and display mathematical expressions. The first step is to plan what we want.

1. Allow lower and upper case variables.
2. Raise exponents to the line above.
3. Accept numerals, letters, +, -, (and = if equations are anticipated.)
4. Accept and process DELETE and RETURN.
5. Reject everything else.

Since both coefficients and exponents are numbers, the computer must find some way to tell the difference. It is obvious that exponents always follow variables, which are letters (we won't allow exponents on the coefficients) so the IIS watches for numbers following letters.

One way it could do this is to set a flag (VAR=1) when a letter is hit. If the flag is set when a number is typed, an exponent is intended. A +, -, or RETURN could reset the flag (VAR=0). This is very flexible, but complicated, especially when you realize that you must provide for "unsetting" the flag if the user deletes a just-entered variable to

correct an error. A routine which uses flags to advantage appears later.

In this case, if we restrict exponents to one digit, we can use a simpler method. When a number comes in from the keyboard, we need only look at the last character in RESP$. If it is a letter, we raise the exponent. This is done by printing a cursor-up, the number, then a cursor-down to return us to the input line.

The routine is in statements 1000-1290 of PET Program 2. It is contained in a little program to illustrate its use. Note how alternate correct answers can be provided. Unless an Intelligent Printing Subroutine (which can be part of the Intelligent Input Subroutine) is used, you must also read in a form of the answer containing cursor controls for screen display. One thing you must do when you code for the pressing of the DELETE key is to provide for erasure of the character on the line above the cursor as well as just to the left on the same line: the character to be removed might be an exponent.

In PET Program 3 we use an IIS for chemical formulae. Here we put any numeral on the line below. You can make the routine more flexible. For example, to allow for hydrates, which require non-subscripted coefficients, the routine could check for a dot (period) immediately preceding the arrival of a number, and act accordingly.

PET Program 4 shows an IIS for going the other way – formula to name. Here the use of flags is demonstrated. Normally we use lowercase, but, after brackets are opened, the routine promotes everything to uppercase whether the user shifts or not. The bracket flag (BFLAG%) is reset when brackets are closed. Note also how the routine makes allowances if the trained typist, by habit, shifts to hit brackets.

Not only will the use of Intelligent Input Subroutines make your programs more user-proof, but well designed subroutines can also give them a professional look. Your programs should always adapt to the user, not force them to conform to the computer.

Notes regarding the listings:

1. Spaces were inserted for clarity. If they are omitted, the statements will fit into 80 character lines, except for a couple of statements with extra-long REMs which you will have to truncate.

2. The letters "REM" were used only where necessary. Note the places where you can make REM-less comments (save space!).

Apple Routines

Here are some designs for Apple routines.

There have been some excellent "input anything" subroutines for Apple published. The problem, of course, was to get around the EXTRA IGNORED response and accompanying loss of part of the input when a comma or colon was included. One of the shortest I've seen is by Ben Colley (*NIBBLE*, volume 2, number 1, page 59).

Program 1 (Apple Version) shows the BASIC Input Anything Subroutine in action. The subroutine starts at line 100, and is amplified by many REMarks. Every character entered is collected in R1$. A count is made of characters, R1. Usually, of course, R1 will equal the length of R1$., but, if there is a deletion, instead of lopping off the last character of R1$, we decrease R1. By doing this we can regain previously typed in characters, as we must if we are allowing for the use of the right-arrow key. We just increase R1 again. There is a problem with providing for deleting, though. Since control characters do not show on the screen, if you have typed a control character in your string, and do some deletions, then what is on the screen might not be what is in the final version of R1$, which becomes RESP$ upon exiting from the subroutine. This is actually an Input *almost* Anything Subroutine, because line 155 rejects control characters. However, you can do it one better by using an IIS that actually prints the control characters, if there is a need for them, in inverse video.

Sometimes it is useful to allow the user to exit from a routine without finishing the input: for example, the user might make an incorrect choice at a menu, wind up in an undesired routine, and want out. Line 120 accomplishes this by using the ESCape key. Alternatively, line 220, which rejects a RETURN if there has been no input, can be the early exit. Change it to IF R1=0 THEN RETURN and check for R1 equalling zero in line 1010.

Program 2 is an IIS designed around a simpler Input Anything Subroutine. This time, when a character is deleted using the left-arrow key, it is erased on the screen. Thus there is no need to save characters for use of the right-arrow key. We can fill RESP$ directly, without the use of the intermediary R1$ and character counter R1.

This IIS accepts and displays chemical formulae. Called by the main program at line 10010, it begins at line 1000. The first job is to look for a RETURN, in line 1050. Letters are processed in lines 1050 and 1060. In lines 1070 to 1090, numbers are placed in the line below, as subscripts should be. First the vertical cursor position is found (CV%), and we VTAB to the line below it. (Mathematicians: just change the + to a minus in the VTAB expression to put your exponents above the line. You will have to check that the previous character input (the last one in RESP$) was a letter so you'll be sure that the number was not a coefficient.) Next, the

CBM/PET INTERFACES

RS-232 SERIAL PRINTER INTERFACE – addressable – baud rates to 9600 – switch selectable upper/lower, lower/upper case – works with WORDPRO, BASIC and other software – includes case and power supply.

MODEL – ADA1450 149.00

CENTRONICS/NEC PARALLEL INTERFACE - addressable – high speed – switch selectable upper/lower, lower/upper case – works with WORDPRO, BASIC and other software – has Centronics 36 pin ribbon connector at end of cable.

MODEL – ADA1600 129.00

CENTRONICS 730/737 PARALLEL INTERFACE – as above but with Centronics card edge connector at end of cable.

MODEL – ADA730 129.00

COMMUNICATIONS INTERFACE WITH SERIAL AND PARALLEL PORTS – addressable – software driven – true ASCII conversion – selectable reversal of upper-lower case – baud rates to 9600 – half or full duplex – X-ON, X-OFF – selectable carriage return delay – 32 character buffer – centronics compatible — much more.

MODEL – SADI 295.00

ANALOG TO DIGITAL CONVERTER – 16 channels – 0 to 5.12 volt input voltage range – resolution is 20 millivolts per count – conversion time is less than 100 microseconds per channel.

MODEL – PETSET1 295.00

REMOTE CONTROLLER WITH CLOCK/CALENDAR – controls up to 256 devices using the BSR X10 remote control receivers – 8 digital inputs, TTL levels or switch closure — 8 digital outputs, TTL levels.

MODEL – PETSET2 295.00

All prices are in US dollars for 120VAC.
Prices on 220 VAC slightly higher.
Allow $5.00 shipping & handling, foreign orders add 10% for AIR postage.
Connecticut residents add 7½% sales tax.
All prices and specifications subject to change without notice.
Our 30 day money back trial period applies.
MASTER CHARGE/VISA accepted.

MENTION THIS MAGAZINE WITH YOUR ORDER AND DEDUCT 5% FROM TOTAL.

IN CANADA order from: Batteries Included, Ltd., 71 McCaul Street, F6 Toronto, Canada M5T2X1, (416)596-1405.
IN THE USA order from your local dealer or direct: Connecticut microComputer, Inc., 34 Del Mar Drive, Brookfield, CT 06804, (203)775-4595.

Dealer inquiries invited.

CmC

Connecticut microComputer, Inc.
34 Del Mar Drive, Brookfield, CT 06804
203 775-4595 TWX: 710 456-0052

TYPE-'N-TALK™

The exciting text-to-speech synthesizer with unlimited vocabulary.

Get your computer talking for only $375.00.

- Text-to-speech algorithm
- A one-watt audio amplifier
- SC-01 speech synthesizer chip (data rate: 70 to 100 bits per second)
- 750 character buffer
- Data switching capability
- Selectable data modes for versatile interfacing
- Baud rate (75-9600)
- Data echo of ASCII characters
- Phoneme access modes
- RS 232C interface
- Complete programming and installation instructions

The Votrax Type-'N-Talk™ is one of the easiest-to-program speech synthesizers on the market. It uses the least amount of memory and it gives you the most flexible vocabulary available anywhere.

Order now. Toll free.

Call the toll-free number below to order or request additional information. MasterCard or Visa accepted. Charge to your credit card or send a check for $375.00 plus $4.00 delivery. Add 4% sales tax in Michigan.

1-800-521-1350

For product information call: 1-313-588-0341

VOTRAX®
NOW YOU'RE TALKING.

500 Stephenson Highway, Troy, MI 48084

Type-'N-Talk™ is covered by a limited warranty. Write Votrax for a free copy.

routine checks to see if brackets had been input. Acting under the assumption that the user would never need to type an eight or nine in a chemical formula, these are promoted to the bracket corresponding to SHIFT-8 or SHIFT-9. Deletion is accomplished by shortening RESP$ in line 1160, after being careful to erase the previous character on the same line and below. (Remember, you could be erasing a subscript.) At this point the routine would check for and process any other desired input characters. However, if we wish to reject all other characters, we just go straight to line 1500.

Program 2 is actually a little routine to test out the subroutine (as is Program 1). It contains one other small subroutine of interest. If this were part of a program where the computer, as well as the user, were required to print out a formula, you would need a formatting routine. This one, appearing in lines 2000 to 2030, is quite simple. Send the formula to the subroutine as B$, and the formula is displayed with subscripts in their proper places. Again, just a minor adjustment is needed to turn this into a mathematical expression formatting subroutine. (See the example and discussion above on IIS's for PET: the logic is the same. Only the cursor control procedures are different.)

There is no doubt that Intelligent Input Subroutines are more difficult to devise and require more coding than simple INPUT statements. Their use, however, should lead to programs which are more user-oriented, a direction that we programmers should be heading as computer usage becomes more widespread.

Program 1: PET Version

```
1 REM  LISTING1
            :
2 REM IAS
         :
10 REM FOR INPUT, GOSUB 1000
20 :
30 REM *** MAIN PROGRAM ***
        :
40 GOSUB1000:IF RESP$="END" THEN E
   ND
50 PRINT"{DOWN}YOU JUST TYPED THIS
     ";RESP$
60 PRINT"{03 DOWN}TYPE SOMETHING E
   LSE OR ENTER ";CHR$(34)"EN
   D"CHR$(34)
70 GOTO40
80 :
90 REM ******************
997 :
998 REM INPUT SUBROUTINE  USES A$,A
   %.
999 REM RETURNS WITH RESP$
         :
1000 RESP$=""
1010 PRINT"${LEFT}";
1020 GETA$:IFA$=""THEN1020
1030 A%=ASC(A$):IFA%=13ORA%=141THENP
    RINT" ":RETURN
1035 REM RETURN PRESSED: ERASE CURSO
    R AND BACK TO MAIN PROGRAM

1040 IFA%<>20THEN1090:SKIP UNLESS DE
    LETE PRESSED
1050 IFLEN(RESP$)<1THEN1020:NOTHING
    TO DELETE
1060 RESP$=LEFT$(RESP$,LEN(RESP$)-1)
    :REMOVE LAST CHARACTER
1070 PRINT"{LEFT}  {02 LEFT}" ;:REM
    ERASE LAST CHARACTER AND C
    URSOR
1080 GOTO1010
1090 IF(A%>=161ANDA%=<169)THENA$=CHR
    $(A%-128):A%=ASC(A$):GOTO1
    110
1095 REM UNSHIFT UPPER ROW (UNLESS Y
    OU
1096 REM WANT TO ACCEPT THE GRAPHICS

1097 REM CHARACTERS
1100 IFA%<31ORA%>160THEN1020
1105 REM OF COURSE 1100 RESTICTS THE
       INPUT A BIT...IF YOU WAN
    T TO BE ABLE
1106 REM TO INPUT TOP ROW GRAPHICS
    SYMBOLS, REMOVE 1100
1110 IFA%=34ORA%=96THENPRINTCHR$(34)
    ;CHR$(34)"{LEFT}";:GOTO113
    0
1120 PRINTA$;
1130 RESP$=RESP$+A$:GOTO1010
```

Program 2: PET Version

```
10 POKE59468,14:REM ALLOW LOWER CA
    SE
20 FORI=1TO3:READQUEST$(I),A1$(I),
    A2$(I),A3$(I)
30 NEXTI:REM   READS IN QUESTION A
    ND TWO POSSIBLE UNFORMATED

40 REM    ANSWERS, AND A FORMATTED
      ANSWER FOR SCREEN DISPLAY

90 REM    ASK THREE QUESTIONS
100 FORI=1TO3
110 PRINT"{CLEAR}{03 DOWN}":PRINTQU
    EST$(I);"{03 DOWN}":PRINT"
          ";
120 GOSUB1000
```

OSI TRS-80 COLOR-80 OSI

GALAXIAN - 4K - One of the fastest and finest arcade games ever written for the OSI, this one features rows of hard-hitting evasive dogfighting aliens thirsty for your blood. For those who loved (and tired of) Alien Invaders. Specify system – A bargain at $9.95 OSI

LABYRINTH - 8K - This has a display background similar to MINOS as the action takes place in a realistic maze seen from ground level. This is, however, a real time monster hunt as you track down and shoot mobile monsters on foot. Checking out and testing this one was the most fun I've had in years! – $13.95. OSI

THE AARDVARK JOURNAL

FOR OSI USERS – This is a bi-monthly tutorial journal running only articles about OSI systems. Every issue contains programs customized for OSI, tutorials on how to use and modify the system, and reviews of OSI related products. In the last two years we have run articles like these!

1) A tutorial on Machine Code for BASIC programmers.

2) Complete listings of two word processors for BASIC IN ROM machines.

3) Moving the Directory off track 12.

4) Listings for 20 game programs for the OSI.

5) How to write high speed BASIC – and lots more –

Vol. 1 (1980) 6 back issues - $9.00

Vol. 2 (1981) 4 back issues and subscription for 2 additional issues - $9.00.

ADVENTURES!!!

For OSI, TRS-80, and COLOR-80. These Adventures are written in BASIC, are full featured, fast action, full plotted adventures that take 30-50 hours to play. (Adventures are interactive 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 8K on an OSI and 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 PLANT. 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.

DEATH SHIP (by Rodger Olsen)

Our first and original ADVENTURE, this one takes place aboard a cruise ship - but it ain't the Love Boat.

VAMPIRE CASTLE (by Mike Bassman)

This is a contest between you and old Drac - and it's getting a little dark outside. $14.95 each.

OSI NEW-NEW-NEW TINY COMPILER OSI

The easy way to speed in your programs. The tiny compiler lets you write and debug your program in Basic and then automatically compiles a Machine Code version that runs from 50-150 times faster. The tiny compiler generates relocatable, native, transportable machine code that can be run on any 6502 system.

It does have some limitations. It is memory hungry – 8K is the minimum sized system that can run the Compiler. It also handles only a limited subset of Basic – about 20 keywords including FOR, NEXT, IF THEN, GOSUB, GOTO, RETURN, END, STOP, USR(X), PEEK, POKE, -, =, *, /, , <,> , Variable names A-Z, and Integer Numbers from 0-64K.

TINY COMPILER is written in Basic. It can be modified and augmented by the user. It comes with a 20 page manual.

TINY COMPILER – $19.95 on tape or disk OSI

SUPERDISK II

This disk contains a new BEXEC* that boots up with a numbered directory and which allows creation, deletion and renaming of files without calling other programs. It also contains a slight modification to BASIC to allow 14 character file names.

The disk contains a disk manager that contains a disk packer, a hex/dec calculator and several other utilities.

It also has a full screen editor (in machine code on C2P/C4)) that makes corrections a snap. We'll also toss in renumbering and program search programs – and sell the whole thing for –

SUPERDISK II $29.95 (5¼") OSI

BARE BOARDS FOR OSI C1P

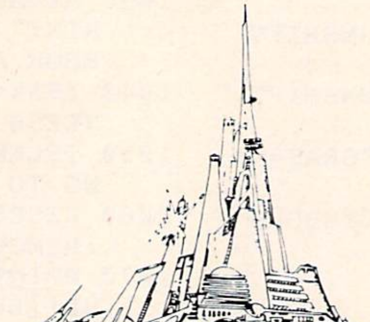
MEMORY BOARDS!!! – for the C1P – and they contain parallel ports!

Aardvarks new memory board supports 8K of 2114's and has provision for a PIA to give a parallel ports! It sells as a bare board for $29.95. When assembled, the board plugs into the expansion connector on the 600 board. Available now!

PROM BURNER FOR THE C1P – Burns single supply 2716's. Bare board – $24.95.

MOTHER BOARD – Expand your expansion connector from one to five connectors or use it to adapt our C1P boards to your C4/8P. - $14.95.

16K RAM BOARD FOR C1P – This one does not have a parallel port, but it does support 16K of 2114's. Bare Board $39.95.



WORD PROCESSING THE EASY WAY– WITH MAXI-PROS

This is a line-oriented word processor designed for the office that doesn't want to send every new girl out for training in how to type a letter.

It has automatic right and left margin justification and lets you vary the width and margins during printing. It has automatic pagination and automatic page numbering. It will print any text single, double or triple spaced and has text centering commands. It will make any number of multiple copies or chain files together to print an entire disk of data at one time.

MAXI-PROS has both global and line edit capability and the polled keyboard versions contain a corrected keyboard routine that make the OSI keyboard decode as a standard typewriter keyboard.

MAXI-PROS also has sophisticated file capabilities. It can access a file for names and addresses, stop for inputs, and print form letters. It has file merging capabilities so that it can store and combine paragraphs and pages in any order.

Best of all, it is in BASIC (0S65D 51/4" or 8" disk) so that it can be easily adapted to any printer or printing job and so that it can be sold for a measly price.

MAXI-PROS – $39.95. Specify 5¼ or 8" disk.

SUPPORT ROMS FOR BASIC IN ROM MACHINES – C1S/C2S. This ROM adds line edit functions, software selectable scroll windows, bell support, choice of OSI or standard keyboard routines, two callable screen clears, and software support for 32-64 characters per line video. Has one character command to switch model 2 C1P from 24 to 48 character line. When installed in C2 or C4 (C2S) requires installation of additional chip. C1P requires only a jumper change. – $39.95

C1E/C2E similar to above but with extended machine code monitor. – $59.95 OSI

ARCADE GAMES FOR OSI, COLOR-80 AND TRS-80 (8K OSI, 16K TRS-80 AND COLOR-80)

TIMETREK - A REAL TIME, REAL GRAPHICS STARTRECK. See your torpedoes hit and watch your instruments work in real time. No more unrealistic scrolling displays! $14.95.

STARFIGHTER - This one man space war game pits you against spacecruisers, battlewagons, and one man fighters, you have the view from your cockpit window, a real time working instrument panel, and your wits. Another real time goody. $9.95

BATTLEFLEET - This grown up version of Battleship is the toughest thinking game available on OSI or 80 computers. There is no luck involved as you seek out the computers hidden fleet. A topographical toughie. $9.95

QUEST - A NEW IDEA IN ADVENTURE GAMES! Different from all the others, Quest is played on a computer generated mape of Alesia. Your job is to gather men and supplies by combbat, 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

Please specify system on all orders

This is only a partial listing of what we have to offer. We offer over 120 games, ROMS, and data sheets for OSI systems and many games and utilities for COLOR-80 and TRS-80. Send $1.00 for our catalog.

OSI

AARDVARK TECHNICAL SERVICES, LTD.
2352 S. Commerce, Walled Lake, MI 48088
(313) 669-3110

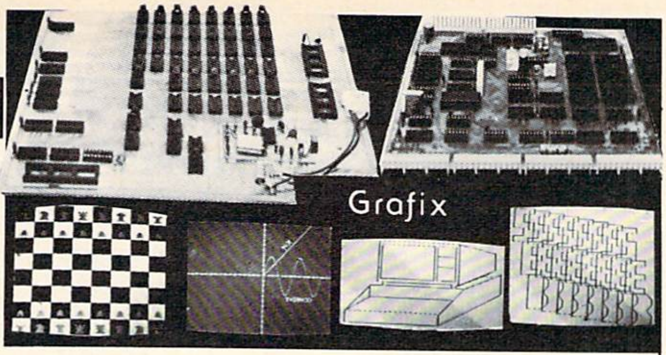
COLOR-80

```
130 IFRESP$=A1$(I)ORRESP$=A2$(I)THE
    N200
140 REM TAKE CARE OF INCORRECT RESP
    ONSE
150 PRINTTAB(20);"{03 DOWN}NO, THAT
    'S WRONG"
160 FORJ=1TO2000:NEXT
170 PRINT"{05 DOWN}          ANSWER
    IS ";A3$(I):GOTO210
190 REM   ACKNOWLEDGE CORRECT RESPO
    NSE
200 SC=SC+1:PRINTTAB(20);"{03 DOWN}
    GOOD"
210 FORJ=1TO4000:NEXTJ
220 NEXTI
290 REM  POSTMORTEM
300 PRINT"{CLEAR}{06 DOWN}YOUR SCOR
    E IS";SC
310 PRINT"{02 DOWN}":END
999 REM  INPUT ROUTINE
1000 RESP$=""
1010 PRINT"S{LEFT}";:REM  ANY CURSOR
     PROMPT YOU WANT
1020 GETA$:IFA$=""THEN1020
1030 LAST%=0:IFRESP$<>""THENLAST$=RI
    GHT$(RESP$,1):LAST%=ASC(LA
    ST$)
1040 IFASC(A$)=13 OR ASC(A$)=141THEN
    PRINT" ":RETURN
1045 REM 1040 -  IF RETURN PRESSED,
    ERASE PROMPT AND RETURN
1050 IFASC(A$)<>20THEN1100:SKIP UNLE
    SS DELETE PRESSED
1060 IFLEN(RESP$)<1THEN1010:NOTHING
    TO DELETE
1070 RESP$=LEFT$(RESP$,LEN(RESP$)-1)
    :REMOVE LAST CHARACTER
1080 PRINT"{LEFT}  {02 LEFT}{UP} {DO
    DOWN}{LEFT}";:GOTO1010
1090 REM PRINT CHARACTER
1100 IFA$="("THENA$="(":REM UNSHIFT
    BRACKET
1110 IFA$=")"THENA$=")":REM UNSHIFT
    BRACKET
1120 IFA$="("ORA$=")"ORA$="="ORA$="+
    "ORA$="-"THEN1180
1130 REM ABOVE ARE THE ONLY SYMBOLS
    ALLOWED
1140 B%=ASC(A$)
1150 IFB%<49ORB%>59THEN1170
1160 IF(LA%>64ANDLA%<91)OR(LA%>192AN
    DLA%<219)THENPRINT" {LEFT}
    {UP}";A$;"{DOWN}";:GOTO119
    0
1170 IF(B%<49OR(B%>=91ANDB%<=192)ORB
    %>219)THEN1010
1180 PRINTA$;
1190 :RESP$=RESP$+A$:GOTO1010
9997 :
9998 REM  DATA
9999 :
10000 DATA(X+1)(Y-1) = ,XY-X+Y-1,XY+Y
    -X-1,XY-X+Y-1
10010 DATA"FACTOR:  X{UP}2{DOWN} - Y{
    UP}2{DOWN} =",(X-Y)(X+Y),(
    X+Y)(X-Y),(X+Y)(X-Y)
10020 DATA"WHAT'S THE SQUARE OF A+B?"
    ,A2+B2+2AB,A2+2AB+B2,"A{UP
    UP}2{DOWN}+2AB+B{UP}2{DOWN
    DOWN}"
```

Program 3: PET Version

```
0 POKE59468,14:GOTO99
1 REM LISTING III
          :
2 I.I.S. FOR INPUT OF CHEMICAL FO
   RMULA  :
3 GOSUB 1000 TO RECEIVE A FORMULA
          :
98 ******  MAIN PROGRAM ******
          :
99 PRINT"{CLEAR}"
100 PRINT"ENTER CHEMICAL FORMULA."
110 PRINT:PRINT"TYPE "CHR$(34)"END"
    CHR$(34)" TO END"
120 PRINT:GOSUB1000
130 PRINT"{02 DOWN}":IFRESP$="END"T
    HENPOKE59468,12:END
140 GOTO100
150 :
160 ****************************
          :
1000 RESP$=""
1010 PRINT"S{LEFT}";: REM ANY CURSOR
     PROMPT YOU WANT
1020 GETA$:IFA$=""THEN1020
1030 A%=ASC(A$):IFA%=13ORA%=141THENP
    RINT" ";:RETURN : ERASE CU
    RSOR AND RETURN
1040 IFA%<>20THEN1090:SKIP TO 1090 U
    NLESS DELETE PRESSED
1050 IFLEN(RESP$)<1 THEN 1020: NOTHI
    NG TO DELETE
1060 RESP$=LEFT$(RESP$,LEN(RESP$)-1)
    :REMOVE LAST CHARACTER
1070 PRINT"{LEFT}  {02 LEFT}{DOWN} {
    UP}{LEFT}";:REM ERASE ON A
    ND BELOW LINE, CURSOR, THE
    N BACK UP
1080 GOTO1010
1090 IF(A$>="A"ANDA$<="Z")OR(A$>="A"
    AND A$<="Z")THEN 1170: IT
    'S A LETTER
1100 IFA$="("THENA$="(":GOTO1170
1120 IFA$="("THEN1170
1130 IFA$=")"THENA$=")":GOTO1170
```

SUPER OSI EXPANSION BOARDS



FOR OSI 1P, 2-4P, 2-8P, C4P, C8P

If your Challenger can't generate displays like those shown above **WHAT ARE YOU WAITING FOR?** The SEB-1 High Resolution Graphics and Memory Board (for CIP and Superboard II) and the SEB-2 High Resolution Graphics and Disk Controller Board (for C2/4/8) simply 'plug-in' to your computer and give you instant access to over 4900 individually addressable pixels in up to 8 colors! Your Hi-Res screen can go from 32×16 alphanumerics to 256×192 point graphics in 11 software selectable modes. The standard video of your computer is left intact, so that none of your current software library is outmoded. Use the graphics for Business, Scientific, Education, or Gaming displays that were impossible—until now!

Installation of either board requires absolutely NO modification of your computer—they just 'plug-in'. Nor do they preclude your using any other OSI-compatible hardware or software. In addition to the Hi-Res Graphics the SEB-1 gives C1 & Superboard II users 16K of additional user memory (over and above that memory devoted to the graphics), two 16 bit timers/counters, an on-board RF modulator, and a parallel port with handshaking. The SEB-2 gives OSI 48-pin BUS users an OSI hardwarew/software compatible Disk controller, and an RF modulator that can be user-populated.

	SEB-1	SEB-2
Assembled and Tested	$249.00 (5K RAM)	$239.00 (1K RAM)
Kit	$165.00 (No RAM)	$199.00 (No RAM)

	SEB-1	SEB-2
Bare Board & Manual	$ 59.00	$ 59.00
Manual only	$ 5.00	$ 5.00

SOFTWARE

TARG-IT! $11.95
An action-packed, super fast arcade game where you try to destroy Ram ships and the Evil Spectre inside the Crystal city. 8K.

PLANETARY LANDER $11.95
Your computer generates contour maps of the surface of a strange planet as you descend looking for the landing pad. As you make your final approach, the computer displays side and top views of the terrain and your ship as you try to land while avoiding the swarms of asteroids above the planet's surface. 8K.

PROGRAMMA-TANK .... $ 9.95
Programma-Tank is a complex simulation of a battle between two robot tanks as well as a fairly painless way to learn assembly language. The opponents program their tanks in 'Smalltank' a miniature programming language, which is much like assembly language. The two strategies are then pitted against one another until a victor emerges. 8K.

RUINS OF TALAMARINE $16.95
This is the first of our "Epic Quests" programs. Epic Quests could be described as graphic adventures, but they are much more. Can you retrieve the stolen Jewels from the ruined Talamarine castle? This series of two programs allows you to create a character with the qualities that you want to do battle with the forces of darkness. 8K.

CODENAME: BARRACUDA $19.95
Another of our "Epic Quests", in which you are an agent working for the US Government trying to rescue a defecting Soviet professor. The problem is that a middle-eastern government has kidnapped the professor and is forcing him to develop atomic weapons for them. The three programs (8K ea.) in this series allow you to create, train and send your character on his mission.

DUNGEONS OF ZORXON $19.95
Can you explore the Dungeons of Zorxon in this "Epic Quest"? Step into a nether world of monstrous beasts and fortunes in gold where magic reigns supreme. Three programs 8K each.

Write for **FREE** catalog
International Requests please supply 2 International Response Coupons

master charge — VISA

ORION SOFTWARE ASSO.
147 MAIN ST, P O BOX 310, OSSINING, NY 10562

COLOR CHART

- ✓ 8 COLORS
- ✓ ALPHANUMERICS
- ✓ HIGH RES. GRAPHICS

COLOR VIDEO BOARD FOR PET, AIM, SYM. . . $139.95

COLOR CHART is a new color video generator board that can be used to provide color CRT capability for the COMMODORE PET, ROCKWELL AIM and SYNERTEK SYM. The **COLOR CHART** board plugs into a 4K ROM socket and with two additional connections appears as 4K of RAM. **COLOR CHART** has several modes of operation from a 32 character X 16 line alphanumeric mode in eight colors to a high resolution graphic mode with 128 X 192 controllable pixels in two colors. **COLOR CHART** generates a composite video signal that can be used with a color monitor or a T.V. set and a RF modulator.

NEED HIGH SPEED RS232 SERIAL I/O FOR PET, AIM, SYM?

Portmaker is the answer! Two ACIA's and a socket for a ROM allow Portmaker to plug into a ROM socket and still allow use of the ROM. The serial ports appear as the top eight bytes of the ROM. Full bipolar RS232 buffering is provided. Baud rate can be jumped from 4800 to 300 baud.

Portmaker DUAL SERIAL PORT **$69.95**

NEW COMPACK $129.95

INTELLIGENT TERMINAL COMMUNICATIONS PACKAGE

Compack is the Intelligent Terminal package you have been waiting for. A new STANDARD TERMINAL PROGRAM from Eastern House Software combined with a special single PORTMAKER provides the highest performance available for the PET. The package, complete with RS232 cable, is available on Commodore 4040 or PEDISK diskette

EXCHANGE 8" DISKETTES WITH MINI's, MAINFRAMES - IBM 3740

PEDISK II

FLOPPY DISK FOR PET, AIM, SYM

PEDISK is the high performance IBM compatable floppy disk system. Available for PET, AIM, or SYM in both 5" and 8" versions. The 5" system provides an economical single disk that can be expanded. Buy a single disk now, add more drives later. The 8" system provides full IBM 3740 compatibility. A new program, EXCHANGE, even allows PEDISK to read and write ASCII files to IBM compatible computers.

SINGLE 5" DISK SYSTEM (540-1) $595.00

SINGLE 8" DISK SYSTEM (877-1) $995.00

PEDISK II SOFTWARE

- PAPERMATE - WORD PROCESSOR .... $ 50.00
- MAE - MACROASSEMBLER/EDITOR $169.95
- FLEXIFILE II - DATA BASE MANAGER .... $ 60.00
- fullFORTH+ - complete FORTH system .... $ 75.00
- IBM EXCHANGE - read/write ACSII files $125.00

Spacemaker II $39.95
Switch 1 of 4 ROM's on a single board. Slide switch electronically controlls selection, no glitches. Remote Control switch capability.

Romdriver $39.95
Remote Control switch output board.

ROM/IO .... $ 9.95
Software for ROMDRIVER.

cgrs MICROTECH
P.O. Box 102 • LANGHORNE, PA 19047
215-757-0284

```
1140 IFA$=")"THEN1170
1150 IFA$>"1"ANDA$<="9"THENPRINT" {L
    LEFT}{DOWN}";A$;"{UP}";:GO
    TO1180:NUMBER SUBSCRIPTED
1160 GOTO1020:REJECT ANYTHING ELSE
1170 PRINTA$;:REM PRINT LETTER
1180 RESP$=RESP$+A$:GOTO1010
```

Program 4: PET Version

```
0 POKE59468,14:PRINT"{CLEAR}":GOT
    O100
10 LISTING IV  I.I.S. CHEMICAL NAM
    ES     :
20 GOSUB 1000. USES A$,A%.
30 RETURNS WITH RESP$.
40 ACCEPTS LETTERS, BRACKETS, SPAC
    E.
50 UNSHIFTS BRACKETS.
60 CAPITALIZES ROMAN NUMBERALS
89 :
90 ****** MAIN PROGRAM *****
91 :
100 PRINT"TYPE A CHEMICAL NAME"
110 PRINT"{02 DOWN}";:GOSUB1000
115 PRINT:PRINT"{DOWN}RESP$ = ";CHR
    $(34)RESP$CHR$(34)
120 IFRESP$="END"THENPOKE59468,12:E
    ND
130 PRINT:PRINT"{03 DOWN}TRY ANOTHE
    R.  TYPE "CHR$(34)"END"CHR
    $(34)" TO END"
140 GOTO110
150 :
160 **************************       ~
    :
1000 BFLAG%=0:RESP$=""
1005 PRINT"${LEFT}";:REM ANY CURSOR ~
    PROMPT YOU LIKE
1010 GETA$:IFA$=""THEN1010
1020 A%=ASC(A$):IFA%=13 OR A%=141 TH
    EN PRINT" ";:RETURN
1030 IFA%<>20THEN1090:SKIP UNLESS DE
    LETE PRESSED
1040 IF LEN(RESP$)< 1 THEN 1020:NOTH
    ING TO DELETE
1050 IFRIGHT$(RESP$,1)="("THENBFLAG%
    =0:REMCLEAR BRACKET FLAG I
    N ERASING BRACKET
1060 RESP$=LEFT$(RESP$,LEN(RESP$)-1)
    :REMOVE LAST CHARACTER
1070 PRINT"{LEFT}$ {02 LEFT}";:REM E
    RASE OLD CURSOR & REPLACE ~
    LAST CHARACTER WITH CURSOR

1080 GOTO1010
1090 IF(A%>64ANDA%<91)ORA%=32ORA%=16
    0ORA%=201ORA%=214THEN1150:
    SKIP IF LETTER
1095 REM ALLOWS LOWER CASE LETTERS O
    NLY, EXCEPT ROMAN NUMBERAL
    S, AND SPACE
1100 IFA$<>"("ANDA$<>")"ANDA$<>"("AN
    DA$<>")"THEN1010:REJECT AL
    L BUT BRACKETS
1110 IFA$="("THENA$="("
1120 IFA$=")"THENA$=")"
1130 IFA$="("THENBFLAG%=1:GOTO1180
1140 IFA$=")"THENBFLAG%=0:GOTO1180
1150 IFBFLAG%=0THEN1180
1160 IFA$="I"THENA$="I":GOTO1180
1170 IFA$="V"THENA$="V"
1180 PRINTA$;:RESP$=RESP$+A$:GOTO100
    5
```

Program 1: Apple Version

```
1  HOME : PRINT "TYPE ANY STRING
      AND HIT RETURN": PRINT ("TOU
      CH ESC TO EXIT"): VTAB 6: GOTO
      1000
97 :
98  REM  THE BASIC SUBROUTINE
99 :
100 RESP$ = "":R1$ = "":R1 = 0
110  GET A$
117 :
118  REM  ESCAPE
119 :
120  IF A$ =  CHR$ (27) THEN  RETURN

127 :
128 : REM  DELETE
129 :
130 IF A$ =  CHR$ (8) THEN 200
137 :
138  REM  RETURN
139 :
140  IF A$ =  CHR$ (13) THEN 220
147 :
148  REM  RIGHT ARROW COPY
149 :
150  IF A$ =  CHR$ (21) THEN 250
152 :
153  REM  REJECT CONTROL CHARACTE
     RS
154 :
155  IF  ASC (A$) <  27 THEN 110
157 :
158  REM  ALLOW ANYTHING ELSE
159 :
160  IF R1 = 0 THEN R1$ = A$: GOTO
     190
170  IF  LEN (R1$) <  = R1 + 1 THEN
```

HUNTINGTON COMPUTING

Now Selling Atari®, PET®, TRS-80® Software

Atari®

Compu-Math Decimals (disk)	$39.95 now	$35.14
Comp Magic Kayos (disk)	$34.95 now	$30.74
Ali Baba & 40 Thieves	$32.95 now	$28.94
Arcade - Ghost Hunter (cass.)	$29.95 now	$26.34
Arcade - Ghost Hunter (disk)	$34.95 now	$30.74
Comp Magic Kayos (cass.)	$34.95 now	$30.74
Epyx Crush Crumble Chomp (cassette or disk)	$29.95 now	$26.34
Creative ATC (cass.)	$11.95 now	$10.44
Mouskattack	$39.95 now	$35.14
The Next Step	$39.95 now	$35.14
Poker Solitaire (cass.)	$14.95 now	$12.74
Gomoku (cass.)	$19.95 now	$16.94
Reversi (cass.)	$19.95 now	$16.94
Cypher Bowl (cass.)	$29.95 now	$25.44
Rescue at Rigel (cass.)	$29.95 now	$25.44
Star Warrior (cass.)	$39.95 now	$33.94
Invasion Orion (cass.)	$24.95 now	$21.24
Datestones of Ryn (cass.)	$19.95 now	$16.94
Conflict 2500 (cass.)	$15.00 now	$12.74
Empire of the Overmind (disk)	$35.00 now	$29.64
Tanktics (cass.)	$24.00 now	$20.44
Atari* Mailing List (disk)	$24.95 now	$21.24
Atari* Character Generator (disk)	$19.95 now	$16.94
Text Wizard (disk)	$99.95 now	$84.44
Atari* Character Gen. (cass.)	$15.95 now	$13.54
Le Stick	$39.95 now	$33.94
Checker King (cass.)	$19.95 now	$16.94
MicroChess (cass.)	$19.95 now	$16.94
Survival/Adventure (disk)	$24.95 now	$21.24
3-D Supergraphics (disk)	$39.95 now	$33.94
3-D Supergraphics (cass.)	$39.95 now	$33.94
Mind-Bogglers I (disk)	$19.95 now	$16.94
Mind-Bogglers I (cass.)	$15.95 now	$13.54
VersaWriter Graphics Tablet	$300.00 now	$254.44
Hidden Words	$17.50 now	$15.44
Spatial Relations	$17.50 now	$15.44
Word-Scramble	$15.00 now	$13.54
Preschool Fun	$15.00 now	$13.54
Fastgammon (cass.)	$19.95 now	$16.94
Assembler (cass.)	$24.95 now	$21.14
6502 Disassembler (cass.)	$11.95 now	$10.14
6502 Disassembler (disk)	$14.95 now	$12.64
Tank Trap (cass.)	$11.95 now	$10.14
Tank Trap (disk)	$14.95 now	$12.64
Tari Trek (cass.)	$11.95 now	$10.14
QS Forth (disk)	$79.95 now	$67.94
Starbase Hyperion (disk)	$22.95 now	$19.44
Name That Song (cass.)	$14.95 now	$12.64
Jaw Breaker (disk)	$29.95 now	$25.44
Pornopoly (disk)	$29.95 now	$25.44
The Broker (CCI) (disk)	$99.95 now	$84.94
Super Modem Pak (CCI) (disk)	$49.95 now	$42.44
Atari* Snuff (CCI) (disk)	$29.95 now	$25.44
Utility Man (CCI) (disk)	$99.95 now	$84.94
Tanktics (cass)	$24.00 now	$20.44
Fantasyland (disk)	$59.95 now	$50.94
Empire of the Overmind (cass.)	$30.00 now	$25.44
Bridge 2.0 (cass)	$17.95 now	$15.24
Nominoes Jigsaw (cass)	$17.95 now	$15.74
Intruder Alert (cass)	$16.95 now	$14.84
Alpha Fighter (disk)	$18.95 now	$16.64
Compu-read (disk)	$29.95 now	$25.44
Letter Perfect (disk)	$150.00 now	$127.44
Sammy Sea Serpent (cass.) PDI	$16.95 now	$14.34
Cribbage (Thesis) (cass.)	$15.00 now	$13.54
Visicalc	$200.00 now	$149.00
Kross 'N Quotes PDI (cass.)	$16.95 now	$14.34
Star Raiders (cart.)	$39.95 now	$33.94
Stock Charting	$24.95 now	$21.14
Adventure International		10% - 15% off sug. retail
Atari* Safari (CDS) (disk)		$33.94
Atari* Safari (CDS) (cass.)		$25.44
Computation (Thesis) (cass.)	$15.00 now	$13.94

Order by Phone

800-344-5109

800-692-4146

In California

Pet®

AVALON HILL GAME COMPANY		
B-1 Nuclear Bomber (cass.)	$15.00 now	$12.77
Midway Campaign (cass.)	$15.00 now	$12.77
No. Atlantic Convoy Raider (cass.)	$15 now	$12.77
Nukewar (cass.)	$15.00 now	$12.77
Conflict 2500 (cass.)	$15.00 now	$12.77
Planet Miners (cass.)	$15.00 now	$12.77
Computer Acquire (cass.)	$20.00 now	$16.97
Lords of Karma (cass.)	$20.00 now	$16.97
VIC SOFTWARE		
UMI Kiddie Checkers	$7.95 now	$6.87
UMI Star Wars	$16.95 now	$14.87
UMI AMOK	$18.95 now	$16.67
UMI Globber	$29.95 now	$21.87
ARTWORX		
Teacher's Pet (disk)	$18.95 now	$16.67
Teacher's Pet (cass.)	$14.95 now	$13.07
Vaults of Zurich (disk)	$25.95 now	$22.77
AUTOMATED SIMULATIONS		
Introductory 3-Pack (disk) (Rescue, Moroc's, and Datestones)	$49.95 now	$39.97
Rescue at Rigel (cass.)	$29.95 now	$25.47
Temple of Apshai (cass.)	$39.95 now	$33.97
Hellfire Warrior (cass.)	$39.95 now	$33.97
Starfleet Orion (cass.)	$24.95 now	$21.27
Invasion Orion (cass.)	$24.95 now	$21.27
Moroc's Tower (cass.)	$19.95 now	$16.97
Datestones of Ryn (cass.)	$19.95 now	$16.97
PERSONAL SOFTWARE		
Checker King (cass.)	$19.95 now	$16.97
Gammon Gambler (cass.)	$19.95 now	$16.97
MicroChess (cass.)	$19.95 now	$16.97
Bridge Partner (cass.)	$19.95 now	$16.97
Time Trek (cass.)	$19.95 now	$16.97
UNITED SOFTWARE OF AMERICA		
KRAM (disk)	$99.95 now	$84.97
Super KRAM (disk)	$175.00 now	$153.97
Request (disk)	$225.00 now	$191.27
Thinker (disk)	$495.00 now	$420.77
Space Intruders (cass.)	$19.95 now	$16.97
All MICRO-ED		10% Off List
All Microcomputer Workshops		15% Off List
VIC SOFTWARE		
Addcom Missile Commander		$16.66
Channel Vic Data Logger		13.16
MMA Star Command		$8.96
TIS Basic Programming I		$17.46

Apple®

See full page of Apple* products elsewhere in this magazine.

Gorgon	$39.95 now	$33.99
Word Star	$375.00 now	$289.00
Mail Merge	$125.00 now	$106.19
Super Sort	$200.00 now	$169.99
VisiCalc 3.3	$200.00 now	$149.00
Wurst of Huntington Computing		$19.99
Nibble Express	$12.95 now	$11.99
Soft Porn Adventure	$29.95 now	$25.39
Time Lord	$29.95 now	$25.39
French Hangman	$29.95 now	$25.39
Alicia-Sp. bilingual reader	$29.95 now	$25.39
H&H Stock Trader	$190.00 now	$161.49
Grow (CIA)	$35.00 now	$31.49
All Edu-Ware		10% to 15% Off List
VersaCalc	$100.00 now	$84.99
Hebrew	$60.00 now	$50.99
All Serendipity		10% to 15% Off List
All Sirius		15% Off List
All Sybex Courses		10% to 15% Off List
Win at the Races	$39.95 now	$33.99
Disk Prep	$25.00 now	$21.19
PLE Chip	$60.00 now	$50.99

We maintain a huge inventory of software for Apple* and hardware. Call us toll free for the latest programs. We also stock a large supply of computer books. Visit us in person at our new 3300-square foot store at 1945 South Dairy in Corcoran, Calif.

Great Grandma Huntington

Great Grandma Huntington always said to try harder – and we do. We will soon have tee shirts for sale with Great Grandma Huntington's picture on them. Watch for Granny!

TRS-80®

BIG FIVE SOFTWARE		
Super Nova (cass.)	$15.95 now	$13.58
Galaxy Invasion (cass.)	$15.95 now	$13.58
Attack Force (cass.)	$15.95 now	$13.58
Cosmic Fighter (cass.)	$15.95 now	$13.58
Meteor Mission II (cass.)	$15.95 now	$13.58
ADVENTURE INTERNATIONAL		
Adv. 4-5-6 (disk)	$39.95 now	$35.08
Adv. 1-2-3 (disk)	$39.95 now	$35.08
Adv. 7-8-9 (disk)	$39.95 now	$35.08
ARTWORX		
Teacher's Pet (disk)	18.95 now	$16.58
Nominoes Jigsaw (disk) color	$21.95 now	$19.28
Nominoes Jigsaw (cass.) color	$17.95 now	$15.78
Bridge 2.0 (cass.)	$17.95 now	$15.78
Hearts (disk)	$19.95 now	$17.48
BRODERBUND SOFTWARE		
Galactic Trilogy (disk)	$39.95 now	$33.98
Galactic Empire (cass.)	$14.95 now	$12.68
Galactic Trader (cass.)	$14.95 now	$12.68
Galactic Revolution (cass.)	$14.95 now	$12.68
Tawala's Last Redoubt (cass.)	$19.95 now	$16.98
DATASOFT		
Iago (disk)	$24.95 now	$21.18
Football Classics (disk)	$24.95 now	$21.18
Arcade-80 (disk)	$24.95 now	$21.18
Iago (cass.)	$19.95 now	$16.98
Football Classics (cass.)	$19.95 now	$16.98
Arcade-80 (cass.)	$19.95 now	$16.98
Sigmon (COLOR) (cass.)	$29.95 now	$25.38
SECS (COLOR) (cass.)	$29.95 now	$25.38
ACORN SOFTWARE		
Invaders From Space (disk)	$20.95 now	$17.78
Duel-N-Droids (disk)	$20.95 now	$17.78
Pinball (disk)	$20.95 now	$17.78
Pigskin (disk)	$20.95 now	$17.78
Quad (disk)	$20.95 now	$17.78
Basketball (disk)	$20.95 now	$17.78
Gammon Challenger (disk)	$20.95 now	$17.78
Everest Explorer (disk)	$20.95 now	$17.78
SuperScript (disk)	$29.95 now	$25.38
System Savers (cass.)	$14.95 now	$12.68
Invaders From Space (cass.)	$14.95 now	$12.68
Duel-N-Droids (cass.)	$14.95 now	$12.68
Pinball (cass.)	$14.95 now	$12.68
Pigskin (cass.)	$14.95 now	$12.68
Quad (cass.)	$14.95 now	$12.68
Basketball (cass.)	$14.95 now	$12.68
Gammon Challenger (cass.)	$14.95 now	$12.68
Everest Explorer (cass.)	$14.95 now	$12.68
All Automated Simulations		10% to 15% Off List
All Avalon Hill		10% to 15% Off List
All Hayden		10% to 15% Off List
All Microsoft		10% to 15% Off List

FOREIGN ORDERS

We regret that we can no longer accept checks (bank charges were sometimes greater than the amount of the check). We will gladly accept U.S. currency, VISA, MasterCard or American Express at no extra charge, or you may make direct wire transfers to our bank, Security Pacific, Corcoran, CA 93212, for a $6.00 charge. All overseas orders are shipped by air.

VISCALC
Special for Pet®, Atari® & Apple®
Regular $200.00 List
Now $149.00

HUNTINGTON COMPUTING
Post Office Box 1235
Corcoran, California 93212
Order by Phone 800-344-5109
In California 800-692-4146

(209) 992-5411

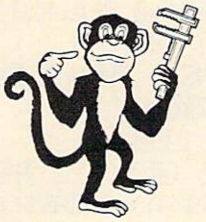
We take MasterCard, American Express or VISA (Include card # and expiration date). California residents add 6% tax. Include $2.00 for postage. Foreign and hardware extra. Send for free catalog. Prices subject to change.

```
     R1$ =  LEFT$ (R1$,R1) + A$: GOTO
     190
180 R1$ =  LEFT$ (R1$,R1) + A$ +
      RIGHT$ (R1$, LEN (R1$) - R1
      - 1)
190 R1 = R1 + 1: PRINT A$;: GOTO
     110
197 :
198  REM  PERFORM DELETION
199 :
200  IF R1 >  = 1 THEN R1 = R1 -
     1: PRINT A$;: GOTO 110
210  GOTO 110
217 :
218  REM  HANDLE RETURN
219 :
220  IF R1 = 0 THEN 110
230 RESP$ =  LEFT$ (R1$,R1): FOR
     A = 1 TO  LEN (R1$) - R1: PRINT
     " ";: NEXT
240  PRINT : RETURN
247 :
248  REM  RIGHT ARROW COPY
249 :
250  IF R1 = 0 OR  LEN (R1$) <  =
     R1 THEN 110
260 A$ =  MID$ (R1$,R1 + 1,1)
270  IF  LEN (R1$) = R1 + 1 THEN
     R1$ =  LEFT$ (R1$,R1) + A$: GOTO
     190
280 R1$ =  LEFT$ (R1$,R1) + A$ +
      RIGHT$ (R1$, LEN (R1$) - R1
      - 1): GOTO 190
997 :
998 : REM  MAIN PROGRAM
999 :
1000  GOSUB 100
1010  IF A$ =  CHR$ (27) THEN  PRINT
     "ESCAPED": END
1020  PRINT : PRINT "YOU TYPED ";
     RESP$: PRINT : PRINT : GOTO
     1000
```

Program 2: Apple Version

```
1  GOTO 10000
997 :
998  INPUT SUBROUTINE
999 :
1000 RESP$ = ""
1010  POKE  - 16368,0
1020  GET A$:A =  ASC (A$): IF A <
     > 13 THEN 1050
1030  IF RESP$ = "" THEN 1020
1040  RETURN
1047 :
1048  REM  TAKE CARE OF LETTERS
1049 :
1050  IF A < 65 OR A > 90 THEN 10
     70
1060 RESP$ = RESP$ + A$: PRINT A$
     ;: GOTO 1020
1067 :
1068 : REM  TAKE CARE OF NUMBERS
1069 :
1070  IF A < 49 OR A > 55 THEN 11
     00
1080 CV% =  PEEK (37) + 1: VTAB (
     CV% + 1)
1090 RESP$ = RESP$ + A$: PRINT A$
     ;: VTAB (CV%): GOTO 1020
1097 :
1098  REM  BRACKETS SHIFTED?
1099 :
1100  IF A$ = "8" OR A$ = "(" THEN
     RESP$ = RESP$ + "(": PRINT "
     (";: GOTO 1020
1110  IF A$ = "9" OR A$ = ")" THEN
     RESP$ = RESP$ + ")": PRINT "
     )";: GOTO 1020
1117 :
1118  REM  DEAL WITH DELETE
1119 :
1120  IF A <  > 8 THEN 1500
1130  IF RESP$ = "" THEN 1020: REM
       NOTHING TO DELETE
1140 H% =  PEEK (36): HTAB (H%): PRINT
     " ";: HTAB (H%):CV% =  PEEK
     (37) + 1: VTAB (CV% + 1): PRINT
     " ";: VTAB (CV%): HTAB (H%)
1150  IF  LEN (RESP$) = 1 THEN RE
     SP$ = "": GOTO 1020
1160 RESP$ =  LEFT$ (RESP$, LEN (
     RESP$) - 1): GOTO 1020
1175 :
1180  REM  PUT ANY OTHER SPECIAL
      CHECKS FOR SPECIFIC INPUT
      HERE (EXAMPLE: HANDLE ESCAPE
      KEY (CHR$(27)) IF YOU WISH
      IT TO HAVE A SPECIAL
      FUNCTION).
1185 :
1500  GOTO 1020: REM  REJECT
        ANYTHING ELSE
1510 :
1997 :
1998  REM  PRINT OUT FORMULA
1999 :
2000  FOR II = 1 TO  LEN (B$):A$ =
     MID$ (B$,II,1)
2010  IF  ASC (A$) > 64 OR  ASC (
```

7 ATARI PRODUCTS



THE MONKEY WRENCH

The Monkey Wrench is a machine language ROM cartridge which extends the operating capability of the ATARI 800 computer. The Monkey Wrench provides 9 new BASIC commands. They are:

Auto Line Numbering	– Provides new line numbers when entering BASIC program lines.
Delete Line Numbers	– Removes a range of BASIC line numbers.
Renumber	– Renumbers BASIC's line numbers including internal references.
Cursor Exchange	– Allows usage of the cursor keys without holding down the CTRL key.
Change Margins	– Provides the capability to easily change the screen margins.
Memory Test	– Provides the capability to test RAM memory.
Hex Conversion	– Converts a hexidecimal number to a decimal number.
Decimal Conversion	– Converts a decimal number to a hexidecimal number.
Monitor	– Enter the machine language monitor. In addition to the BASIC commands, the Monkey Wrench also contains a machine language monitor with 15 commands used to interact with the powerful features of the 6502 microprocessor

Cartridge and Manual – $49.95

TYPING EXERCISE FOR ATARI

Typing Exercise is a great educational program for those who wish to improve their typing skills. Typing Exercise consits of two programs. TYPING 1 contains 13 typing drills; 9 drills progress thru alphabet and 4 thru numerics. TYPING 2 is a timed typing test. Time and words per minute are calculated for you. . . . . **810 Diskette – $12.95**

EPROM CARTRIDGE

The EPROM cartridge is a specially designed printed circuit board which will allow the user to install his or her own EPROM software. Uses 2716, 2532, 2732, type EPROMs

Cartridge and Manual – $19.95

OTHER NICE STUFF

Atari 400 16K Memory	$345.00
Atari 800 16K Memory	$779.00
Commodore UIC-20 Computer	$265.00
Syncom or Memorex 5¼" disks	10 for $30.00
Mini-Flex disk File Case (holds 50 5¼" disks)	$24.95
Ribbon Cartridge For Starwriter, Diablo, etc.	Mylar –$5.00, Cloth – $6.00
Starwriter 25 cps printer with tractors Parallel Interface	$1645.00

MAE (Macro Assembler Editor)

MAE contains the most powerful 6502 assembler and test editor currently on the market. If you are looking for a professional development tool that can greatly increase the productivity of your programming staff, then MAE may be the answer. The following are just some of MAE's features – Write for detailed spec sheet:

- MAE was written entirely in machine language – not in Basic like some assemblers we know of. Thus you get very fast and accurate assemblies.
- Contains a machine language monitor with numerous commands for debugging machine code.
- 38 error codes, 27 commands, 26 pseudo ops, and 5 conditional assembly operators.
- Contains a word processor, example files, and learning aid.
- Requires at least 32K of memory.
- All commands oriented for disk operation with ATARI 810 disk drive.
- Macro, Conditional Assembly, and Interactive Asembly capability.
- Sorted Symbol Table.
- Optionally creates executable object code in memory or relocatable object code on disk.
- 50 page manual.

810 Diskette and Manual – $169.95
(requires license agreement)

MACRO ASSEMBLER AND TEXT EDITOR (ASSM/TED)

ASSM/TED is a high powered Macro assembler and text editor for use with ATARI 800 computers with at least 40K of memory.

- Written entirely in Machine Language – Not in Basic like some we know of. Thus you get very fast and accurate assemblies.
- 36 Error Codes, 26 Commands, 22 Pseudo ops.
- Macro and Conditional Assembly Capability.
- Input/Output of source files to cassette deck.
- Multiple source files on cassette may be assembled.
- Built-in machine language monitor. . . . . **Cassette and Manual – $49.95**

810 Diskette and Manual – $53.95

MACHINE LANGUAGE MONITOR FOR ATARI

The Machine Language Monitor for ATARI provides 21 commands which allows the user the ability to interact with the 6502 microprocessor. It is compatible with ATARI BASIC and (once loaded) is ready for your use at anytime. The monitor comes on cassette or on diskette for the ATARI 810 disk.

Cassette version – $24.95 Diskette version – $29.95

MEMORY TEST FOR ATARI

When you purchase a new ATARI or add on new RAM modules, you need to be sure that the memory is working properly. Remember, you only have a short guarantee on your memory. Memory Test performs the most extensive memory check available. . . . . **Cassette and Manual – $6.95**

EASTERN HOUSE SOFTWARE
3239 Linda Drive
Winston-Salem, N.C. 27106 U.S.A.

Call Orders: (919) 924-2889

VISA

We Pay Postage on Prepaid USA/Canada Orders. Add $7.50 to Cover Shipment Elsewhere

MasterCard

NEW PRODUCT

SWARM-100

For 24 Pin ROM Machines Only.

JUST PLUG IT IN

- No soldering
- No messy wires

SOFTWARE SELECTABLE

1. Software select one of two operating systems. (BASIC 2.0 / BASIC 4.0)
2. Software select utility ROMs at conflicting addresses.

$150.00

Master Charge and VISA accepted.

BATTERIES INCLUDED

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

```
     A$) < 49 THEN  PRINT A$;: NEXT
     : RETURN
2020 CV% =  PEEK (37) + 1: VTAB (
     CV% + 1): PRINT A$;: VTAB (C
     V%)
2030  NEXT : RETURN
2040 :
9997 :
9998  REM  *** MAIN PROGRAM ***
9999 :
10000  HOME : PRINT "TYPE IN CHEM
     ICAL FORMULAS": PRINT : PRINT
     "(TYPE " CHR$ (34)"END" CHR$
     (34)" TO END.)": POKE 34,4
10010  VTAB 15: GOSUB 1000: PRINT

10020  IF RESP$ = "END" THEN  POKE
     34,0: END
10030  PRINT : PRINT "YOU TYPED I
     N ";:B$ = RESP$: GOSUB 2000:
      PRINT
10040  FOR I = 1 TO 5: CALL  - 91
     2: NEXT : GOTO 10010
```

Program 1: Atari Version

Here's an "input anything" subroutine for Atari users. It will accept upper or lowercase, punctuation, and numbers. It will not accept any editing or cursor controls except BACK S (backspace). Line 1080 zeroes out the system's inverse video flag to cancel reverse field, and line 1120 traps bad characters. Programmer's note: Line 1060 shows one way to safely DIMension an array from within a subroutine without worrying about an error message. If INIT is zero, the array should be dimensioned, and INIT is set to one, preventing the array from being redimensioned unless INIT is reset with CLR. This prevents you from having to use a TRAP statement here.

```
10 PRINT "TYPE IN A LINE AND HIT RETURN:
":GOSUB 1000
20 PRINT "YOU TYPED:":? I$
30 END
1000 REM "Input (Almost) Anything"
1010 REM Subroutine for Atari
1020 REM Traps CTRL characters
1030 REM reverse field, all cursor
1040 REM controls except BACK S.
1050 REM Length of string limited by LN
1060 IF INIT=0 THEN DIM I$(100):LN=100:O
PEN #1,4,0,"K:":INIT=1
1070 LNS=1
1080 GET #1,A:POKE 694,0:REM Kill RVS
1090 IF A=155 THEN ? :RETURN
1100 IF A=126 AND LNS>1 THEN LNS=LNS-1:I
$(LNS)=" ":PRINT CHR$(A);
1110 IF LNS>LN THEN 1080
1120 IF A<32 OR A=96 OR A>122 THEN 1080
1130 I$(LNS,LNS)=CHR$(A):? CHR$(A);:LNS=
LNS+1
1140 GOTO 1080
```

Program 2: Atari Version

This program will let the user enter a chemical formula as described in the article, with numbers subscripted after capital letters. This routine only accepts uppercase letters, and changes a typed "9" or "0" to "(" and ")", respectively. There is an "intelligent printing routine" in the main program. Note that BACK S deletes the character behind and under the cursor, requiring two lines of space on the screen.

```
10 PRINT "ENTER CHEMICAL FORMULA:"
20 GOSUB 1000:PRINT
30 PRINT "I$=";CHR$(34);I$;CHR$(34):PRIN
T
40 PRINT "YOU TYPED:"
45 REM Intelligent Print Routine
50 FOR I=1 TO LEN(I$)
60 A=ASC(I$(I,I)):T=ASC(I$(I-(I>1)))
70 IF A>48 AND A<57 AND (T>64 AND T<91 O
R T=41) THEN ? "{DOWN}";CHR$(A);"{UP}";:
GOTO 90
80 PRINT CHR$(A);
90 NEXT I
100 PRINT :PRINT
110 END
1000 REM Modified Input Subroutine
1010 REM for chemical formulas.
1020 REM Shifts subscripts down,
1030 REM changes 9 and 0 into
1040 REM left and right brackets.
1050 REM
1060 IF INIT=0 THEN DIM I$(100):LN=100:O
PEN #1,4,0,"K:":INIT=1
1070 LNS=1:I$=" "
1080 GET #1,A:POKE 694,0:REM Kill RVS
1090 IF A=155 THEN ? :RETURN
1100 IF A=126 AND LNS>1 THEN LNS=LNS-1:I
$(LNS)=" ":PRINT "{LEFT} {DOWN}{LEFT} {L
EFT}{UP}";
```

```
1110 IF LNS>LN THEN 1080
1120 IF A<32 OR A>90 THEN 1080
1130 T=LNS-(LNS>1):T=ASC(I$(T,T))
1135 IF (T>64 AND T<91 OR T=41) AND A>48
 AND A<57 THEN ? "{DOWN}";CHR$(A);"{UP}"
;:GOTO 1170
1140 IF A=57 THEN A=40
1150 IF A=48 THEN A=41
1160 ? CHR$(A);
1170 I$(LNS,LNS)=CHR$(A):LNS=LNS+1
1180 GOTO 1080
```
©

Machine Language: Jump To It!

Jim Butterfield
Toronto, Canada

The 6502 Jump (JMP) has advantages over the various Branch instructions. For one thing, it may go anywhere in memory, where the Branches can hop only a hundred and twenty locations or so either way. Sometimes it's useful that Jump is unconditional: you want to go somewhere and your condition flags are not definitely known.

Jump has its limitations, too. It takes up one more byte than a Branch. You cannot use it conditionally. And a Jump is not "truly relocatable." If you had a program with no Jumps, only Branches, and you decided to move it to a new location in memory, the Branches would still work. After all, Branches use relative addressing which says to hop ahead or back a number of bytes; so when the program is moved, the Branches are still valid in their new place. Not so with the Jump: when your program moves, all Jumps to within that program must be changed.

Often, Jump and Branches can be used together to give advantages of each: the conditional test of the Branch, and the unlimited reach of the Jump. It's not uncommon to see coding like:

```
        BEQ NEXT
        JMP NOTEQ
NEXT    ....
```

The effect is a little like "inverting" the Branch instruction. The code continues if the equals condition is found (BEQ goes to NEXT and continues); but jumps away if not-equals is the case. It seems backwards: the Equals text causes us to change direction if we find not-Equals; but it works well without fuss or bother.

Jump Tables

Jump Tables are a collection of Jump instructions, one behind the other, arranged in some part of memory. They tend to baffle the beginning expert.

"Beginning experts," by the way, are programmers who have learned all the instructions, but haven't yet absorbed the wisdom that comes with programming experience. When confronted with a Jump table, the reasoning goes as follows: "I am asked to go to this location, which will immediately Jump somewhere else. What's the point? I can go directly to the destination and save myself a few microseconds and perhaps a few bytes of memory."

The concepts of microsecond and bytes can be quite valid in certain environments. But most of us should be placing the emphasis elsewhere. Machine language is plenty fast for most applications; you won't have time for an extra cup of coffee in those five hundred microseconds of time you save. Memory is cheap and plentiful; you're unlikely to run out of space in the average machine language program you code. A sound program will probably be economical of space and time, but the main objective is to have a well-constructed program.

What's the purpose of a Jump table? To give the program a standard set of locations through which it can access other programs. This can be very useful in providing easy testing and easy compatibility between different machines.

Suppose, for example, you were writing a program that did quite a bit of screen output. If you wrote for a PET/CBM and then wanted to transfer it to an Apple, AIM or Atari, you might have to change every call to the output routine; there might be dozens of these. On the other hand, if you arranged your code so that every output call went to your own location PRJUMP which in turn jumped to the PET/CBM output routine, you'd need to change only the address of your PRJUMP instruction to switch output for the other machine.

Additionally, building a Jump table encourages you to identify in advance those program connections that might need changing. When it's complete, you'll have an easily identifiable list of those connection points, laid out neatly in your program.

Commodore has carefully laid out a Jump table near the top of ROM memory for the use of Machine Language programmers. Every model of PET, CBM and VIC uses the same entry points, so that the user with experience on one model can easily move to the other. This also makes for a great deal of compatibility between machines; many programs will transport directly without change from one to the other. The "difficult" machines for compatibility are the very first model PET and the VIC: the Jump table is still the same, but differing zero page allocations make it hard to transfer programs without change.

The most important Jump table elements are: $FFD2, for output; $FFE4, for GET; $FFE1, for testing the STOP key. These give you input and output control, and allow you to "guard" loops during debugging.

JSR $FFD2 sends the contents of the A register, a PET-ASCII character, to the output. This is normally the screen, although it can be switched to send to other devices. All registers – A, X, and Y – are preserved.

JSR $FFE4 gets one character and places it into the A register as a PET-ASCII value. It normally gets from the keyboard buffer, but can be switched to receive from other devices. All registers might be changed.

JSR $FFE1 does nothing unless the RUN/STOP key is depressed; in which case it exists to BASIC READY. It's very handy if your program gets stuck in a loop; remember that the JSR $FFE1 must also be in the loop to work. The A register will be changed.

The PET/CBM Switch

We can switch the input or output to devices other than keyboard/screen. Providing the new devices have been OPENed previously (probably in BASIC), we may switch to them by using the logical file number – not the device number.

If we had previously opened a disk file for reading with OPEN 1,8,5,"INFILE,S,R" we may now switch the input to this file with LDX #$01 : JSR $FFC6; later we may restore normal keyboard input with JSR $FFCC. Between these two, we may place as many GET calls (JSR $FFE4) as we wish to receive from the disk file; at a later time, we may switch the file in again and receive more data.

In the same way, we might have opened a printer file for output with OPEN 2,4 and may at any time switch output to the printer with LDX #$02 : JSR $FFC9; later we may switch back to screen output with JSR $FFCC. In between, the JSR $FFD2 call will send to the printer rather than to the screen.

Files may be switched in and out repeatedly as desired. Don't switch input and output at the same time – you would hopelessly confuse the IEEE bus. And remember that when you're finished and return to BASIC, you should CLOSE all open files.

Jump tables, which are seemingly excess code, can greatly enhance program organization and portability. Get into the habit of constructing your own on larger programs. And make a point of knowing the ones provided by your computer manufacturer. ©

ONE ROOF

8315 Firestone Blvd.
DOWNEY, CA.

years

Data Equipment Supply Corp.
8315 Firestone Blvd., Downey, CA 90241 DES

commodore

from the DES SUPPLY DIVISION . . .

— ASK FOR OUR CATALOG —

DISKETTES (5¼ & 8")
- BASF, Memorex, Dysan, Verbatim, Scotch 3M

DES "DATA-DISC"
SPECIAL - as low as $2.00

Ribbons All Types
Mailing Labels - All Sizes
Printer Paper - All Sizes

VOLUMN PRICES for SINGLE USERS

★SOFTWARE HOUSES!
Check us for Quanity $

from DES "VIC-VILLE"© (VIC 20) . . .

***ANNOUNCING!!* DES 48K Expansion Board**........ Available in March

Basic and Machine Language *(3K expansion needed)* *by KAVAN of Germany*
* RENUMBERS * AUTOLINE * FIND * TRACE * HELP
* DISASSEMBLE/ASSEMBLE * DUMP * APPEND/DELETE
* SPECIAL SOUND & VIDEO COMMANDS and more

$69.95

GAMES: From our young professionals (future genius programmers)
Robert Winter - Ralph Orton - Dan Haste - Robert Burnett

- **Goldbrick $12.95** Many levels of play, sound and color.
- **A Maze Ing $10.95** Travel through the maze game of skill and tense action.
- **Gobbler $9.95** Sound Easy? You have 25 seconds to get him and the time gets shorter at each higher level.
- **Hang-U $10.95** Traditional Hangman plays against the VIC's 250 word dictionary OR another person.
- **Coggle $9.95** Computerized version of Boggle. Great for parties. Keeps score for 4 people.
- **Baseball Strategy $10.95** The excitement of base-ball as a video strategic game. Fun for family.
- **Attack on Silo III $10.95** You are the commander of Silo III. Defend your base.
- **Yahtzee $10.95** Solitaire version of this famous dice game, good graphics & lots of fun.

ASK FOR DEALER DISCOUNT!!

CBM 4040 - 8050 DISK DRIVES - 4022 PRINTERS - MODEMS

from the DES LAB . . .

DES Master Switch........................ **$59.95**
4 ROMS in 1 Space
DES Portexpander.......................... **$29.95**
Dual use for the Users Port

Send for our pricing lists on all supplies and peripherals.

DES **DATA EQUIPMENT SUPPLY CORP.**
8315 Firestone Blvd., Downey, CA 90241
(213) 923-9361 (714) 778-5455

PAYMENT
☐ **CHECK #** ____________
☐ **VISA**
☐ **MASTERCHARGE** **Exp. Date** ________
Acct. # ____________
Name ____________
Address ____________
City ____________
State ____________ **Zip** ______

From TYPE-SHARE . . .

Use Your Computer as a Typesetting Terminal!

- You create files on your computer complete with typesetting commands
- Transmit your files to one of our Type-Share Centers for processing.
- We will send you back camera-ready-copy at a minimal cost.

If you don't have a terminal system we'll sell you one complete with software for as low as *$475.00!!*

Call or write TYPE-SHARE INC. for more details (division of DES)

TNW Corp. - Jini Micro Systems, Inc. - Micro Computer Industries Ltd. - Voicetek - The Code Works - CMC - Software by Sasso - HES - Memorex Dysan - NEC - BASF

firm foundation!

(714) 778-5455
(213) 923-9361

PETASCII To ASCII Conversion

Brian Niessen
Chemainus, B.C.

This very useful machine language routine is well worth its weight in gold when it comes to using your PET/CBM wth ASCII devices such as MODEMS and printers. It's short length (56 bytes) and ease of use make it very easy to add to any existing programs. The assembler listing included is for BASIC 3.0 ROM's. The changes needed for BASIC 2.0 ROM's and BASIC 4.0 ROM's are listed below.

For BASIC 2.0, change COMMA to $CE11 from $CDF8, change EVAL to $CCB8 from $CC9F, and change PTR to $96 from $44. For BASIC 4.0, change COMMA to $BEF5 from $CDF8 and change EVAL to $BD98 from $CC9F. The PTR location is the same for BASIC 4.0 as it is for BASIC 3.0, as Commodore promised it would be.

The machine language program works as follows. It is invoked with a SYS(NNN),A$ statement. The (NNN) is the first location of the ML routine and the string variable A$ is the string to be converted. The ML program then jumps to the routine COMMA, which checks for – you guessed it – the comma which seperates the SYS(NNN) and the A$. The next jump is to a subroutine called EVAL which looks at the variable A$ and puts its location in PTR and PTR + 1.

By using an indirect indexed addressing mode, you can get the length of the string by loading the Y register with zero and executing a LDA(PTR),Y. The length of the string is then transferred into the accumulator and saved into location STRLEN. Next, the Y register is incremented so it has a value of one, and another LDA(PTR),Y is executed. This time the LO byte of the beginning of the actual string in memory is returned. This is saved in the location STRLO. Again the Y register is incremented and the LDA(PTR),Y instruction is executed again, returning the HI byte of the location of the string, which is saved in STRHI.

Now knowing the location and length of the string to be converted, the routine can convert it from PETASCII to ASCII on a character by character basis, beginning with the last one and working forward. It uses the following algorithm:

**IF(A)> = 65 AND (A)< = 90 THEN (A) = (A) OR 32
ELSE IF (A)> = 193 AND (A)< = 219 THEN (A) = (A) AND 127**

Characters above $7B have no ASCII equivalent and are left as is.

The subroutine then returns to BASIC, with the contents of the string A$ converted to ASCII. The string can now be printed normally.

One thing to remember is that the original contents of the string are changed. If the string were dynamically declared, as in the following example, it will be changed within the program:

10 A$ = "STRING":SYS(NNN),A$

because the PET, to save memory space, sets the pointers to the actual string in the program. Instead use:

10 A$ = "STRING" + " ":SYS(NNN),A$

which will cause the string to be stored away in high memory.

I hope this will prove as useful to others as it

CBM/PET identifies characters somewhat differently than the industry standard, the ASCII code. In ASCII, the number 65 means uppercase "A," as it does to PET BASIC. However, if you POKE 32768,65, the screen will either display a spade or lowercase "a," depending on whether you are in "text" or "graphics" mode at the time. Everything is kept straight for you by BASIC, but when you try to communicate with a device outside the computer (a MODEM or a printer), some adjustments may need to be made. This is why wordprocessors like WordPro always first ask you to indicate whether you are using a Commodore printer, an ASCII printer, etc. Mr. Niessen's machine language program does the translating. It takes the CBM/PET character codes and modifies them to conform to ASCII. Within the article, he provides the necessary changes to make the program work on all CBM models: Original (2.0), Upgrade (3.0), and BASIC (4.0).

has been to me. It was tested on the EPSON MX-80 printer.

References:

Commodore User's Reference Manual – PN 321604
Page E-1, Paragraph 1

The PET Revealed
Nick Hampshire, Computabits Ltd, P.O. Box 13, Yeovil, Somerset, England

PET/CBM Personal Computer Guide
Adam Osborne-Carroll S. Donahue
Osborne/McGraw-Hill

```
LOC   CODE       LINE
0000             ;**********************************
0000             ;* PETASCII TO ASCII SUBROUTINE   *
0000             ;* BY: BRIAN NIESSEN.             *
0000             ;*                                *
0000             ;* BOX 571                        *
0000             ;* CHEMAINUS, B.C.  VOR 1KO       *
0000             ;* (604) 246-4556                 *
0000             ;**********************************
0000             ;* TO USE: SYS(NNN),A$            *
0000             ;* WHERE (NNN) IS START LOCATION  *
0000             ;* A$ IS STRING TO BE CONVERTED   *
0000             ;**********************************
0000             ; LOCATIONS FOR 2001 (BASIC 3.0)
0000             COMMA  = $CDF8
0000             EVAL   = $CC9F
0000             STRLO  = $DA
0000             STRHI  = $DB
0000             STRLEN = $F9
0000             PTR    = $44          ;POINTER USED IN EVAL ROUTINE
0000                    * = 634        ;STARTING LOCATION FOR ROUTINE
027A  20 F8 CD          JSR COMMA      ;CHECK FOR COMMA
027D  20 9F CC          JSR EVAL       ;EVALUATE EXPRESSION
0280  A0 00             LDY #0         ;SAVE STRING LOCATION
0282  B1 44             LDA (PTR),Y
0284  85 F9             STA STRLEN
0286  C8                INY
0287  B1 44             LDA (PTR),Y
0289  85 DA             STA STRLO
028B  C8                INY
028C  B1 44             LDA (PTR),Y
028E  85 DB             STA STRHI
0290             ; PROCESS STRING ONE CHARACTER AT A TIME.
0290  A4 F9             LDY STRLEN     ;GET STRING LENGTH
0292  B1 DA      LOOP   LDA (STRLO),Y
0294  C9 41             CMP #$41       ; IF $41<=A<=$5A THEN A=A OR $20
0296  90 12             BCC DONE       ; IF 'A <=A<= 'Z THEN A=A OR 32
0298  C9 5B             CMP #$5B
029A  B0 04             BCS SECOND
029C  09 20             ORA #$20
029E  D0 0A             BNE DONE       ;JUMP ALWAYS
02A0             SECOND                ;FIRST FAILED-TRY SECOND
02A0  C9 C1             CMP #$C1       ;IF $C1<=A<=$DA THEN A=A AND $7F
02A2  90 06             BCC DONE
02A4  C9 DB             CMP #$DB
02A6  B0 02             BCS DONE
02A8  29 7F             AND #$7F
02AA  91 DA      DONE   STA (STRLO),Y
02AC  88                DEY
02AD  C0 FF             CPY #$FF
02AF  D0 E1             BNE LOOP
02B1             ;
02B1  60                RTS            ;RETURN TO BASIC
02B2                    .END
```

©

Telecommunications:

How To Use A Modem

Michael E. Day
Chief Engineer, Edge Technology

So you finally went out and bought that MODEM you wanted. Now what do you do?

First of course *read the manual!* This is especially important with MODEMs, as their operation differs from manufacturer to manufacturer. You should read the entire manual even if you don't understand some of it, you will need the information later. You should go back and read it completely again (after you have used your MODEM). Much of the information it contains which may be undecipherable the first time through will be clearer.

One of the first things you will want to do is see if the MODEM works, and how it works. To do this you will need to call a computer so that you can communicate with it.

If you don't have a computer to call, a good way to start is to call one of the Computerized Bulletin Board Systems (or message systems) that is probably in your area. If you don't have a number, ask your local computer store, or go to a computer club meeting and ask someone.

The bulletin board systems are useful in several ways. First, they give you something to do with your MODEM immediately. Second, they tend to hold your interest for a rather long period of time. The first session on a message system often lasts thirty minutes to an hour or more. This can be rather expensive if the call is long distance! Assuming that the call is local, (or you don't care, if that is possible) the long time spent is useful. It will exercise your MODEM, and give you a good sense of how it works.

There are two major ways to communicate via a MODEM. The method just discussed (using a bulletin board system) is referred to as the *conversational mode.* This is because you are, in a sense, "conversing" with the computer. You ask it for some information, or request a specific function, and it responds directly to that request. You are interacting directly with the computer. The other means of communication via the MODEM is the *data transfer mode.* In this mode of operation, the computer is in control of the communications. This is generally used to transfer programs or text between two computers.

The Conversational Mode

The conversational mode is where you are in direct communication with the remote computer. This is similar to sitting in front of your own computer and writing a program. In fact, the only difference is that the computer is at the other end of the phone line.

There are different types of conversational modes employed depending on the system involved. The most common is echoplexing. Echoplexing is often referred to by the physical means of implementing it (full duplex). Although this is common practice, it is not valid since any of the communications methods can be implemented with a full duplex setup. Echoplexing means that, when you type a character, it gets sent out of the MODEM, the remote computer receives it, and echoes it back to you. This way you can verify that it was properly received, since you see every character exactly as the remote computer sees it.

A problem with echoplexing is that if the phone line is weak, noisy, or has excessive propagation delays, (such as very long distance calls) the character that is echoed back can be destroyed or lost by its own reflection, or by the next character that is transmitted. This is solved by not echoing the character. Instead, the character is echoed locally either by the MODEM itself, (this is what the half duplex switch is for on some MODEMs) or by the local computer or terminal. This is referred to as local echo. Although it is often referred to as half duplex, this is not the proper term since half duplex, by definition, is the way the system hardware is set up and not the way the system echoes characters. In fact, if the MODEM is "103 compatible" it is always in full duplex mode even if it has a half duplex switch on. By the way, if you are communicating with an echoplexing (full duplex) system and you have the switch set to half duplex, you will get two characters for every one you type. This is because the MODEM is echoing the character back, and the remote computer is also echoing it back.

Finally, there is a special mode that has grown up on the true half duplex systems. This is referred to as *conversational half duplex*. This is a modified version of local echo. In local echo mode, any character can be sent at any time. In true half duplex communications, only one computer can communicate at any one time, so the local computer cannot respond while the remote computer is transmitting. This can be a problem if the remote computer is sending a bunch of data or stuck in a loop – there is no way to stop it.

This problem is usually corrected in one of two ways. The first method is to add some more hardware in the form of a reverse channel or supervisory channel (a special signal that can be sent back over the phone lines without interfering with the data). It cannot carry any large amount of information and, in fact, is normally only used to request the remote computer to stop what it is doing and return control to the local computer.

The other way that the communications problem is solved is by program control. After the remote computer sends a specific amount of data, it waits for the local computer to indicate if it has received the data or if it wants the data to be sent again, or to have control returned to it. This has the advantage over the reverse channel type systems of being able to send any data again that was improperly received; but it has the disadvantage of being slower.

A response-expected type system is used in packet switching systems (they are, by nature, half duplex type systems) because the response can be shifted in time. That is, several groups of data can be sent with the response to be returned later. In fact, the response can sometimes lag behind the sent data by a number of data groups. This, of course, means that it is important that the groups be properly identified so that it is known which one was bad and needs to be resent.

The data transfer mode is normally seen as having two subdivisions: the send mode and the receive mode. It might be noted at this point that there is another mode that is often grouped with the send and receive modes: the local mode. The local mode is not a form of communication, but rather a mode of non-communication. The local mode is any operation that occurs locally such as preparing the text or program to send, or preparing the computer to receive data. Specifically, if the action involved does not use the MODEM to interact with the other computer, it is a local operation.

When operating in the send or receive modes, one computer must be in the send mode, and the other must be in the receive mode. Except in the case of fully-automated computer systems, the specific mode is normally selected by the user at the beginning of the operation while in local mode. The computer then takes over and establishes communication with the other computer and transfers the desired information.

The operation of the data transfer mode can be quite complex. The first action must be the establishment of synchronization with the other computer. This is done by the designated primary computer (depending on the system involved, this could be the sending computer or the receiving computer) by sending out a periodic request to the other computer. When the other computer answers, they then become synchronized. This means that the two computers are now listening to each other and are in a position to respond to requests from the other.

The next action is dependent upon the type of system involved. In a non-automatic system, it is assumed that the information has been predefined at both computers and no information needs to be sent to define it. In an automatic system, the predefinition is not assumed, and so the information must be transferred. Generally this consists of the program (or file) name of the data to be transferred.

Next, the data itself is sent. It is usually broken up into small pieces during the sending process. The reason for breaking up the data is to reduce the retransmission time involved if an error is encountered. This way, only a small piece has to be retransmitted if an error occurs, not the entire program.

Normally, the size of these pieces is related to the computer, and typical sizes are 128, 256, or 1024 bytes. The piece of information is referred to as a *record*. The exact structure of the record varies from system to system; there is no standard yet developed. In time a common standard will develop. Until then, we will have to live with what is out there.

In my next column I will describe a specific application of program transfer with a MODEM. ©

Customizing Apple's COPY Program

Roger B. Chaffee
Menlo Park, California

How many times have you used the COPY or COPYA program from the Apple DOS Master Disk? How many times have you had to tell the program that you wanted Slot 6, Drive 1, even though you have only one drive, and it's *always* in Slot 6, Drive 1? Every time, right?

Well, the nice people at Apple who brought you the COPY programs have made a system that works well, and is also easy to modify. Many programs, like Muffin and the DOS itself, are written in machine language (ML), and are very difficult to modify, or even to examine and understand. The main routines of COPY and COPYA, however, are written in BASIC, and only the nitty-gritty of buffer management and interface with the RWTS routine are done in ML.

Take a look at COPYA, which is written in AppleSoft BASIC, or at COPY if you want to use Integer BASIC. To look at the program, type LOAD COPYA [LOAD COPY] and then LIST. You can also list specific line numbers or ranges. Here's an explanation of some of the program lines. Later, we'll look at how to change them to fit your own system.

(Some of the information in this article depends on which program you are looking at, as in the LOAD commands above. When this is true, the information given is for the COPYA program, and the information for COPY follows it in brackets.)

Line 70 [90]: Load in the ML routines. (There is a hidden CTRL-D in this line, which will disappear if you use the BASIC editor to replace the line.)

Line 90 [120]: CALL 704 to initialize the ML routines. CS gets the current slot number, which is probably six, and location 720 gets the current drive number, either one or two. The values come from the IOB used by the DOS.

Lines 100,110 [130,140]: Set locations 715, 716 to the page numbers of the first and last pages that the ML routines will use for buffer space. They are different for AppleSoft and Integer because the two BASICs use memory and pointers differently.

Line 130 [150]: Call subroutines to ask the user for the slot and drive of the Master ("Original") diskette. MS gets the Master Slot number (zero to seven), and MD gets the Master Drive number (one or two).

Line 132 [160]: Call subroutines to ask the user for the slot and drive of the Slave ('Duplicate') diskette. SS gets the Slave Slot number (one to seven), and SD gets the Slave Drive number (one or two).

Line 165 [190]: Here's where the program stops asking for information, and starts copying the disk. Do not modify anything from here through the END statement in Line 305 [420] unless you are very sure of what you are doing! Before this line, MS and MD must be set to the slot and drive numbers for the Master diskette, and SS and SD must be set to the slot and drive numbers for the Slave diskette.

Line 310 [430]: This subroutine asks for a slot number, and is called for both Master and Slave slots. It also prints information on the screen, so you shouldn't simply replace it by a RETURN statement.

Line 320 [440]: This subroutine asks for a drive number, and is called for both Master and Slave slots. Again, don't just remove it, because it prints as well as asking you for the number.

Line 330 to 340 [450 to 460]: This subroutine gets a one-digit number, from L to H, from the keyboard. If RETURN is pressed, it uses the number already in N.

Line 350 [470]: This subroutine prints "DEFAULT=" and then puts on a flashing cursor.

Line 360 [480] to the end: This subroutine prompts you to change diskettes, if MS=SS and MD=SD. Otherwise, it just returns, with no action.

Okay, so that's how COPY gets it's parameters. But how can you change things so that it works just right for your own Apple system, which has a fixed number of drives in fixed slot locations?

Case 1: One Drive

The first case assumes you have only one drive, which is always in slot 6, drive 1. There is no choice. You always want slot 6, drive 1. The simplest way to make this happen is probably to replace lines 130 and 132 [150 and 160] by the statement

```
MS=6: SS=6: MD=1: SD=1
```

Sams offers five books designed to help you explore the basic workings of your Apple microcomputer. Here you'll find all of the information necessary to understand and use the limitless computing power of the Apple system. With these great books from Sams, your venture into the world of the Apple microcomputer will be a fruitful one!

APPLESOFT LANGUAGE Beginning programmers are given a comprehensive, step-by-step introduction to Applesoft, the language of the Apple II microcomputer. No. 21811. $10.95

INTIMATE INSTRUCTIONS IN INTEGER BASIC This complete introductory overview of Integer BASIC enables even the most nontechnical reader to write and use simple programs. No. 21812. $7.95

6502 SOFTWARE DESIGN The 6502 microprocessor is the heart of your Apple microcomputer. 6502 Software Design introduces you to programming in 6502 assembly language allowing you to perform high-speed processing and peripheral control not generally possible with higher-language programs. No. 21656. $12.95

MOSTLY BASIC: APPLICATIONS FOR YOUR APPLE® II Twenty-eight BASIC language programs for your Apple II microcomputer are thoroughly outlined and discussed. These programs are useful in all facets of the home and business. No. 27189. $12.95

APPLE® INTERFACING You are provided with real, tested circuits which interface to your Apple microcomputer for remote control, data acquisition, and control of external electronic and electromechanical devices. No. 21862. $10.95

Each book presents a different slice of the intricate workings of your Apple microcomputer. To take full advantage of this powerful machine, you must understand it from the inside out.

To order these Sams books or to get the name of your local Sams retailer, call toll free and reference this number: AD163

800-428-3696

SAMS BOOKS™

Howard W. Sams & Co., Inc.
4300 West 62nd Street, P.O. Box 7092
Indianapolis, IN 46206

GET TO THE CORE OF YOUR APPLE®

Offer good in USA only and expires 7/31/82. Prices subject to change without notice.

but that won't give you the nicely formatted screen to tell you what's happening. Here's a better way: replace the subroutine which gets the input values.

```
AppleSoft:    330 K=141
[Integer:     450 REM]
```

Then when any value is needed, the default will be used. To make it work just right, you also need to go back to line 132 [160] and replace the statement N=3-MD by the statement N=MD, which will set the slave drive to the same value (1) as the master drive.

Case 2: Two Drives The Easy Way

Suppose you have two drives, and you want the program to decide which gets the master and which gets the slave diskette. Make the same subroutine replacement as before:

```
AppleSoft:    330 K=141
[Integer:     450 REM]
```

Now whichever drive you used most recently will be the master, and the other will be the slave.

Case 3: Two Drive The Right Way

Finally, suppose you have two drives, both in the same slot, but you want to be able to choose which one will be the master. This time you can't just remove the subroutine which gets the numbers, because you want the program to ask you for the drive number. Instead, you can stop the program from asking you for the slot numbers:

AppleSoft:
Change GOSUB 330 to GOSUB 340 in Line 310.
[Integer:
Change GOSUB 450 to GOSUB 460 in Line 430.]

In this last case, to make sure that the slave drive is the "other" one, that is the one you didn't specify for the master, the simplest change is this:

```
AppleSoft:
  Move statement 330 to 331
  Insert
    330 IF LEFT$(I$,1)="D" THEN 340
[Integer:
  Move statement 450 to 451
  Insert
    450 IF I$(1,1)="D" THEN 460
```

Leave the rest of the line alone.

Apology And Exhortation

As you read this, it probably sounds more complicated than it really is. If you have done any BASIC programming on your Apple, you already know how to modify programs, and COPY is just another BASIC program. There's no reason that you should stick with a general-purpose program, when it is easy and maybe even instructive to fit the Apple's general program to your specific needs. ©

Odds And Ends

With each CBM/PET disk drive comes a program to simplify using disk files. It's popularly called "The Wedge."

The Bug In The Universal Wedge

Keith Peterson
Kansas City Computers Inc.
Independence, MO

Most people are familiar with Universal Wedge, the program for easier disk access that comes on the disk with all of Commodore's disk drives. But most people don't realize that there is a bug in the Wedge that can cause some trouble when the Wedge is used on a 4.0 BASIC system.

The real cause of the problem is a small number of Original BASIC systems that required special handling when loading a file. These systems had what is known as the -04 ROMs, and several of the first Original BASIC PETs had them.

Universal Wedge checks a location in the BASIC system to see if it contains the -04 ROMs. If so, the Wedge increments the end of BASIC pointer by one, fixing the -04 ROM bug.

All this is fine until you use the Wedge in a 4.0 BASIC system. Then the Wedge, looking for the -04 ROMs, finds in the 4.0 BASIC ROM what it thinks is an indication of the Original BASIC -04 ROM, and increments the end of BASIC pointer accordingly. So now the program is one byte too long.

After loading a program, changing it, and saving it back out several times, a large number of extra bytes accumulates on the end of the program. This can interfere with or crash certain machine language routines, use up memory, and cause some confusion when you're looking for the end of the program. It will also cause some append routines to fail to function.

What's the fix? Simple. Just follow these steps:

1. Type: LOAD "universal wedge",8
2. Type: POKE 2109,133
3. Type: SAVE "fixed wedge",8

You now have a version of Universal Wedge called "fixed wedge" that will work correctly on either Original BASIC or 4.0, and with the -04 ROMs as well. ©

This helpful POKE will work on any CBM/PET with Toolkit installed.

Improving The Toolkit's TRACE Function

Robert Lenoil
Brooklyn, NY

Don't all you Toolkit owners wish that you could turn TRACE and STEP on and off from within your programs? After all, why TRACE an entire program, when you only need to debug one small section? Well, here's good news: The TRACE and STEP functions can be turned on and off from BASIC, by means of a simple POKE command! Here's why:

The Toolkit uses memory location 124 (206 for small keyboard PETs) as a flag for the status of the TRACE function: 0 = off, 2 = trace, 3 = step. All you have to do is POKE 124 (206) with the proper number, to accomplish the corresponding Toolkit function. In other words:

```
POKE 124,0 = OFF
POKE 124,2 = TRACE
POKE 124,3 = STEP
```

One caution: if the Toolkit is not initialized, any of the above POKEs will crash the PET. Therefore, it would be wise to execute this line before any of the POKEs:

```
IF PEEK(124) = 10 THEN SYS xxxxx
```

(where xxxxx is the number used to initialize your Toolkit). ©

This program uses the Atari's NOTE and POINT commands (which permit fast reading and writing with the disk drive). The program runs on an Atari 400/800 with at least 16K of RAM, an Atari 810 disk drive, and an optional printer.

Atari Mailing List Program

Garry J. Patton
Romulus, NY

This program is written to take advantage of the NOTE and POINT capabilities of the Atari. It is relatively fast and requires the 810 disk drive with DOS II and 16K (if only 75 records are desired). On a single diskette, 750 records can be accessed if needed. This requires 24K. The program is versatile and can be used for any home use for maintaining a list of important addresses. One of my gripes about current commercial mailing list programs is the time required to get at an address quickly. By the time the machine is powered up, the program loaded, and the list searched, as much as two to three minutes have gone by. That seems a little excessive. This program stores all addresses on the diskette and can access them quickly. It doesn't require loading in the program and then loading in the data file. It also doesn't search every file – only until it finds the appropriate address. It then stops and prints the name.

For most of us, 50-100 names are all that are required to keep up our Christmas list. However, if you have an exhaustive list up to 750 addresses can be stored on a single diskette if desired. As the number of addresses on file increases, the initial start up time increases; thus, you lose the advantage of quickly finding that missing address. So only create the number of files that you think you will need. Presently the program is set for 100 names.

Because of its uniform file structure, this program can easily be adapted to search for any major field and to sort by each field. However, these niceties are not generally required by most of us. In order to keep RAM to a minimum, these "extras" were not included.

The program has several error trapping features. If you encounter an error, just type GOTO MENU and try again. The TRAP command is an excellent tool to prevent those unwanted crashes. When writing a program, this command can help keep your program out of trouble.

The heart of this program is the FILE$. The Atari has excellent string handling capabilities which are used to our advantage here. Each subset of the FILE$ is a major field. As mentioned above, each of these fields could easily be searched for individually and sorted or manipulated as desired with only the addition of a short additional subroutine. The FILE$ is structured:

FILE$(1,15) = LAST NAME
FILE$(16,25) = FIRST NAME
FILE$(26,49) = STREET ADDRESS
FILE$(50,69) = CITY
FILE$(70,71) = STATE
FILE$(72,81) = ZIP
FILE$(82,95) = PHONE
FILE$(96,116) = EXTRA

A data file called MAILDATA.FIL is created to store each record. An error trapping routine alerts you if the data file is not on your diskette, and then it asks if you want to create one. Be sure you have enough sectors left on your diskette before creating the file. (It requires .94 SECTORS for each record. Thus, if you want 100 records, have 94 SECTORS left.) The number of records created is controlled by line 87. Set the FOR/NEXT loop to the number desired if you want different than 100 records. (CAUTION! Be sure then that lines 13, 87, 352, 604, and 2501 all match! These lines all have references to the number of records.) If you want more or fewer records, be sure to change the appropriate lines.

Each record's position in the file is stored in two arrays named SECTOR and BYTE. The reason for the need for increased memory for more records is that these arrays must be DIM'ed for the proper amount. This requires memory. The more records created (say 750 instead of 100) the more memory you will need. Remember, too, that your start-up time will increase because at initialization each record must be counted and its location stored in the appropriate array.

Lines 10-30 – Initializes the program, opens the screen editor, the keys, and MAILDATA.FIL

Lines 75-89 – if MAILDATA.FIL is not located on the diskette, you are given the choice of creating the file or ending. 100 records are created with a dummy FILE$ of +'s. Each record's position is NOTED and stored in the arrays SECTOR and BYTE for future reference. POKE 559,0 turns off ANTIC (the screen goes blank because ANTIC controls the screen updating). With ANTIC off 6502 can go merrily about its way doing the task at hand without interruptions from ANTIC. This can save up to 30% of the time required for com-

pletion. POKE 559,34 turns ANTIC back on. MAILDATA.FIL is now created and ready to receive your entries.

Lines 100-354 – This is the heart of the program. With the appropriate prompts each field of FILE$ is created here. A short subroutine at 1000-1010 causes the cursor position to flash, advances the cursor, and assigns the ATASCII value of the key struck to the variable XX. If the delete key (126) is struck, the flag Q is set; then the entire field is erased, and you start again. If the return key (155) is struck, this terminates the entry to the field.

Each field is assigned a discrete length and is protected to that length. If you go over, an error message routine at 700-708 tells you of your error, and you start again. If the field you enter is less than the prescribed length, the field is padded with blanks until it equals the proper length.

EDIT is a flag that says you've come here from some other subroutine in the program and allows return to that area. Flags allow one to give a program versatility and to reduce RAM useage as an area of your program can be used for several purposes. This reduces repetition. You'll find flags used throughout this program to direct you in and out of various areas smoothly depending on your needs.

You'll note that this program is "with the times" and allows for the new ZIP + four, go ahead and use it. If not, it will be here soon. Space is reserved for the additional + *four*. So you can add it later.

Line 344 allows you to change your mind before the record is committed to the disk. If you keep the file, the next free space is POINTED to by the SECTOR/BYTE arrays, and your file is now stored for your future use.

Lines 500-570 – This is the main MENU giving you your choice of options to operate the program. Each choice sends you to the appropriate subroutine to complete your task as requested.

Lines 600-610 – At initialization, if MAILDATA.FIL is available on the disk, each record is looked at to determine each record's position, to determine the number of names on file to date, and to create the SECTOR/BYTE arrays for future use.

Lines 2000-2504 – This subroutine allows you to search for a name. You can search by last name only or first and last names. Using last name only allows faster execution time because fewer manipulations are required, but, if you have ten Smith's, the first and last names will be needed to return the correct one. Otherwise, the first Smith encountered will be printed.

Lines 2390-2410 look at the console keys and allow you to page through the data file, record by record. This is a nice convenience that will allow you to occasionally page through your listings and find outdated files or the names of "ex-friends" whose name you don't want any longer.

Lines 4000-4055 – This subroutine is used for the printer. You can print out one record at a time or you can choose to print out all of the addresses you have stored on file. You will be asked how many lines you want between records (1-9). This will allow for labels to be printed to cut down the drudgery of addressing all those Christmas cards.

```
1 REM ***  MAILING LIST PROGRAM  ***
2 REM ***           by           ***
3 REM ***      GARRY PATTON      ***
4 REM ***       PO BOX 137       ***
5 REM *** ROMULUS, NY 14541+0137 ***
6 REM
7 REM
10 DIM FILE$(116),LN$(16),FN$(11),ST$(25
),PHONE$(15),C$(21),STATE$(3),ZIP$(11),E
XTRA$(22),TAB$(39)
13 DIM SECTOR(100),BYTE(100):TAB$="
                                  "
20 OPEN #3,4,0,"S:":OPEN #1,4,0,"K:":Z=1
4:MENU=500:EDIT=0
30 TRAP 75:OPEN #2,12,0,"D:MAILDATA.FIL"
:TRAP 40000:RN=0:GOTO 600
75 ? CHR$(253):? :? "MAILDATA.FIL IS NOT
 ON THIS DISK":POKE 752,1:? "DO YOU WANT
 TO CREATE MAILDATA.FIL"
76 ? "(Y or N)"
78 GET #1,K:IF K<>78 THEN IF K<>89 THEN
75
81 IF K=78 THEN END
82 TRAP 95:CLOSE #2:OPEN #2,8,0,"D:MAILD
ATA.FIL"
83 POKE 559,0:FILE$="":FOR ZZ=1 TO 116:F
ILE$(ZZ,ZZ)="+":NEXT ZZ:FOR ZZ=1 TO 100:
NOTE #2,S,B
89 SECTOR(ZZ)=S:BYTE(ZZ)=B:? #2;FILE$:NE
XT ZZ:POKE 559,34:CLOSE #2:OPEN #2,12,0,
"D:MAILDATA.FIL":GOTO MENU
95 POKE 559,34:? CHR$(253):? :? "  ERROR
!!":END
100 REM CREATE FILE$
110 FILE$="":? "{CLEAR}"
112 O=0:LN$="":X=2:Y=4:POSITION X,Y:? "L
AST NAME: ":FOR ZZ=14 TO 28:POSITION ZZ,
Y:? "-":NEXT ZZ
```

```
114 POSITION Z+O,Y:GET #3,W:GOSUB 1000:I
F Q=1 THEN Q=0:GOTO 112
115 IF XX=155 THEN 130
119 LN$(LEN(LN$)+1)=CHR$(XX):IF LEN(LN$)
>15 THEN GOSUB 700:GOTO 112
126 GOTO 114
130 IF LEN(LN$)<15 THEN LN$(LEN(LN$)+1)=
" ":GOTO 130
136 IF EDIT THEN RETURN
140 FILE$(LEN(FILE$)+1)=LN$
150 O=0:FN$="":Y=5:POSITION X,Y:? "FIRST
 NAME: ":FOR ZZ=14 TO 23:POSITION ZZ,Y:?
 "-":NEXT ZZ
154 POSITION Z+O,Y:GET #3,W:GOSUB 1000:I
F Q=1 THEN Q=0:GOTO 150
156 IF XX=155 THEN 162
158 FN$(LEN(FN$)+1)=CHR$(XX):IF LEN(FN$)
>10 THEN GOSUB 700:GOTO 150
161 GOTO 154
162 IF LEN(FN$)<10 THEN FN$(LEN(FN$)+1)=
" ":GOTO 162
166 IF EDIT THEN RETURN
170 FILE$(LEN(FILE$)+1)=FN$
176 O=0:ST$="":Y=6:POSITION X,Y:? "STREE
T: ":FOR ZZ=14 TO 37:POSITION ZZ,Y:? "-"
:NEXT ZZ
180 POSITION Z+O,Y:GET #3,W:GOSUB 1000:I
F Q=1 THEN Q=0:GOTO 176
186 IF XX=155 THEN 194
188 ST$(LEN(ST$)+1)=CHR$(XX):IF LEN(ST$)
>24 THEN GOSUB 700:GOTO 176
192 GOTO 180
194 IF LEN(ST$)<24 THEN ST$(LEN(ST$)+1)=
" ":GOTO 194
196 IF EDIT THEN RETURN
200 FILE$(LEN(FILE$)+1)=ST$
204 O=0:C$="":Y=7:POSITION X,Y:? "CITY:
":FOR ZZ=14 TO 33:POSITION ZZ,Y:? "-":NE
XT ZZ
208 POSITION Z+O,Y:GET #3,W:GOSUB 1000:I
F Q=1 THEN Q=0:GOTO 204
216 IF XX=155 THEN 224
218 C$(LEN(C$)+1)=CHR$(XX):IF LEN(C$)>20
 THEN GOSUB 700:GOTO 204
220 GOTO 208
224 IF LEN(C$)<20 THEN C$(LEN(C$)+1)=" "
:GOTO 224
226 IF EDIT THEN RETURN
230 FILE$(LEN(FILE$)+1)=C$
234 O=0:STATE$="":Y=8:POSITION X,Y:? "ST
ATE: ":POSITION Z,Y:? "--"
236 POSITION Z+O,Y:GET #3,W:GOSUB 1000:I
F Q=1 THEN Q=0:GOTO 234
242 IF XX=155 THEN 250
243 STATE$(LEN(STATE$)+1)=CHR$(XX):IF LE
N(STATE$)>2 THEN GOSUB 700:GOTO 234
244 GOTO 236
250 IF LEN(STATE$)<>2 THEN ? CHR$(253):P
OSITION X,Y:? TAB$:GOTO 234
256 IF EDIT THEN RETURN
260 FILE$(LEN(FILE$)+1)=STATE$
264 O=0:ZIP$="":Y=9:POSITION X,Y:? "ZIP:
 ":POSITION Z,Y:? "-----+----"
266 POSITION Z+O,Y:GET #3,W:GOSUB 1000:I
F Q=1 THEN Q=0:GOTO 264
270 IF XX=155 THEN 274
271 ZIP$(LEN(ZIP$)+1)=CHR$(XX):IF LEN(ZI
P$)>10 THEN GOSUB 700:GOTO 264
273 GOTO 266
274 IF LEN(ZIP$)<10 THEN ZIP$(LEN(ZIP$)+
1)=" ":GOTO 274
275 IF EDIT THEN RETURN
276 FILE$(LEN(FILE$)+1)=ZIP$
280 O=1:Y=10:POSITION X,Y:? "PHONE: ":PH
ONE$="(---) --- ----":POSITION Z,Y:? PHO
NE$
282 POSITION Z+O,Y:GET #3,W:GOSUB 1000:I
F Q=1 THEN Q=0:GOTO 280
284 IF XX=155 THEN 294
287 IF O=4 THEN PHONE$(O,O)=CHR$(XX):O=6
:GOTO 282
288 IF O=9 THEN PHONE$(O,O)=CHR$(XX):O=1
0:GOTO 282
292 PHONE$(O,O)=CHR$(XX):IF O<14 THEN 28
2
294 IF LEN(PHONE$)<14 THEN PHONE$="(---)
 --- ----"
300 IF EDIT THEN RETURN
304 FILE$(LEN(FILE$)+1)=PHONE$
310 O=0:EXTRA$="":Y=11:POSITION X,Y:? "E
XTRA: ":FOR ZZ=14 TO 34:POSITION ZZ,Y:? "
-":NEXT ZZ
314 POSITION Z+O,Y:GET #3,W:GOSUB 1000:I
F Q=1 THEN Q=0:GOTO 310
318 IF XX=155 THEN 332
322 EXTRA$(LEN(EXTRA$)+1)=CHR$(XX):IF LE
N(EXTRA$)>21 THEN GOSUB 700:GOTO 310
328 GOTO 314
332 IF LEN(EXTRA$)<21 THEN EXTRA$(LEN(EX
TRA$)+1)=" ":GOTO 332
334 IF EDIT THEN RETURN
336 FILE$(LEN(FILE$)+1)=EXTRA$
340 IF LEN(FILE$)<>116 THEN ? CHR$(253):
? :? "ERROR IN FILE$!!":FOR DELAY=1 TO 5
00:NEXT DELAY:GOTO 100
344 POSITION 3,15:? "File as shown (Y or
 N)?":POSITION 3,16:? "N cancels this pr
inting.":GET #1,K
346 IF K<>78 THEN IF K<>89 THEN 344
348 IF K=78 THEN ? "{CLEAR}":GOTO MENU
352 RN=RN+1:IF RN>100 THEN ? CHR$(253):G
OTO 400
```

```
353 POINT #2,SECTOR(RN),BYTE(RN):PRINT #
2;FILE$:GOTO MENU
354 PRINT #2;FILE$:GOTO MENU
400 ? "{CLEAR}":POSITION 4,10:? "I'm sor
ry, but MAILDATA.FIL is full!":FOR D=1 T
O 500:NEXT D:GOTO MENU

500 ? "{CLEAR}":POSITION 7,3:? "CHOOSE A
PPROPRIATE NUMBER":POSITION 12,0:? "RECO
RDS ON FILE ";RN
510 POSITION 4,5:? "1 Add New Name":POSI
TION 4,6:? "2 Edit Name":POSITION 4,7:?
"3 Delete Name"
520 POSITION 4,8:POKE 752,1:? "4 Print N
ame":GET #1,K
530 IF K<>49 THEN IF K<>50 THEN IF K<>51
 THEN IF K<>52 THEN 500
532 IF K=51 THEN 4000
534 IF K=49 THEN 100
536 IF K=50 THEN 3000
540 POSITION 4,12:? "WILL YOU BE USING P
RINTER (Y or N)?":GET #1,KK
541 IF KK=78 THEN 2000
542 POSITION 4,14:? "WILL YOU LPRINT ALL
 OR SOME (A or S)":GET #1,KKK
543 IF KKK=65 THEN GOTO 5000
570 GOTO 2000
600 REM COUNT RECORDS ON FILE
604 TRAP 95:POKE 559,0:FOR NR=1 TO 100:N
OTE #2,S,B:SECTOR(NR)=S:BYTE(NR)=B
606 INPUT #2,FILE$:IF FILE$(1,5)<>"+++++
" THEN RN=RN+1
610 NEXT NR:POKE 559,34:TRAP 40000:GOTO
MENU
700 ? CHR$(253)
704 FOR ZZ=1 TO 4:POSITION X,Y:? TAB$:FO
R D=1 TO 30:NEXT D:POSITION X,Y:? "ITEM
TOO LONG! STAY WITHIN -'S!"
708 FOR D=1 TO 100:NEXT D:NEXT ZZ:POSITI
ON X,Y:? TAB$:RETURN
1000 POSITION Z+O,Y:? CHR$(W)
1004 IF PEEK(764)=255 THEN POSITION Z+O,
Y:? " ":GOTO 1000
1006 GET #1,XX:IF XX=126 THEN Q=1:RETURN

1010 POSITION Z+O,Y:? CHR$(XX):POKE 764,
255:O=O+1:RETURN
2000 REM PRINT NAME
2010 ? "{CLEAR}":POSITION 3,5:? "DO YOU
WANT TO SEARCH FOR:":POSITION 10,7:? "1
Last name only"
2020 POSITION 10,8:? "2 Last name and fi
rst name"
2030 GET #1,K:IF K<>49 THEN IF K<>50 THE
N 2000
2040 POSITION 3,12:? "LAST NAME: ";:INPU
T LN$
2050 IF LEN(LN$)>15 THEN ? CHR$(253):POS
ITION 3,12:? TAB$:POSITION 3,12:? "NAME
TOO LONG!":FOR D=1 TO 500:NEXT D
2051 IF LEN(LN$)>15 THEN POSITION 3,12:?
 TAB$:GOTO 2040
2070 IF K=49 THEN FN$=" ":GOTO 2200
2080 POSITION 3,14:? "FIRST NAME:";:INPU
T FN$
2090 IF LEN(FN$)>10 THEN ? CHR$(253):POS
ITION 3,14:? TAB$:POSITION 3,14:? "NAME
TOO LONG!":FOR D=1 TO 500:NEXT D
2091 IF LEN(FN$)>10 THEN POSITION 3,14:?
 TAB$:GOTO 2080
2200 FOR P=1 TO RN:POINT #2,SECTOR(P),BY
TE(P):INPUT #2,FILE$
2210 IF FILE$(1,LEN(LN$))=LN$ AND K=49 T
HEN POP :GOTO 2300
2220 IF FILE$(1,LEN(LN$))=LN$ AND FILE$(
16,15+LEN(FN$))=FN$ THEN POP :GOTO 2300
2225 NEXT P
2230 ? "{CLEAR}":POSITION 3,10:? "I'm so
rry but I do not find:":? :? "   ";FN$;"
 ";LN$
2240 FOR D=1 TO 500:NEXT D:GOTO MENU
2300 ? "{CLEAR}":POSITION 12,0:? "RECORD
 NUMBER ";P
2305 POSITION 2,4:? "LAST NAME: ":POSITI
ON 14,4:? FILE$(1,15)
2310 POSITION 2,5:? "FIRST NAME: ":POSIT
ION 14,5:? FILE$(16,25)
2320 POSITION 2,6:? "STREET: ":POSITION
14,6:? FILE$(26,49)
2330 POSITION 2,7:? "CITY: ":POSITION 14
,7:? FILE$(50,69)
2340 POSITION 2,8:? "STATE: ":POSITION 1
4,8:? FILE$(70,71)
2350 POSITION 2,9:? "ZIP: ":POSITION 14,
9:? FILE$(72,81)
2360 POSITION 2,10:? "PHONE: ":POSITION
14,10:? FILE$(82,95)
2370 POSITION 2,11:? "EXTRA: ":POSITION
14,11:? FILE$(96,116)
2372 IF KK=89 THEN KK=0:GOSUB 4500
2374 IF EDIT THEN RETURN
2378 POKE 53279,8
2380 POSITION 4,20:? "Press |OPTION| to
page forward":POSITION 4,21:? "Press |SE
LECT| to page back"
2384 POSITION 4,22:POKE 752,1:? "Press |
START|  to return to menu"
2390 IF PEEK(53279)=5 THEN P=P-1:GOSUB 2
500:GOTO 2300
2400 IF PEEK(53279)=3 THEN P=P+1:GOSUB 2
500:GOTO 2300
2410 IF PEEK(53279)=6 THEN GOTO MENU
```

```
2420 GOTO 2390
2500 IF P<1 THEN P=1
2501 IF P>100 THEN P=100
2504 POINT #2,SECTOR(P),BYTE(P):INPUT #2
,FILE$:RETURN
3000 REM EDIT
3010 EDIT=1:GOSUB 2000
3020 POSITION 10,13:? "|To EDIT| Choose
one":POSITION 4,15:? "1 LAST"
3030 POSITION 20,15:? "2 FIRST":POSITION
 4,16:? "3 STREET":POSITION 20,16:? "4 C
ITY"
3040 POSITION 4,17:? "5 STATE":POSITION
20,17:? "6 ZIP":POSITION 4,18:? "7 PHONE
":POSITION 20,18:? "8 EXTRA"
3050 POSITION 4,21:? "S |SAVE| as shown"
:POSITION 4,20:? "E |EXIT|"
3055 GET #1,K:IF K=83 THEN 3200
3060 IF K=49 THEN GOSUB 112:FILE$(1,15)=
LN$:GOTO 3020
3065 IF K=50 THEN GOSUB 150:FILE$(16,25)
=FN$:GOTO 3020
3070 IF K=51 THEN GOSUB 176:FILE$(26,49)
=ST$:GOTO 3020
3075 IF K=52 THEN GOSUB 204:FILE$(50,69)
=C$:GOTO 3020
3080 IF K=53 THEN GOSUB 234:FILE$(70,71)
=STATE$:GOTO 3020
3085 IF K=54 THEN GOSUB 264:FILE$(72,81)
=ZIP$:GOTO 3020
3090 IF K=55 THEN GOSUB 280:FILE$(82,95)
=PHONE$:GOTO 3020
3095 IF K=56 THEN GOSUB 310:FILE$(96,116
)=EXTRA$:GOTO 3020
3097 IF K=69 THEN EDIT=0:GOTO MENU
3110 GOTO 3055
3200 POINT #2,SECTOR(P),BYTE(P):FOR D=1
TO 50:NEXT D:PRINT #2;FILE$
3230 EDIT=0:GOTO MENU
4000 REM DELETE NAME
4010 EDIT=1:GOSUB 2000
4020 POSITION 4,15:? "Do you want to |DE
LETE| this file       (Y or N)?"
4030 GET #1,K:IF K<>78 THEN IF K<>89 THE
N 4020
4040 IF K=78 THEN EDIT=0:POSITION 4,18:?
 "File |NOT| deleted":FOR D=1 TO 500:NEX
T D:GOTO MENU
4050 POINT #2,SECTOR(RN),BYTE(RN):INPUT
#2;FILE$:POINT #2,SECTOR(P),BYTE(P):PRIN
T #2;FILE$
4055 FOR ZZ=1 TO 116:FILE$(ZZ,ZZ)="+":NE
XT ZZ:POINT #2,SECTOR(RN),BYTE(RN):PRINT
 #2;FILE$:RN=RN-1:EDIT=0:GOTO MENU
4500 T=0:FOR ZZ=16 TO 25:IF FILE$(ZZ,ZZ)
<>" " THEN T=T+1
4510 NEXT ZZ
4530 TT=0:FOR ZZ=50 TO 69:IF FILE$(ZZ,ZZ
)<>" " THEN TT=TT+1
4540 NEXT ZZ:LPRINT FILE$(16,15+T);" ";F
ILE$(1,15):LPRINT FILE$(26,49)
4570 LPRINT FILE$(50,49+TT);", ";FILE$(7
0,71);" ";FILE$(72,81):RETURN
5000 POSITION 4,16:? "HOW MANY LINES BET
WEEN ADDRESSES":POSITION 4,17:? "(1-9)":
GET #1,K:IF K<49 OR K>57 THEN 5000
5005 KKKK=K-48
5010 FOR ZZ=1 TO RN:POINT #2,SECTOR(ZZ),
BYTE(ZZ):INPUT #2;FILE$
5012 T=0:FOR QQ=16 TO 25:IF FILE$(QQ,QQ)
<>" " THEN T=T+1
5013 NEXT QQ
5014 TT=0:FOR QQ=50 TO 69:IF FILE$(QQ,QQ
)<>" " THEN TT=TT+1
5015 NEXT QQ
5020 LPRINT FILE$(16,15+T);" ";FILE$(1,1
5):LPRINT FILE$(26,49)
5030 LPRINT FILE$(50,49+TT);", ";FILE$(7
0,71);" ";FILE$(72,81)
5040 FOR ZZZ=1 TO KKKK:LPRINT :NEXT ZZZ
5050 NEXT ZZ:KK=0:KKK=0:KKKK=0:GOTO MENU
```

©

Browsing The VIC Chip

Jim Butterfield
Toronto, Canada

The computer is called VIC, for Video Interface Computer ... but there's a chip inside which is also called VIC, for Video Interface Chip. The chip bears the number 6560 or 6561; it's used to make good things happen on your television screen.

Beginners often don't realize that memory addresses are used for more than memory storage. In the VIC computer, addresses 36864 to 36879 may be PEEKed or POKEd. These locations are not used for memory – they hold controlling information for the VIC chip. We're going to look through those addresses, experimenting as we go. We may learn some new things about our computer.

• *Location 36864* (Hex 9000). Values 0 to 127 set the position of the left border on your screen. The usual value is five. Try the following quick "slide change" line:

```
FOR J=5 TO 30:POKE 36864,J:NEXT J
  :FOR J=30 TO 5 STEP -1:POKE 36864,J:NEXT J
```

If you add 128 to the value in 36864, the screen will go into interlace mode. In most cases, all you'll notice if you POKE 36864,133 is a little "dither" in the screen detail. However, a few television sets are built in such a way that they won't work unless you set interlace mode with this POKE.

• *Location 36865* (Hex 9001). Value 0 to 255 set the position of the top border on your screen. The usual value is 25. Try making the screen "curtsy" with:

```
FOR J=25 TO 45:POKE 36865,J:NEXT J
  :FOR J=45 TO 25 STEP -1:POKE 36865,J:NEXT J
```

• *Location 36866* (Hex 9002). Part of this location tells the chip how many columns on the screen. This will always be 22. But there's an extra – a value of 128 may be added to set "alternate screen" mode. Normally, the 128 is added in, and you'll find 150 stored in this location. If you want to go to an alternate screen, remove the 128 element with POKE 36866,22 and the screen will now take its information from a new area. There are quite a few things you need to do if you wish to play with this – see "VIC: Alternate Screens" in *Home and Educational COMPUTING*, Fall, 1981.

• *Location 36867* (Hex 9003) is a busy one. In fact, it's always changing. Try typing ?PEEK(36867) several times and you'll see that you get different values – sometimes 46, sometimes 174. Let's ignore that extra 128 for the moment; we'll deal with it again when we describe the following location.

The basic value held in this location, normally 46, is the number of rows on the screen multiplied by two (23 rows, right?). You won't want to change this one.

There's one more thing hidden in this location, and it's important. If you add one to the value, the character generator will switch to "double character mode." This means that each character you type will occupy double the usual screen space.

This won't work automatically, however. If we want to draw characters that are twice as big, we must supply the VIC with "pictures" of the new characters; the old pictures won't do since they are not big enough to fill the new space. So prepare for a little confusion when you try this next experiment. Strange things will happen because we haven't built and connected up new character tables.

Type POKE 36867,47. The screen will go rather strange. Don't worry about it for the moment; just press the screen clear key (shifted, of course). The screen will clear, although the cursor looks rather odd. Not to worry, we'll forge right ahead.

The first character in the VIC's table of characters is the "@" symbol; the next is an "A," then a "B" and so on. Now: type the @ key. Instead of getting the first character, we get the first two, one above the other. Try typing the "A" and you'll get B-over-C, the next two characters in the list.

What's happening here? Each character you type is filling double its normal screen area. In doing so, it's grabbing twice as much information from the "character picture" table... and, since that table hasn't been changed, that means two characters. Since the VIC knows (or at least thinks) that the character pictures are twice as big, it reaches further into that table for each character that it needs.

When you decide to use this feature, you'll write your own character picture tables and everything will sort itself out. This feature is likely to be used most for high-resolution graphics. The elements of the character picture table will control individual dots, or pixels, on the screen.

You may bring your VIC back to normal by typing POKE 36867,46 but you'll need to type blind since the screen isn't much help. You might prefer to turn the computer off for a moment; when you turn on again, everything will be back as it was.

• *Location 36868* (Hex 9004). This location changes continuously. It's connected with the high-bit (128 value) in the previous location. In principle, it tells you precisely where on the screen the picture is being drawn at this instant. In practice, it's not much use to BASIC programmers – by the time

you read it, a different part of the screen will be active.

• *Location 36869* (Hex 9005). This is a very important address. It controls the location of two tables: the table where screen characters are held, and the table which holds the character pictures. Let's take them one at a time.

The screen table holds the five hundred or so characters that are displayed on the screen. It's quite a job to calculate the screen address: let's take a shot at it.

Take the contents of location 36869, divide by 16, and throw away the remainder. That should give you a number from 8 to 15. Subtract 8 and double it, giving an even number from 0 to 14. Now: if the contents of 36866 are 128 or greater, add one to get a value from 0 to 15. Multiply the result by 512. At this point you should have a value from 0 to 7680. That's where your screen table is located; it will normally be at location 7680, but it might move if you add extra memory.

That's quite a calculation; some of the things it implies deserve a separate article. For the moment, let's observe that the screen table address must always be in the range of 0 to 7680, and must be a multiple of 512. If you wish to set up your own screen table within this range, do the calculation in reverse: divide by 512, subracting 1 if odd, dividing by two, adding eight, and, finally, multiplying by 16. Whew! We can see that the "alternate screen" bit (128 value) in 36866 is really part of the much larger screen address.

The character picture table address is also defined in this location. We'd need to change this if we wanted to define our own characters, single or double. Of course, we'd also need to define character pictures for all characters we wished to print. The computation of the address is complex.

Take the contents of location 36869 and divide by 16. Now take the remainder – not the quotient – and, if it's greater than seven subtract eight. On the other hand, if the remainder is not greater than seven, add 32. By this time, you'll have an adjusted remainder which is either less than seven or between 32 and 39. Multiply this value by 1024 and you've found your character picture table address. It will be in the range of either zero to 7168 or 32768 (the normal value) to 39936, and will be an exact multiple of 1024. If you wish to set up your own character picture table, you'll usually want it to point to a RAM address in the range of zero to 7168. In such a case, you'd reverse the calculation: take the address, divide by 1024 and add eight and you're there.

Don't forget that the screen table address and the character picture address are packed together into this location. You'll need to set them both at the same time. By the way, the official name for the two tables are the "Video Matrix" and the "Character Cell table."

Feel free to play with this location; POKE values as you wish. But, unless you plan carefully, all you'll get is a crazy screen.

This was a tough one ... now we can try some easier locations.

• *Locations 36870 to 36871* (Hex 9006 and 9007). Here's your input from a Light Pen. No, a light pen isn't a ballpoint that weighs less than half an ounce – it isn't a pen at all. It's a device that plugs into your VIC that looks a little like a pen. Point it at the screen, and these locations will tell you where you are pointing.

A standard Atari light pen may be used. It is expected that Commodore will manufacture their own light pen soon. Many light pens have either a button or a spring-loaded switch in the tip which signals whenever the light pen operator wants attention. The switch is implemented in the VIC computer, but is not connected to the VIC chip (you'll find it mixed in with other things in location 37151).

You can read the X and Y positions of the light pen in locations 36870 and 36871 respectively. You won't read row and column values: the numbers will vary between zero and 255, and you'll need to do some calibration for the particular model of light pen that you have fitted.

Watch for "jitter" on these values. Even though the light pen doesn't move, the readings may jump about a little on successive readings. Depending on what you're doing, you may wish to use an averaging technique to make the readings smoother. Another method is called "hysteresis"; in simple terms, it means that a value is ignored unless it differs from previous readings by more than a given threshold amount.

• *Locations 36872 to 36873* (Hex 9008 and 9009). These are paddle input values. Two paddles, similar to Atari paddles, may be connected and their values will be read here. You may not be able to track the full range of rotation of the paddles.

Once again, watch for jitter on the input values here.

To keep the record straight, a joystick can also be connected to the VIC ... but the position of its buttons are not detected by the VIC chip. They arrive in other locations (37151 and 37152).

• *Locations 36874 to 36876* (Hex 900A to 900C). These are VIC's voices. Setting a value of 128 or higher into any of these locations produces sound; the value you POKE produces the pitch. By POKEing two or three locations, you can produce harmony. All voices are controlled by the sound level which is set at address 36878; try POKE

36878,15 before you play with the voices so that you'll get good volume. A value of less than 128 in any of the voice locations makes that voice silent.

It's interesting to note that the voice controlled by 36874 is the softest, and the voice at 36876 is the sharpest. So you'd use 36876 to carry the melody, and the two other voices as the sidemen.

• *Location 36877* (Hex 900D). This is similar to the music voices, except that it generates noise. A value of 128 or more produces noise. The higher the value, the higher the pitch of the noise (from growl to hiss). Once again, this is controlled by the sound level of 36878.

• *Location 36878* (Hex 900E). If the number in here is less than sixteen, it represents the sound amplitude (see the four previous locations). If it's sixteen or more, an extra factor is at work: multi-color.

Normally, each character position on the VIC contains only two colors: background and fore-ground. If we decide to use multi-color, we can add an extra two colors to each character: the border color plus one more that we may select. We select this color in the high part of location 36878. If you divide the contents of this location by 16, discarding the remainder, you'll get the designation of the "auxiliary color."

Interestingly, each character on VIC's screen is independently selected as two-color or multi-color, allowing us to have a mixed screen. This is done in the color nybble table which sets each character's color. Try the following: Clear the screen and type the letter A in the upper left-hand corner. Now go to a new line and type POKE 38400,8. You'll see that the letter A has suddenly turned weird and multi-colored, but the rest of the screen is unchanged. Notice that we did not POKE the VIC chip, but an entirely different memory location that is keyed to the one screen address. To do the job properly, you'll need to define your own character pictures.

• *Location 36879* (Hex 900F). The last location in the VIC chip, but a busy one. Let's break it down into its three elements.

Divide the contents of this location by 16, and note the result as "Screen Background Color." Now take the remainder; if it's eight or more, subtract and note: Foreground/Background = ON. The remaining value of zero to seven can be labelled "Frame Color."

The Frame Color is a favorite of mine; it's an easy signal to the user of some situation I want to tell him about without affecting the contents of the screen itself. If there's a danger, an error, or a game explosion, I can flip the border to red with POKE 36879,26 and later restore it with POKE 36879,27. Another example: Rather than typing a PLEASE WAIT message, I might walk the border through a range of colors so that the user can tell something is happening.

Screen Background color can be a very nice psychological support. If you set up a system so that accounts receivable can be done on one back-ground color and accounts payable on a different one, the user can be "keyed" to recognize that he's in the right program. It's a little like decorating each floor of a building in a different color so that people won't get the wrong one. Try combinations such as POKE 36879,155 and see how you like the effect.

Now for the Foreground/Background busi-ness. Normally (F/B = On) we know that we can type characters of many colors on a single color background. Sometimes, it can be very handy to do the opposite; in other words, we want to type single color characters on a background whose color may vary from character to character.

Try POKE 36879,19. Now clear the screen and type a few characters. Change color and type some more. Do you see what's happening? You are changing the color of the background, not the color of the characters themselves.

By playing around with these locations, you can discover potential that you never knew existed. Once you know it's there, you can then exploit it for your own special effects.

There's a rich variety of controls and informa-tion available in the VIC chip. You may not need to use them all .. but isn't it fascinating to play around?

Table 1: VIC 6560 Chip

$9000	Inter-lace	Left Margin (=5)		36864
$9001	Top Margin (=25)			36865
$9002	Scrn Ad bit 9	# Columns (=22)		36866
$9003	bit 0	# Rows (=23)	Double Char	36867
$9004	Input Raster Value: bits 8-1			36868
$9005	Screen Address bits 13-10	Character Address bits 13-10		36869
$9006	Light Pen Input	Horizontal		36870
$9007		Vertical		36871
$9008	Paddle Inputs	X		36872
$9009		Y		36873
$900A	ON	Voice 1	Frequency	36874
$900B	ON	Voice 2		36875
$900C	ON	Voice 3		36876
$900D	ON	Noise		36877
$900E	Multi-Color Mode (=0)	Sound Amplitude		36878
$900F	Screen Background Color	Foregnd /Backg	Frame Color	36879

©

When typing in this program, be especially careful when typing the numbers in the DATA statements and USR commands. A mistake could cause your machine to lock up, *that is, no longer respond to the external world, requiring a power-on reset.*

Using TextPlot For Animated Games

David Plotkin
Richmond, CA

If you're like me, the first thing you did when you bought your new Atari was run out to buy some games for it, probably with visions of multi-colored, arcade-style entertainment in mind. The computer store where I purchased my Atari also sells Apples. The wide assortment of exciting, machine language games available on the Apple and *not* on the Atari was a real disappointment. Time and time again I saw fascinating games which were not available to me. The recent release of many new Atari programs has somewhat alleviated this, but the problem still exists. To make things even more frustrating, many interesting games are not all that complex from a programming standpoint.

I decided to try my hand at programming these games myself. Having completed the book on how to program in BASIC, I charged ahead and wrote my first "Arcade-style" game, which I entitled "Space Rocks." It was a home-grown version of "Asteroids." The program had it all: graphics, sound, multiple missiles in flight at once, a fancy space ship, scoring, and music. It was also extremely slow. I had spent two weeks on it and each move took almost a minute. Ridiculous? Of course. I tried to speed it up by simplifying the graphics, but never did get it running very fast.

The next step was to try writing a program in a text mode. The Atari can manipulate text quickly, and so I met with a limited success. Using a custom designed character set also added to the text-mode games. Nevertheless, when there is more than one character to move, it can still be quite slow. I briefly considered learning machine language, but it's not something I'm eager to tackle.

The program "TextPlot" (**COMPUTE!**, November 1981, #18) is a first-rate gaming tool. As the author said, it allows you to use text and text characters in graphics modes. It also works with an alternate character set, as also mentioned briefly. But here's the kicker – since it draws the text character (and erases it also) using a machine language routine, it can be used to animate in high resolution graphics modes at machine language speeds. Thus, your character "A" ; redefined to a space ship or missile, literally zips across the screen, and five or ten "A's" can move across the screen without the frustrating BASIC's characteristic of "taking turns."

By drawing the non-moving portions of your picture in BASIC graphic mode, and the moving portion using TextPlot, you can write some colorful and challenging games. The program below demonstrates my own efforts in this regard, which I will tell you about shortly, but first some pointers:

1) Animation is done by drawing, erasing, and redrawing in a new position. The erasing can be done in two ways. You can call the USR command with the character ASCII code, but in the *background* color. Or you could call USR command with the ASCII code 32 (blank space) in *any* color. By looping and using a variable either in the color slot or in the ASCII code slot, drawing and erasing is easy. Increment the X and/or Y coordinates (such as MX1 and MY1 in the program) between erase and the redrawing, and the character moves smoothly across the screen. This incrementing, by the way, was done in BASIC (MX1 = MX + 1, etc.) and seems to be the limiting factor in how many characters can move across the screen at once without significant "taking turns."

2) It is possible to define a creature or object which consists of two or three refined characters which move together. It is best to increment the location of all three characters and then call the machine language routine to move them the most smoothly.

3) There is a large difference between vertical and horizontal resolution. Moving a character one space horizontally is equivalent to moving eight spaces vertically. Remember this when moving diagonally. Also, BASIC commands such as DRAWTO, PLOT, LOCATE etc. work on the graphic mode coordinate system. Thus, the horizontal location in mode seven can vary from zero to 159, but the X coordinate input to the USR call can vary only from zero to 19, normally. Therefore X coordinate = horizontal location/eight. The vertical resolution is the same as the Y coordinate.

Note that, in the program, I have varied the X coordinate from 60 to 79 instead of zero to 19. What this does is move the character down one pixel for each multiple of twenty (60 to 79 moves the character down three pixels from where it would be a zero to 19). A character moving horizontally will pass across the screen lower and lower

CALL TOLL FREE 800 424-2738

Programs for ATARI 400/800 Unless Otherwise Noted

the PROGRAM STORE

Save $$ on Popular Programs for Atari

LIMITED TIME OFFER

Now through May 15, take advantage of our price reduction on these popular programs!

CAVERNS OF MARS

From APX
An award-winning arcade game that will challenge your skill for many, many plays. You command the fighter Helicon VII on a mission to invade the Martian high command. The headquarters are located deep in the caverns, so you must descend through radar, rockets, laser gates, and space mines to reach your destination. Once there, you activate the fusion bomb and try to get out before it detonates. Good luck!

16K tape...~~$29.95~~
24K disk...~~$29.95~~ Save 20%!

NOW THRU MAY 15 ONLY: pay only $23.96

EVEREST EXPLORER

By Godwin & Knowlton from Acorn
This is a game of logistics in which you try to lead a team up Mount Everest. You must outfit the expeditionary party and buy supplies within your assigned budget. Then you must manage the climbers' and sherpas' scheduling, allocate supplies, and try to keep pace with the contingencies. If your skill, the weather, and luck hold out, you'll make it. But remember, you'll also have to get back down safely.

40K disk with "Save Game" feature...~~$19.95~~ Save 20%!

NOW THRU MAY 15 ONLY: pay only $15.96!

SHADOW HAWK ONE

From Horizon
Hi-resolution excitement in a space game that provides action in 3-dimensions, 3-axis of rotation! As commander of SHADOW HAWK ONE, it's up to you to destroy the tyrannical Galactic Empire. While you have the most advanced space fighter available, it's you alone against the many enemy craft. A taxing game that combines action with strategy, dexterity with planning, and reflexes with skill. SHADOW HAWK ONE is a fast-paced simulation that will rivet you to your seat!

48K disk...~~$49.95~~ Save 20%!

NOW THRU MAY 15 ONLY: pay only $39.96!

Mouskattack

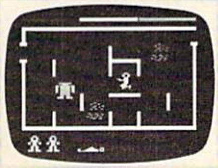
By John Harris from On-Line

A new "eat-em-up" arcade game from the author of JAWBREAKER! This time you're a plumber, trying desparately to finish an installation in Rat Alley's mazelike corridors. Armed only with two cats, two traps and considerable courage, you try to dodge the man-eating rats while completing your job. If you're really adventuresome, you can try the two-player option where an extra player doesn't mean more help: it means more competition and danger!
32K disk...$34.95

Stocks & Bonds

From Avalon Hill
Enjoy the thrills and challenge of stocks and bonds trading without the risk. This simulation provides all the action, complete with ticker-tape style prices, a newswire service, split and bankrupcy notices, and more. Can be played alone or in a multiplayer game. The complete instruction book lists information about each security on the game's "Big Board."
32K tape...$19.95
Deluxe Disk Edition: Supports simultaneous play by up to 4 players. 40K disk...$25

K-RAZY SHOOT OUT



From K-Byte
Get this collector's item while you can: It's the first non-Atari game in a ROM cartridge. Just slip the cartridge into the slot and you can instantly be playing this action-packed arcade game. The object is to move your Space Commander through the Alien Control Sectors, shoot the Alien Droids and escape from the sectors. The sectors are a maze of passages and barriers that can be used for protection, but don't let your commander touch them -- they're electrified!

ROM Cartridge...$49.95

KAYOS SPACE WAR GAME

From Computer Magic
A machine language space game with plenty of action, graphics and sound. You have five base stations with which to fight off the invasion of Kayotians. They arrive in large Mother Ships which are a launching base for their attack ships. Knock out a mother ship and it simultaneously launches a 1000 point bonus ship and a destructive bomb. KAYOS has options to allow easier play for beginning and intermediate-level players.

8K tape or disk...$34.95

CHALLENGEM 1

Anthony Targonski from Compu-Things
The CHALLENGEM series of simulations are fantasy situation games. Using your ability and reasoning powers, you must manipulate the available data to accomplish the task at hand.
ATTACK OF THE YETI, 16K tape, $14.95
ROMAN CONQUEST, 16K tape, $14.95
DISASTER AT SEALAB, 16K tape, $14.95
All Three on Disk...$39.95

THE NEXT STEP

By Bob Stewart from On-Line
A powerful utility for creating graphics on the Atari. Using your joystick, you can make freehand designs in any of Atari's 14 graphics modes (including the undocumented multicolor modes). Your designs are automatically translated into basic code that you can incorporate into your programs. The package includes a graphics utility, a color utility, several programing examples written in BASIC, plus a comprehensive 36 page instruction booklet.
32K disk...$39.95

Master Memory Map

From Santa Cruz Software
If you are serious about programming the Atari, this booklet could become one of your most important tools. A highly detailed map of the Atari's memory, it details thousands of locations and routines. Using this booklet makes easier the use of display list, player/missile, and interrupt graphics. The "Miscellaneous Notes" section contains a wealth of knowledge picked up by Santa Cruz in their explorations of the Atari. Also included are notes on the new GTIA graphics chip.
$5.95

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. D.C. residents, add 6% tax. Charge card customers: include all embossed information on card.

THE PROGRAM STORE
4200 Wisconsin Avenue NW, Dept. U04 Box 9609
Washington, D.C. 20016

at higher values of the X coordinate without changing the Y coordinate. This invalidates the relationships shown above between coordinates and screen position, which only work if the X coordinate is between zero and 19.

4) A LOCATE statement meant to find or detect one of the generated text characters does not care what the character is, only the color of the character. This is because the text character is just a series of pixels set to a particular color.

5) The alternate character set is located in an area of RAM protected by POKEing a lower number of pages into location 106, which stores the amount of pages (multiply by 256 to get bytes) available in memory. This is a fairly common technique of protecting memory, since the computer doesn't know about the memory above location 106 (see line 3200 in the program) and, thus, doesn't use it.

In the original version of the character generator, a step-back of five pages (1280 bytes) was used. The character set is four pages (1K) long, plus one extra. This works fine in Graphics mode zero, but does not work for this program. I found that the minimum step-back is 16 pages (4K), although any multiple of 4K (32 pages or 48 pages) will work. Intermediate values led to part of the screen being blank or runny dots and lines being displayed. I think this has to do with the display list not crossing a 4K boundary (the display list in Gr. 7 is right around 4K long, so it would have to be located *on* a 4K boundary) but I'm not sure. Perhaps a more advanced programmer could shed some light on this. A final point on this: after every GRAPHIC command, you need to include a POKE 756,PEEK(106) + 1 to point the Character Base address to the redefined character set, since the GRAPHIC command resets the pointer to the ROM character set.

Rules Of The Game

Now to the program. You are chief gunnery officer of the Space Fortress Reliable, located at the outermost fringes of the Galactic Empire. Although the fortress is protected by shields, there are four "channels" through the shields to allow for supply ships and transportation of personnel. Since attacking vessels can also make use of these channels, a big laser is mounted to fire down each of the channels.

The channels are located directly above, below, left, and right of the fortress. Their width is such that only one ship at a time can attack from any direction. The laser is aimed in the appropriate direction by pushing the joystick in that direction. Once the laser is aimed, it fires automatically.

As the attack progresses, however, and energy is used up, the shields begin to withdraw towards the fortress to maintain integrity. The enemy ships can come out of hyperspace and begin the attack through the channels closer to the fortress, so you have less time to fire on them. Watch out especially for the ships to the left and right which, although they start farther away than the ships above and below, move eight times as fast. Good luck, and good hunting.

Program Line No.	Description
1-10	Go to the subroutines for redefining the character set and initializing TextPlot.
20	Initialize graphics, set character base address to redefined character set.
30	Initialize variables.
40-100	Draw the fortress and background.
110-120	Print "SCORE 000" on the screen.
130-170	Erase last gun position.
180-220	Read current joystick position.
230-280	Aiming and Firing sequence. The gun is drawn in the new position, and the laser is fired. If the ship is hit, then it explodes.
290-310	Updates the score on the screen, dight by digit. Jumps to the end of the game on high score.
320-350	If a ship was destroyed, then uses the random number generator to decide whether a new ship is to be launched. The starting position of the new ship is moved closer to the fortress as the score increases.
360-400	Moves each ship toward the fortress, if the fortress is hit by a ship then jumps to the end of game routine.
500-620	End of game routine when fortress is destroyed.
700-710	End of game routine on winning game.
20000-20430	Subroutine for Textplot.
32000-32160	Subroutine for redefining character set.

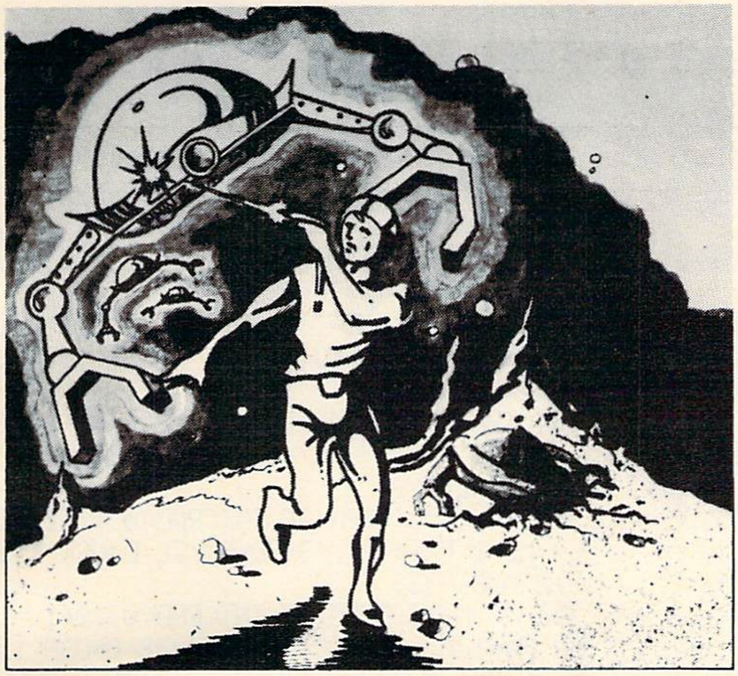
Variables

SC = Score J = joystick position

J1 = 1,2,3,4 depending on joystick position
MX1 to MX4 = X coordinate of enemy ships
MY1 to MY4 = Y coordinate of enemy ships
M1 to M4 = status of enemy ships; = 0 when ship is blown up = 1 when ship is intact
Starx,Stary = X and Y coordinates of stars
ML = memory location
START = byte address of RAMTOP
Z,Y,STAR,N,W,I = loop variables.

If you do not want to punch the program in, I will be happy to generate a copy for you on tape (sorry, I don't have a disk drive yet, but there will be no protects on the tape copy, so you can easily transfer to disc). Send a cassette with a self addressed stamped (requires 40 cents postage) envelope and a check for $3 to:

David Plotkin
5720 Alameda Ave.
Richmond, CA 94804



```
1 GOSUB 32000:CLR
10 GOSUB 20000
20 GRAPHICS 7+16:POKE 756,PEEK(106)+1
30 SETCOLOR 2,3,4:SC=0:J1=1:MX1=0:MY1=0:
MX2=0:MY2=0:MX3=0:MY3=0:MX4=0:MY4=0:M1=0
:M2=0:M3=0:M4=0
40 COLOR 1:FOR Y=35 TO 45:PLOT 72,Y:DRAW
TO 95,Y:NEXT Y
50 PLOT 72,35:DRAWTO 69,32:PLOT 73,35:DR
AWTO 69,32:PLOT 72,36:DRAWTO 69,32
60 PLOT 72,45:DRAWTO 69,48:PLOT 73,45:DR
AWTO 69,48:PLOT 72,44:DRAWTO 69,48
70 PLOT 95,35:DRAWTO 98,32:PLOT 94,35:DR
AWTO 98,32:PLOT 95,36:DRAWTO 98,32
80 PLOT 95,45:DRAWTO 98,48:PLOT 94,45:DR
AWTO 98,48:PLOT 95,44:DRAWTO 98,48
90 FOR STAR=1 TO 80:STARX=RND(0)*158+1:S
TARY=RND(0)*94+1:PLOT STARX,STARY:NEXT S
TAR
100 COLOR 0:FOR X=73 TO 94 STEP 2:PLOT X
,40:NEXT X
110 D=USR(1536,83,3,0,0):D=USR(1536,67,3
,1,0):D=USR(1536,79,3,2,0)
120 D=USR(1536,82,3,3,0):D=USR(1536,69,3
,4,0):D=USR(1536,48,3,1,8):D=USR(1536,48
,3,2,8):D=USR(1536,48,3,3,8)
130 ON J1 GOTO 140,150,160,170
140 D=USR(1536,32,1,70,24):GOTO 180
150 D=USR(1536,32,1,72,34):GOTO 180
160 D=USR(1536,32,1,70,43):GOTO 180
170 D=USR(1536,32,1,68,34)
180 J=STICK(0):IF J=15 THEN GOTO 290
190 IF J=10 OR J=14 OR J=6 THEN J1=1:D=U
SR(1536,16,1,70,24):GOTO 230
200 IF J=7 THEN J1=2:D=USR(1536,17,1,72,
34):GOTO 230
210 IF J=5 OR J=13 OR J=9 THEN J1=3:D=US
R(1536,18,1,70,43):GOTO 230
220 IF J=11 THEN J1=4:D=USR(1536,19,1,68
,34)
230 COLOR 1:SOUND 0,25,10,8:SOUND 1,28,1
0,8:ON J1 GOTO 250,260,270,280
250 PLOT 84,27:DRAWTO 84,0:COLOR 0:PLOT
84,27:DRAWTO 84,0:IF M1=1 THEN M1=0:D=US
R(1536,15,3,MX1,MY1):SC=SC+2
255 GOTO 290
260 PLOT 104,40:DRAWTO 159,40:COLOR 0:PL
OT 104,40:DRAWTO 159,40:IF M2=1 THEN M2=
0:D=USR(1536,15,3,MX2,MY2):SC=SC+2
265 GOTO 290
270 PLOT 84,54:DRAWTO 84,95:COLOR 0:PLOT
 84,54:DRAWTO 84,95:IF M3=1 THEN M3=0:D=
USR(1536,15,3,MX3,MY3):SC=SC+2
275 GOTO 290
280 PLOT 63,40:DRAWTO 0,40:COLOR 0:PLOT
63,40:DRAWTO 0,40:IF M4=1 THEN M4=0:D=US
R(1536,15,3,MX4,MY4):SC=SC+2
290 SOUND 0,0,0,0:SOUND 1,0,0,0:SOUND 3,
0,0,0:IF SC>999 THEN GOTO 700
300 V1=INT(SC/100):V2=INT(SC/10-V1*10):V
3=SC-V1*100-V2*10:V1=V1+48:V2=V2+48:V3=V
3+48
310 D=USR(1536,V1,3,1,8):D=USR(1536,V2,3
,2,8):D=USR(1536,V3,3,3,8)
320 IF M1=0 THEN IF INT(RND(0)*2+1)=1 TH
EN M1=1:MX1=70:MY1=SC/75:D=USR(1536,20,2
,MX1,MY1)
330 IF M2=0 THEN IF INT(RND(0)*2+1)=1 TH
EN M2=1:MX2=79-SC/400:MY2=33:D=USR(1536,
21,2,MX2,MY2)
340 IF M3=0 THEN IF INT(RND(0)*2+1)=1 TH
EN M3=1:MX3=70:MY3=70-SC/75:D=USR(1536,2
2,2,MX3,MY3)
350 IF M4=0 THEN IF INT(RND(0)*2+1)=1 TH
EN M4=1:MX4=60+SC/400:MY4=32:D=USR(1536,
23,2,MX4,MY4)
360 IF M1=1 THEN D=USR(1536,20,0,MX1,MY1
):MY1=MY1+1:D=USR(1536,20,2,MX1,MY1):IF
MY1>=24 THEN GOTO 500
370 IF M2=1 THEN D=USR(1536,21,0,MX2,MY2
):MX2=MX2-1:D=USR(1536,21,2,MX2,MY2):IF
MX2<=72 THEN GOTO 500
380 IF M3=1 THEN D=USR(1536,22,0,MX3,MY3
):MY3=MY3-1:D=USR(1536,22,2,MX3,MY3):IF
MY3<=43 THEN GOTO 500
390 IF M4=1 THEN D=USR(1536,23,0,MX4,MY4
):MX4=MX4+1:D=USR(1536,23,2,MX4,MY4):IF
MX4>=68 THEN GOTO 500
400 GOTO 130
500 SOUND 0,50,8,8:SOUND 1,100,8,8:SOUND
 2,200,8,8:SOUND 3,5,8,8
510 D=USR(1536,15,3,68,34):D=USR(1536,15
,3,70,43):D=USR(1536,15,3,72,34):D=USR(1
536,15,3,70,24)
520 D=USR(1536,15,3,69,36):D=USR(1536,15
,3,69,40):D=USR(1536,15,3,70,30):D=USR(1
536,15,3,71,27)
530 FOR N=0 TO 3:SOUND N,0,0,0:NEXT N
540 FOR N=1 TO 150:NEXT N
550 FOR N=0 TO 3:SOUND N,N*80+5,8,8:NEXT
 N
560 COLOR 3:PLOT 84,40:DRAWTO 84,20:DRAW
TO 84,60:PLOT 84,40:DRAWTO 114,40:DRAWTO
 54,40:PLOT 84,40:DRAWTO 114,20
570 PLOT 84,40:DRAWTO 114,60:PLOT 84,40:
DRAWTO 54,60:PLOT 84,40:DRAWTO 54,20
580 FOR W=0 TO 15:FOR W1=1 TO 100:SETCOL
OR 2,W,5:NEXT W1:NEXT W
590 FOR N=0 TO 3:SOUND N,0,0,0:NEXT N
600 GRAPHICS 2+16:? #6;"GAME OVER..FINAL
":? #6;"SCORE ";SC:? #6;"TO PLAY AGAIN":
? #6;"PRESS TRIGGER"
```

```
610 IF STRIG(0)=1 THEN GOTO 610
620 GOTO 20
700 GRAPHICS 2+16:? #6;"GOOD GAME!!!":?
#6;"!YOU WON!!!!":? #6;"YOUR SPACE FORTR
ESS":? #6;"SURVIVED"
710 ? #6;"TO PLAY AGAIN":? #6;"PRESS !TR
IGGER!":GOTO 610
19999 END
20000 ML=1536:FOR I=0 TO 252:READ A:POKE
 ML+I,A:NEXT I:RETURN
20010 DATA 104,240,10,201,4,240
20020 DATA 11,170,104,104,202,208
20030 DATA 251,169,253,76,164,246
20040 DATA 104,133,195,104,201,128
20050 DATA 144,4,41,127,198,195
20060 DATA 170,141,250,6,224,96
20070 DATA 176,15,169,64,224,32
20080 DATA 144,2,169,224,24,109
20090 DATA 250,6,141,250,6,104
20100 DATA 104,141,251,6,104,104
20110 DATA 141,252,6,14,252,6
20120 DATA 104,104,141,253,6,133
20130 DATA 186,166,87,169,10,224
20140 DATA 3,240,8,169,20,224
20150 DATA 5,240,2,169,40,133
20160 DATA 207,133,187,165,88,133
20170 DATA 203,165,89,133,204,32
20180 DATA 228,6,24,173,252,6
20190 DATA 101,203,133,203,144,2
20200 DATA 230,204,24,165,203,101
20210 DATA 212,133,203,165,204,101
20220 DATA 213,133,204,173,250,6
20230 DATA 133,187,169,8,133,186
20240 DATA 32,228,6,165,212,133
20250 DATA 205,173,244,2,101,213
20260 DATA 133,206,160,0,162,8
20270 DATA 169,0,133,208,133,209
20280 DATA 177,205,69,195,72,104
20290 DATA 10,72,144,8,24,173
20300 DATA 251,6,5,208,133,208
20310 DATA 224,1,240,8,6,208
20320 DATA 38,209,6,208,38,209
20330 DATA 202,208,228,104,152,72
20340 DATA 160,0,165,209,145,203
20350 DATA 200,165,208,145,203,104
20360 DATA 168,24,165,203,101,207
20370 DATA 133,203,144,2,230,204
20380 DATA 200,192,8,208,183,96
20390 DATA 169,0,133,212,162,8
20400 DATA 70,186,144,3,24,101
20410 DATA 187,106,102,212,202,208
20420 DATA 243,133,213,96,0,1
20430 DATA 28
32000 POKE 106,PEEK(106)-16:GRAPHICS 0:S
TART=(PEEK(106)+1)*256:POKE 756,START/25
6:POKE 752,1
32020 FOR Z=0 TO 1023:POKE START+Z,PEEK(
57344+Z):SETCOLOR 2,0,RND(0)*255+1:NEXT
Z:RESTORE 32100
32030 READ X:IF X=-1 THEN RESTORE :RETUR
N
32040 FOR Y=0 TO 7:READ Z:POKE X+Y+START
,Z:NEXT Y:GOTO 32030
32100 DATA 632,145,82,44,222,57,52,74,13
7
32101 DATA 640,24,24,24,60,126,126,60,25
5
32102 DATA 648,128,176,248,255,255,248,1
76,128
32103 DATA 656,255,60,126,126,60,24,24,2
4
32104 DATA 664,1,13,31,255,255,31,13,1
32105 DATA 672,231,231,126,60,24,24,24,0

32106 DATA 680,3,7,15,252,252,15,7,3
32107 DATA 688,24,24,24,24,60,126,231,23
1
32108 DATA 696,192,224,240,63,63,240,224
,192
32109 DATA -1
```

©

Accurate Timing In Atari BASIC

John Navas, II
San Mateo, CA

Trying to accurately control time intervals in Atari BASIC is a frustrating experience. A FOR...NEXT loop is the method most often recommended to create a simple time delay. An example of this approach is:

FOR N = 1 TO 1000:NEXT N

By adjusting the TO value or expression to a smaller or larger number than 1000, the length of the time delay can be made shorter or longer. Unfortunately, however, this method has several drawbacks:

1. The exact time delay desired can only be found through a process of trial and error.
2. The time delay for a given TO value or expression will vary significantly, depending on where it is located in a program (shorter near the beginning and longer near the end).
3. The length of the time delay is significantly affected by the display mode (a given FOR ...NEXT loop takes about 30% less time in GRAPHICS 3 than in GRAPHICS 0).
4. Accurately timing more than one interval at the same time is very difficult.
5. The BASIC program can't do anything else during the time delay interval.

These BASIC problems can be easily overcome in machine language. Although machine language programs can use a similar loop counter approach, they also have ready access to a number of superior alternatives provided by ATARI's unique hardware and Operating System. These alternatives include the five system timers, the Real-Time CLOcK (RTCLOK), and the POKEY hardware timers (normally used to generate tones with the SOUND statement). Happily, there are simple ways to gain access to some of these facilities in Atari BASIC. This article describes how to use the system timers.

First, a short note on how system timers 3, 4 and 5 work. (Although there are five timers in all, only these three are readily useable in BASIC.) Each timer consists of a two-byte counter, CDTMV3-CDTMV5, and a one-byte flag, CDTMF3-CDTMF5. The actual hardware addresses are given in Program 1. To use a Timer, its flag must first be set to any non-zero value, and then its counter must be set to the value of the desired time interval. The operating system will subtract one from the counter every 1/60 second during the start of each new TV-picture frame (VBLANK) until the counter reaches zero. At zero, the operating system changes the flag back to zero and stops counting. All the BASIC program needs to do once the timer is running is to PEEK at the flag periodically to see if the time interval has run out. Note that all three timers may be used at the same time, if desired.

The timer counters are typical 6502 two-byte binary numbers, with the least significant byte at the lower hardware address. This means that each count in the lower address byte is worth 1/60 second up to a maximum of 255 counts for 4.25 seconds; each count in the higher address byte is worth 256 counts in the lower address byte, or just over 4.25 seconds. Only the lower address byte need be set if the desired time intervals do not exceed 4.25 seconds. Otherwise, Program 1 gives a simple method of calculating the correct values for each byte given a desired time interval in seconds.

Setting the timers is a little tricky. There are two or three bytes to set and the Operating System must be prevented from starting the countdown until all of them are set. Fortunately, there is an easy way: setting the CRITIC flag to a non-zero value suspends a number of Operating System processes including system timer counting. Program 1 shows the recommended procedure to set system timer 4 using the CRITIC flag.

Important: Do not leave CRITIC set to a non-zero value any longer than necessary!

Although the system timers are extremely useful, they do have a few limitations:

1. The maximum time interval which can be set directly is a little more than 18 minutes (65535 counts of 1/60 second). Of course, longer intervals could be controlled by counting multiple runs of a timer.
2. Very small timer counts will be imprecise because the Operating System is synchronized to the TV display and not to the BASIC program.
3. When one timer is being set, the CRITIC flag stops the other timers. In multiple timer applications a small amount of time may be "lost".
4. The Operating System uses System Timer 3 to OPEN the 410 program recorder. Any

prior value in the timer will be lost.

5. System timers 3, 4 and 5 are stopped during I/O operations because the Operating System sets the CRITIC flag. Significant amounts of I/O will cause time to be "lost".

For many applications, these limitations do not present any real problems. The system timers are powerful tools which are almost as easy to use as relatively crude FOR...NEXT loops; give them a try! For those other applications with different timing needs, there are the other timing alternatives mentioned above.

References:

ATARI Personal Computer System OPERATING SYSTEM User's Manual, Atari Inc., Copyright 1980, Chapter 6.

Stewart, Ed, "Unleash the P ower of Your Atari CPU", **COMPUTE!**, April 1981 #11, p. 102.

```
5 GRAPHICS 3
10 CDTMV3=540:CDTMF3=554:REM TIMER 3
15 CDTMV4=542:CDTMF4=556:REM TIMER 4
20 CDTMV5=544:CDTMF5=558:REM TIMER 5
25 CRITIC=66:REM CRITIC FLAG
30 NONZERO=1:ZERO=0
35 TIME=30.5:REM TIME EXAMPLE (SEC.)
40 TIME=TIME*60:REM CNVRT TO 1/60 CNT
45 HI=INT(TIME/256):REM TIMER HI BYTE
50 LO=TIME-(HI*256):REM TIMER LO BYTE
55 POKE CRITIC,NONZERO:REM STOP TIMER
60 POKE CDTMF4,NONZERO:REM SET FLAG
65 POKE CDTMV4,LO:REM SET LO BYTE
70 POKE CDTMV4+1,HI:REM SET HI BYTE
75 POKE CRITIC,ZERO:REM START TIMER
80 PRINT "TIMER STARTED"
85 COLOR INT(RND(0)*3)+1:REM EXAMPLE
86 PLOT RND(0)*39,RND(0)*19:REM EXAM
87 SOUND 0,RND(0)*128,10,8:REM EXAMPL
90 IF PEEK(CDTMF4) THEN 85:REM CHECK
95 PRINT "TIME'S UP!"
```

©

Carl Moser's MAE Assembler for the Apple and PET has proven popular with machine language programmers. Here, Dr. Lindner demonstrates how you can add your own, custom pseudo op's.

Extending MAE

Dr. Harald Lindner
Krefeld, West Germany

Carl Moser's excellent MAE macroassembler and text editor for the PET and the Apple is a very powerful program, but it can still be improved. I should like to describe how two new pseudo operation codes ".SL" and ".SH" can be added. These pseudo op's .SL "store low byte of a two byte address" and .SH "store high byte of a two byte address" may be used to generate jump-tables (e.g. the TOOLKIT, Commodore's BASIC-interpreter, and machine language monitor).

MAE's pseudo op's are contained in the table from 57C3 to 5834 (all addresses in hexadecimal). Each entry consists of four bytes. The first two bytes contain the ASCII-codes of the pseudo opcode name, and the last two bytes contain the address of the corresponding subroutine in the usual 6502-format low/high.

Fortunately, eight spare zero bytes (582B-5833) at the end of the table leave room for the addition of two new pseudo opcodes.

The Simple Solution

MAE contains two subroutines, Low at 5E39 and High at 5E4A, to evaluate the expressions "#L," and "#H," as in LDA #L,$B312 and LDY #H,$B312. The simple solution is to replace the eight zeros at 582B-5833 by:

```
582B  53  4C  39  5E  ;SI LOW
582F  53  48  4A  5E  ;SH HIGH
```

MAE will now handle the new pseudo op's .SL and .SH as expected. The only remaining problem is the relocating loader, which doesn't yet like these pseudo op's.

A Better Solution

Let us quickly consider the relocating information in MAE files. Assembler source files are program files, whereas MAE's OUTPUT command generates sequential files. The information for the relocating loader is contained in bytes whose low nibble is F (15 decimal). These bytes do not constitute valid 6502 instructions. After an ordinary machine language instruction such as LDA 4A or STX 6502 the bytes 0F to 7F have the following meaning:

0F: fixed address, do not alter.
1F: low byte of an address, the high byte follows.
2F: high byte of an address, the low byte follows.
3F: a byte value follows, do not alter.
4F: an address (2 bytes) follows, alter unless 0F follows.
5F: pseudo op .RS (resolve info set).
6F: pseudo op .RC (resolve info clear).
7F: pseudo op .DS (define storage).

Our new pseudo ops .SL and .SH will have to provide the following informations to the relocating loader:

a byte value follows (via 3F),
this is the low or high, respectively,
part of an address (via 1F, 2F).

This is achieved by the following coding:

```
75BD  SL  JSR  6347
          LDA  49
          JSR  5B67
          JMP  5E41
75C8  SH  JSR  6347
          LDA  4A
          JSR  5B67
          JMP  5E52
```

This routine may be located conveniently at 75BD-75D2 which area is not occupied by MAE (at least not the PET version). MAE's pseudo op table is to be altered as follows:

```
582B  53  4C  BD  75
582F  53  48  C8  75
```

Finally, the relocating loader must learn to consider the directives 1F and 2F after a byte value (via 3F). This is easily done by replacing JMP 0584 with JMP 0639 at location 059A of the relocating loader (1979 version). Your MAE assembler will now handle the new pseudo op's .SL and .SH perfectly. ©

Last month, in Part I, a method of communicating with the disk drive in machine language was outlined. Here, the techniques are brought together into one program (a mixture of machine language and BASIC). The BASIC part is:

```
100 OPEN 6,8,15
110 PRINT #6, "I0"
120 OPEN 2,8,3, "#"
125 SYS 1200
190 CLOSE 2 : CLOSE 6
```

The article concludes with a program which will perform quality checks on disks and can also recover scratched disks. The Disk Checkout program will work on all disk units. For the new single-drive disk units, type S when you are asked for the Drive #.

Part II:

Disk Checkout For 2040, 4040, And 8050 Disks

Jim Butterfield
Toronto, Canada

Here's a listing of the complete machine language program:

```
  ; select command channel
04B0 A2 06                LDX #6
04B2 20 C9 FF             JSR SWITCHOUT
                              PUT
  ; send U1 message
04B5 A2 00                LDX #0
04B7 BD 0A 05 ULOOP       LDA UMESSAGE,
                              X
04BA F0 06                BEQ QUIT1
04BC 20 D2 FF             JSR OUTPUT
04BF E8                   INX
04C0 D0 F5                BNE ULOOP
04C2 20 CC FF QUIT1       JSR RESTOREIO
  ; send B-P message
04C5 A2 06                LDX #6
04C7 20 C9 FF             JSR SWITCHOUT
                              PUT
04CA A2 00                LDX #0
04CC BD 19 05 BLOOP       LDA BMESSAGE,
                              X
04CF F0 06                BEQ QUIT2
04D1 20 D2 FF             JSR OUTPUT
04D4 E8                   INX
04D5 D0 F5                BNE BLOOP
04D7 20 CC FF QUIT2       JSR RESTOREIO
  ; GET VALUES
04DA A2 02                LDX #2
04DC 20 C6 FF             JSR SWITCHINP
                              UT
04DF A2 00                LDX #0
04E1 20 E4 FF READ        JSR GET
  ; convert to hexadecimal
04E4 48                   PHA
04E5 4A                   LSR A
04E6 4A                   LSR A
04E7 4A                   LSR A
04E8 4A                   LSR A
04E9 20 FD 04             JSR HXPRNT
04EC 68                   PLA
04ED 20 FD 04             JSR HXPRNT
04F0 A9 20                LDA #$20
04F2 20 FF D2             JSR OUTPUT
04F5 E8                   INX
04F6 E0 08                CPX #8
04F8 90 E7                BCC READ
04FA 4C CC FF             JMP RESTOREIO
  ; hex conversion subroutine
04FD 29 0F       HXPPRNT  AND #$0F
04FF 09 30                ORA #$30
0501 C9 3A                CMP #$3A
0503 90 02                BCC NUM
0505 69 06                ADC #$06
0507 4C D2 FF NUM         JMP OUTPUT
  ; canned messages: U1 and B-P
050A 55 31 3a 33 20 30 20 31 38 20
     30 30 0d 00
0519 42 2d 50 3a 33 20 31 0d 00
```

Putting It Together

Type in the BASIC program first. Now, call the Machine Language Monitor with SYS4 and display

the desired part of memory with .M 04B0 0520 – you'll see whatever values are currently in memory. Ignore them; move the cursor back and type over so that you get:

```
04B0  A2 06 20 C9 FF A2 00 BD
04B8  0A 05 F0 06 20 D2 FF E8
04C0  D0 F5 20 CC FF A2 06 20
04C8  C9 FF A2 00 BD 19 05 F0
04D0  06 20 D2 FF E8 D0 F5 20
04D8  CC FF A2 02 20 C6 FF A2
04E0  00 20 E4 FF 48 4A 4A 4A
04E8  4A 20 FD 04 68 20 FD 04
04F0  A9 20 20 D2 FF E8 E0 08
04F8  90 E7 4C CC FF 29 0F 09
0500  30 C9 3A 90 02 69 06 4C
0508  D2 FF 55 31 3A 33 20 30
0510  20 31 38 20 30 30 0D 00
0518  00 42 2D 50 3A 33 20 31
0520  0D 00 00 00 00 00 00 00
```

Remember to press RETURN on each line to enter the values. One last change: display .M 0028 0028 and change what you see to:

```
.: 0028 01 03 22 05 22 05 22 05
```

You may now return to BASIC (with .X) and SAVE the program if you wish. If you have carefully proofread the code, type RUN and watch the program do the same thing as in BASIC. The values are output in hexadecimal rather than decimal.

You may have noticed that we have done a lot of work to produce a machine language program that is bigger than the BASIC one we wrote in the first place. For our example, speed isn't much of a factor.

In doing so, however, we've established that you can indeed work a disk from machine language. And sometimes that can be very useful indeed.

Disk Test

Before I take a diskette "on the road," there are two things that I often want assurance about. First, are all the files good? Secondly, are the empty blocks in good condition?

This program tests the disk for these two properties, and adds a third: it will reclaim one scratched file if desired – and if the file is still intact on disk.

The program is constructed to work on 2040, 4040 and 8050 disks. On 2040's, however, it can only usefully perform the first of the three activities.

Checking Files

There's more to checking a file than seeing if it is there. Some of the questions that need to be asked are:

–Are all the blocks of the file OK?
–Are all blocks allocated?
–Is there conflict with any other file?
–Is the block count correct?

This program checks all of the above. In doing so, it has turned up a minor bug in the disk system: files joined with CONCAT and APPEND are likely to have the wrong block count. This doesn't hurt anything, but gives you misinformation on your directory listing.

The files are shown on the screen as they are checked. If there is an error, the program will stop with an error notice. See the note below on errors.

Checking Blocks

If you plan to write on a disk, and aren't sure everything is in good order, option two, checking blocks, is convenient.

The program reads all blocks and ensures that they are in sound working order. 2040 type disks (DOS1) can't be checked in this way; when a 2040 diskette is new-ed, all blocks are not written. This program checks by reading, not by writing; it can't do a valid job if some blocks have never been written.

The blocks and sectors are shown on the screen as they are checked. If there is an error, the program will stop with an error notice. See the note below on errors.

Un-SCRATCHing

The program searches the directory for the names of scratched files and asks you whether you want to recover any file. When you say *yes*, you will be asked to identify the file type – this information was lost when the file was scratched. Then, a number of checks are very carefully made:

–Are all blocks good?
–Are all blocks free?
–Is the block count correct?

If the file passes all the above tests, the unscratching takes place, and the disk is asked to perform a VERIFY/VALIDATE/COLLECT which re-allocates the blocks.

Only one file can be reclaimed during a run. Multiple runs can unscratch many files, one at a time. The reason for this is related to the safety interlocks. When you reclaim a file, its blocks will be re-allocated. If you attempt to reclaim a second file and somehow its blocks conflict with the first file, this will be spotted: an allocated block will be found and the unscratch will not take place. If the program had been constructed so as to reclaim a whole series of files in one shot, this level of protection would have been lost.

The Program

The program is written entirely in BASIC, so a user can modify it to his particular needs. Except for a very small part of un-scratch, the program

does not write to the disk; it only reads. Be aware, however, that closing a direct channel will force a BAM write to take place; this makes it desirable for a program to run through to completion rather than be stopped halfway through. See the error note below.

The Block Availability Map (BAM) is printed, and this may be an interesting thing to view for users who are not aware of the disk's organization. The centre track (track 18 or 39) is reserved for the directory, and file space is allocated close to this centre track where possible; this minimizes drive head movement. When a file is written on a track, it is not written to consecutive sectors, but "hops around" in order to optimize speed.

The program decides what sort of disk it has based on information supplied by the diskette itself. Thus, a 2040 diskette placed into a 4040 drive will be recognized as being 2040 format.

The directory is read from disk the "hard" way – as a bit map. This allows us to see things that a "normal" directory won't tell us, allowing us to find file starting locations and to see scratched files if we wish.

This program won't attempt to read a disk which is bad format. If you can't initialize a disk, this won't help you.

Error Procedures

If the program stops on an error, we have a delicate condition – an opened direct file. Unless you really know what you're doing, I would recommend removing the diskettes and turning the power off the disk unit.

If you find a problem on a disk, get it out of your inventory as quickly as possible. Copy the files you can salvage over to a fresh diskette. Diskette problems don't solve themselves: once a disk is in trouble, the errors can propogate and eventually harm your good files. It won't happen often; but none of us need to have it happen even once.


```
100 PRINT"DISK CHECKER  - JIM BUTTE
    RFIELD"
110 DIM A(255),C%(77,28),D%(1),N$(2
    24),T%(224,1),S%(224,1),L%
    (224),R%(77)
120 D%(Ø)=58:D%(1)=42:Z$=CHR$(Ø)
130 DATA 1,17,2Ø,24,19,3Ø,17,35,16,
    Ø
140 DATA 65,17,2Ø,24,18,3Ø,17,35,16
    ,Ø
150 DATA 67,39,28,53,26,64,24,77,22
    ,Ø
160 B$=CHR$(17):INPUT"DRIVE#";D$:IF
    D$="S"THEND$="Ø":B$=CHR$(3
    )
170 IFD$<>"Ø"ANDD$<>"1"GOTO16Ø
180 OPEN15,8,15,"I"+D$:GOSUB3Ø2Ø
190 OPEN3,8,3,"$"+D$:GOSUB3Ø2Ø
200 AØ=1:GET#3,A$:A=ASC(A$+Z$)
210 READA1:IFA=A1GOTO25Ø
220 F%=F%+1:IFF%=3GOTO29Ø
230 READA1:IFA1=ØGOTO21Ø
240 GOTO23Ø
250 READA1:IFA1=ØGOTO27Ø
260 READB1:FORJ=AØTOA1:R%(J)=B1:NEX
    TJ:AØ=J:GOTO25Ø
270 IFA=1ORA=65THEND1=1:T9=35:S9=3:
    D9=18
280 IFA=67THEND1=257:T9=77:S9=4:D9=
    39
290 IFT9=ØTHENCLOSE3:PRINT"?? DISK ~
    NOT RECOGNIZED ??":STOP

300 REM    GET AND PRINT BAM
310 PRINT"        FREE BLOCK MAP"
320 FORJ=1TOD1:GET#3,A$:NEXTJ
330 FORJ=1TOT9:T1=Ø
340 IFJ=51THENGET#3,A$,A$,A$,A$
350 GET#3,A$:C=ASC(A$+Z$)
360 PRINTRIGHT$(" "+STR$(J),2);" ";

370 K1=Ø:FORK=ØTOS9-1:GET#3,A$:A=AS
    C(A$+Z$)
380 FORL=ØTO7:A%=A/2:D1=A-A%*2:IFK1
    <=R%(J)THENC%(J,K1)=D1:PRI
    NTCHR$(D%(D1));
390 A=A%:T1=T1+D1:K1=K1+1:NEXTL,K:P
    RINT
400 IFT1<>CTHENPRINT"?";
410 NEXTJ
420 REM    DO SPECIFIC JOB
430 PRINT:CLOSE3:PRINT "    CHOOSE ~
    --"
440 PRINT "1. CHECK ALL FILES"
450 PRINT "2. CHECK FOR BAD SPOTS"
460 PRINT "3. RECOVER SCRATCHED FIL
    E"
470 PRINT "  YOUR CHOICE? ";
480 GETX$:IFX$=""GOTO48Ø
490 X=ASC(X$)-48:IFX<1ORX>3GOTO48Ø
500 PRINTX$:OPEN2,8,2,"#Ø":GOSUB3Ø2
    Ø
510 ONXGOTO52Ø,79Ø,89Ø
520 REM     CHECK FILES
530 T=D9:S=1
540 GOSUB2ØØØ
550 FORD=2TO255STEP32:IFA(D)<128GOT
    O59Ø
560 D3=D3+1:T%(D3,Ø)=A(D+1):S%(D3,Ø
    )=A(D+2):L%(D3)=A(D+28)+A(
    D+29)*256
570 IFA(D)=132THENT%(D3,1)=A(D+19):
```

```
    S%(D3,1)=A(D+20)
580 N$="":FORK=D+3TOD+18:N$=N$+CHR$
    (A(K)):NEXTK:N$(D3)=N$
590 NEXTD
600 T=A(0):S=A(1):IFT=D9GOTO540
610 FORD=1TOD3:L%=0
620 PRINTN$(D);
630 T=T%(D,0):S=S%(D,0)
640 IFT>T9ORS<0THENT=0
650 IFT<1ORS>R%(T)THENPRINT" BAD CH
    AIN":GOTO770
660 IFC%(T,S)=1THENPRINT" UNALLOCAT
    ED BLOCKS":GOTO770
670 IFC%(T,S)>1THENPRINT" CONFLICT ~
    ";N$(C%(T,S)-1):GOTO770
680 C%(T,S)=1+D
690 GOSUB3000
700 L%=L%+1
710 FORJ=0TO1:PRINT#15,"M-R";CHR$(J
    );B$:GET#15,A$
720 A(J)=ASC(A$+Z$):NEXTJ
730 T4=T:S4=S:T=A(0):S=A(1):IFT<>0A
    NDE=0GOTO640
740 T=T%(D,1):S=S%(D,1):T%(D,1)=0:I
    FT<>0GOTO640
750 IFL%<>L%(D)THENPRINT" INCORRECT
     BLOCK COUNT":GOTO770
760 PRINT:PRINT"{UP}                ~
             {UP}"
770 NEXTD
780 PRINT:PRINTD3;"FILES":GOTO1270
790 REM   SCAN SECTORS
800 IFF%=0THENPRINT"SORRY .. CAN'T ~
    DO IT":GOTO1270
810 FORT=1TOT9:PRINT"TRACK";T
820 FORS=0TOR%(T)
830 PRINT"{UP}{10 RIGHT} SECT";S
840 GOSUB3000
850 NEXTS
860 PRINT"{UP}                    {
    UP}"
870 NEXTT
880 PRINT"DISK OK":GOTO1270
890 REM   UNSCRATCH
900 K=0:PRINT"I WILL LOOK FOR DISCA
    RDED FILES..."
910 T=D9:S=1
920 GOSUB2000
930 FORD=2TO255STEP32:IFA(D)<>0ORA(
    D+1)=0GOTO980
940 IFK=0THENPRINT"DO YOU WANT TO R
    ECOVER:"
950 GETX$:FORK=D+3TOD+18:PRINTCHR$(
    A(K));:NEXTK:PRINT"? ";
960 GETX$:IFX$<>"Y"ANDX$<>"N"GOTO96
    0
970 PRINTX$:IFX$="Y"GOTO1010
980 NEXTD
```

```
990 T=A(0):S=A(1):IFT=D9GOTO920
1000 PRINT"THAT'S ALL ":GOTO1270
1010 T6=T:S6=S:D6=D:T=A(D+1):S=A(D+2
     ):L%(0)=A(D+28)+A(D+29)*25
     6:L%=0
1020 GETX$:PRINT"IS THIS FILE:"
1030 PRINT" 1. SEQUENTIAL"
1040 PRINT" 2. PROGRAM"
1050 PRINT" 3. USR"
1060 IFA(D+19)=0GOTO1080
1070 PRINT" 4. RELATIVE"
1080 PRINT"   WHICH NUMBER? ";
1090 GETX$:IFX$=""GOTO1090
1100 X=ASC(X$)-48:IFX<1ORX>4GOTO1090

1110 PRINTX$:X=X+128
1120 IFX=132THENT%(0,1)=A(D+19):S%(0
     ,1)=A(D+20):IFT%(0,1)=0GOT
     O1020
1130 IFT>T9ORS<0THENT=0
1140 IFT<1ORS>R%(T)THENPRINT" BAD CH
     AIN!":GOTO1260
1150 IFC%(T,S)=0THENPRINT" ALLOCATED
      BLOCKS!":GOTO1260
1160 GOSUB3000:L%=L%+1
1170 FORJ=0TO1:PRINT#15,"M-R";CHR$(J
     );B$:GET#15,A$
1180 A(J)=ASC(A$+Z$):NEXTJ
1190 T4=T:S4=S:T=A(0):S=A(1):IFT<>0G
     OTO1130
1200 T=T%(0,1):S=S%(0,1):T%(0,1)=0:I
     FT<>0GOTO1130
1210 IFL%<>L%(0)THENPRINT" INCORRECT
      BLOCK COUNT!":GOTO1260

1220 T=T6:S=S6:D=D6
1230 GOSUB3000
1240 PRINT#15,"M-W";CHR$(D);B$;CHR$(
     1);CHR$(X)
1250 PRINT#15,"U2:2,";D$;T;S:GOSUB30
     20:GOTO1300
1260 PRINT "SORRY - IT WON'T WORK"
1270 CLOSE2
1280 INPUT"** GOT TIME TO VERIFY/COL
     LECT DISK";X$
1290 IFASC(X$)=78THENEND
1300 CLOSE2:PRINT#15,"V";D$:END
2000 REM    GRAB FULL DISK BLOCK
2010 GOSUB3000
2020 FORJ=0TO255:PRINT#15,"M-R";CHR$
     (J);B$:GET#15,A$
2030 A(J)=ASC(A$+Z$):NEXTJ:RETURN
3000 REM    READ BLOCK
3010 PRINT#15,"B-R:2,";D$;T;S
3020 REM   GET ERROR STATUS
3030 INPUT#15,E,E$,E1,E2
3040 IFE<>0THENPRINT"{REV}DISK ERROR
     :{OFF}"E;E$,E1;E2
3050 RETURN
```

©

INSIGHT: Atari

Bill Wilkinson
Optimized Systems Software
Cupertino, CA

This month, I present a session on how to steal a system. Before all you kleptomaniacs rejoice, though, I should explain that I mean to show you how to take control of your Atari's software system when a user pushes the SYSTEM RESET button. This will, I hope, be useful to BASIC and assembly language programmers alike.

There will be more on the inner workings vs. outer appearances of Atari BASIC; and, as space permits, I will have my usual assortment of cute tricks and Did You Knows.

In a departure from the norm, I will review a product here. Since my company, Optimized Systems Software, both solicits and sells software for Atari (and Apple) computers, I do not think it would be fair for me to do software reviews. But, unless you and/or my dear editor object, I may, from time to time, discuss new and wondrous happenings in the world of Atari.

A Short Review

It generally strikes me as unfair for a magazine to carry a review of something it sells; but every other magazine does it, so I prevailed on **COMPUTE!** to let me review *COMPUTE!'s First Book of Atari.* I am doing so on the condition that the review must be run as I submit it. (Okay, okay, Richard...you can fix my punctuation.) I have to do a good review or they won't let me do it again (just kidding...I think).

First, let me say that I did not start reading **COMPUTE!** regularly until about December, 1980, so most of the material in the book was new to me. Boy was it new to me! Quite frankly, I had lulled myself into thinking that, until I started writing my column, the poor Atari user had no insight (an insight gag) into the workings of Atari BASIC, DOS, etc. *Not so!!* There was a *lot* of good stuff published in **COMPUTE!** during 1980.

I don't want to make this review sound like a whitewash, so let's get the bad stuff over with first. The first warning that needs to be given is that, in general, this is not a book for software hackers: there is little of interest to the assembly language programmer (but see below for some notable exceptions), and the person who has read *and understood* (!!) the technical manuals and, perhaps, *De Re Atari* won't find much he or she didn't know. However, for most people there is much useful material here. There are some bloopers in the material presented, things which probably wouldn't get past the current, more Atari-sophisticated, editorial staff. There's a little duplication of material. And there's a lot of stuff that has been updated by better articles which have appeared (usually in **COMPUTE!**) in 1981 and 1982. Actually, my biggest complaint is in a reprinted article titled "The OUCH in Atari BASIC": the article states, *and* the editorial lead-in agrees (and the lead-in was written recently – Oh, for shame!), that keywords can't be used in variable names. Yet, the very next article in the book says that all keywords can be used as variable names! (Still not quite true – "NOT" is poison as the first three characters of a name, and a few keywords, such as "TO" and "STEP" can't be used as-is. Oh well, this was 1979 and 1980. And, come to think of it, even *Atari's BASIC Reference Manual still* says not to use keywords as names. Of course, it also says that substrings are limited to 99 characters, so maybe it's not a good reference point.)

OK. So much for the bad stuff. "Not possible!" you scream? Sorry, but it's true. I really don't have any dirt to sling. Oh, some of the little example programs might now be found in the Atari manuals, etc., but they aren't *bad,* just not of as much value as the rest of the book. And I wish I had the time and space to correct every little goof I found. (But I gotta tell you *one:* the order and size of variables and their names has *no* impact at all on the speed of an Atari BASIC program. Honest.) With those caveats in mind, we examine the value of the book.

And the book is of value. If you had to choose between losing your left pinkie (not quite up to the left thumb, anymore), the *First Book,* and *De Re Atari,* you should really think about how useful a little pinkie is. If you *must* choose between *De Re* and *First Book,* let your experience level be your guide: if you almost understand the Atari technical manuals, you are probably ready for *De Re.* If you are just learning to program, stick with *First Book.* If you're in the middle, better let the little pinkie go.

My own favorite pair of articles from the book are "Inside Atari BASIC" and "Atari Tape Data Files," both by Larry Isaacs. I am just now getting to the point where I am discussing things in "Insight: Atari" that Larry explored over a year ago (there will be overlap, hopefully to your benefit). Other articles worth mentioning include the following (an asterisk indicates something of interest to assembly language buffs):

"Printing to the Screen from Machine Language"*
(not because of what the presented program does as much as for some of the techniques it introduces)

"The Fluid Brush"
(Ditto. And its ideas have been much copied.)

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

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!

#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.

#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.

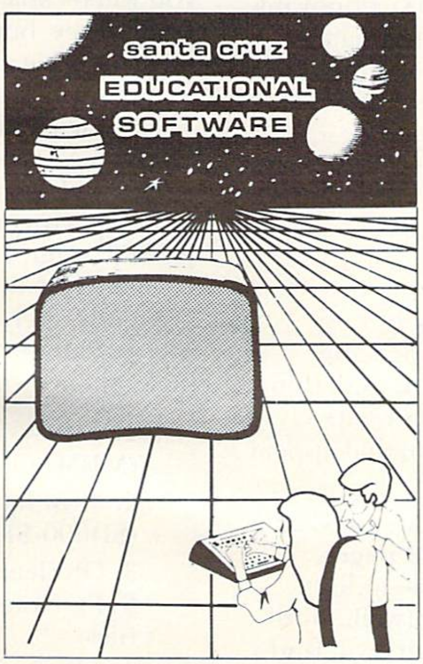
#4: BASICS OF ANIMATION — Shows you how to animate simple shapes using the PRINT and PLOT commands, and also has nice little PLAYER/MISSILE Graphics demo to learn. This would be an excellent way to start making your programs come alive on the screen. Recommended for new owners.

#5 PLAYER MISSILE GRAPHICS — This complex subject is demonstrated by starting with simple examples, and building up to a complete game and also an animated business chart on multiple pages! As always, the computer does most of the calculations. Requires 32K disk or tape and costs **$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.

Tricky Tutorials (except #5) require 16K memory for cassette orders and 24K for disk. The price is $19.95each. You may order 1,2,3, & 4 for $64.95.All six in a colorful binder cost $99.95.

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**

By the time you read this all computers (400/800) being produced should have the fabled GTIA chips included. ATARI service may upgrade older computers...call and ask (it's easy to do yourself). We have one and the improvements that graphics modes 9,10, and 11 offer are great!! To help you figure out what to do with the new modes a new Tricky Tutorial will be offered in March on Modes 9 to 11. Either give us a call or write around that-time.

****

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
***Includes $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 *

"Player/Missile Graphics" *
(by Chris Crawford. What more need I say?)

"Adding a Voice Track to Atari Programs" *
(This one was even swiped! There are more sophisticated methods shown in *De Re*, but this is adequate for many purposes.)

"Atari Memory Locations" *
(Just a table. You need to read the *Technical Manual* and/or *De Re* first, but this will serve as a handy reminder.)

"Input/Output on the Atari"
(I hesitated on this one: you should ignore what it says about XIO! It's misleading. Read my Atari I/O series.)

You'll note that most of that stuff is kind of heavyweight. Well, that's what appeals to me and, I think, to a large portion of **COMPUTE!**'s Atari readership. However, there are several little goodies, usable by virtually anyone, which deserve honorable mention. No commentary on these: their names tell it all and you just have to try them to appreciate them:

"Reading the Atari Keyboard on the Fly"
"Al Baker's Programming Hints"
"Atari Sounds Tutorial"
"Using the Atari Console Switches"
"Atari Meets the Real World"

You may have your own favorites, but my criterion for a good article (or good magazine-published program) is that it teaches you something. Thus I rate type-it-in-and-run-it games relatively low. (There are remarkably few of them that appear in the book.)

In final summary, I have to say that, for $12.95, you are unlikely to find this much (184 pages, including – can you believe it – a usable index) useful Atari material presented again (well...until the *Second Book* ?). Real software hackers will find some of the material too elementary, but they are probably the only ones that will be disappointed.

Stealing A System

During my series on Atari I/O (**COMPUTE!** November, 1981, through March, 1982, issues 18 through 21), I mentioned (more than once) the "proper" way to add device drivers to OS. I summarize it here again:

1. Inspect the system MEMLO pointer (at $2E7).

2. Load or relocate your routine at/to the current value of MEMLO.

3. Add the size of your routine to MEMLO.

4. Store the resultant value back in MEMLO.

5. If your routine is a device driver, connect it to OS by adding its name and address to HATABS.

6. Fool OS so that steps three through five will be re-executed if SYSTEM RESET is hit.

In **COMPUTE!** (January, 1982, #20) we added the driver for the "M:" device by following steps one through five as above. We discussed step six briefly, but did not show how to implement it. This month, we will show how to fool OS. And, rather than repeating the lesson about adding device drivers, we will take this opportunity to show how to give Atari BASIC some measure of control over what happens on RESET.

In particular, we "steal" the system in a way that the user who hits RESET will cause a TRAPpable error in the running BASIC program. In other words, if you write your BASIC program in a way that TRAP (to a line number) is always active, you will be able to detect when your user hits the RESET key, but your program will not stop running, will not lose its variable values, and will be impacted in the minimum possible way.

Some cautions are in order (it seems like I always have to say that): *before* vectoring through RAM (and thus allowing our little trick) Atari's OS ROMs perform several actions when SYSTEM RESET is hit. If you need to know exactly what happens, try to get hold of the CIO listings (they are moderately readable); generally, the following lists all that matters except to those who would make their own cartridges:

1. The system resets any memory size pointers (MEMLO, MEMTOP, etc.).

2. Most hardware registers are reset to zero ($D000-$D0FF, $D200-$D4FF).

3. OS clears its own RAM ($200-$3FF, $10-$7F). Note that this zaps all IOCB's for all files.

4. All the ROM-based device drivers are initialized (via their own initialize routines).

5. CIO's initialization is called, which effectively marks all files as properly closed.

6. Screen margins, etc., are reset and the E: device is opened on file channel #0 (which is equivalent to GRAPHICS 0 from BASIC).

7. The file manager's initialization routine is called via an indirect call through location DOSINIT ($0C).

8. *If* there are no cartridges installed, then DOS is invoked by an indirect jump through location DOSVEC ($0A). If a cartridge is installed and wants control, though, OS goes to the cartridge instead of DOSVEC.

(NOTE: OS/A + uses a variation on 7., above, so don't bang your head against the wall trying what is written here with OS/A+. I will be glad to tell you of the differences if the manual is not clear enough.)

Program 1 takes into account not only all of the above, but also the requirements of Atari BASIC related to executing a pseudo-warmstart. I will not try to explain why the various JSRs and tests shown are needed; just take my word for it that they are indeed necessary (I found out the hard way). Actually, the part pertaining to stealing the DOSINIT vector is straightforward, as you may note, and changing MEMLO is trivial.

Once again, for space and time reasons, I have cheated with this program: I have assumed that my routine can load and execute at $1F00 and can move MEMLO to $2000. Those of you who want to do the whole thing right can follow the techniques I showed in **COMPUTE!** February, 1982, #21, for generating relocatable programs. Also, please note that the listing, as is, is designed to produce an AUTORUN.SYS file. You may need to do a little surgery to use it in other ways (e.g., remove the load-and-go vector at the end, JMP directly to the start of the BASIC cartridge, etc. – experiment).

The most important thing to note about this routine's implementation is how the address found in DOSINIT is moved into the JSR instruction (at the label RESET). Obviously, you could go look at the contents of DOSINIT and code the JSR directly, omitting the move of the address. And this will work as long as you use the same version of OS and DOS. *But* ... all too many Atari software developers fell into the trap of thinking that OS and DOS were immutable, only to have Atari announce DOS 2S and OS version B.

To Atari's credit, they have

carefully documented which locations, vectors, etc., are guaranteed to remain unchanged. If you write your code properly, you should never have to change it. Incidentally, another example of this same concept appeared in my last article: the vector from VVBLKD was preserved, rather than simply JuMPing to the routines in ROM.

Enough preaching: investigate the listing. But I leave you with one last freebee. If you change three lines of code in the listing (lines 1950 to 1970) to the following two lines, you will cause BASIC to reRUN the program currently in memory, rather than causing an error TRAP.

```
1950 JSR $B755
1960 JMP $A962
```

The following short BASIC program illustrates the use of our stolen pointer:

```
 10 TRAP 100
 20 PRINT "LOOPING AT
    LINE 20"
 30 GOTO 20
 40 STOP : REM
    can't get here from there...
    or anywhere
100 IF PEEK(195)<>255 THEN
    PRINT "HOW? WRONG
    ERROR CODE!":STOP
110 PRINT "RESET KEY WAS
    HIT...I WILL START
    AGAIN"
120 RUN
```

Note that you *can't* get out of this program with the RESET key! Now, if you also trap the BREAK key, the user is truly locked in your program. If you have BASIC A+, of course, you can trap the BREAK key via SET 0. If not, then refer to the listing titled "Idiot-Proofing the Keyboard" in *De Re Atari*. (Summary of that listing: since the BREAK key is one of the two IRQ's not vectored through RAM, you must change the system master IRQ vector to point to your own routine. In your routine, you check for and ignore BREAK key IRQ's and pass other IRQ's on unchanged. Not trivial, perhaps, but certainly less complicated than what we have done above.)

Inside Atari BASIC, Part 3: Enhancements

After skipping last month, we return to our discussion of the hows and whys of Atari BASIC. Recall that in **COMPUTE!** February, 1981, #21, we discussed how BASIC checks your entered line for correct syntax and produces a tokenized result. Let us begin this month with a discussion of how BASIC executes (RUNs) a program.

First, note that if you enter a direct line (one without a line number), BASIC arbitrarily assigns it to line number 32768 and then pretends that it is like any ordinary line. That means that even direct lines must go through the tokenizing and execution process. It also means that BASIC makes little or no distinction between statements (within a program) and commands (given directly); thus you can LIST or RUN or even CONTinue from inside a BASIC program.

Whenever a line is finished executing, BASIC checks to see if the next line exists (it doesn't if a direct line was just executed) or if the next line has a line number greater than 32767 (i.e., if the line executed is the last one prior to the direct line). If either condition prevails, BASIC pretends to itself that it got an "END" statement and, presto, you are back staring at the "READY" prompt.

But let us assume that the direct command given was "RUN." The execution of a RUN statement simply causes all BASIC's pointers and flags to be reinitialized, including setting BASIC's "next line" pointer to the beginning of the program. Then RUN returns to what we call "Execution Control" which decides that it needs to start executing the next line...which conveniently is the first line.

So far, so good. But how does BASIC know what to do with the tokens? The answer is that it doesn't, really. Recall that there are two separate kinds of execution (as opposed to variable) tokens: statements and operators. Each of these has a table of two-byte pointers residing in BASIC's ROM. Execute Control simply picks up the next byte of the program, assumes that it is a statement token (incidentally, in the range of $00-$7F), and uses double its value as an index into the table of statement pointers. It uses the address thus found as an indirect jump and goes to the appropriate statement execution routine.

In a non-syntaxed BASIC (i.e., Microsoft), much of the preceding applies virtually unchanged. But, when the statement execution routine gets control in such a BASIC, it has no idea what the next character or token in the program might be, so it must needs go through a set of checks to determine what is legal and what is not.

In Atari BASIC, though, the statement execution routine *knows* that the byte(s) that follow constitute legal syntax! So it need not waste time checking for legality. Since the bytes following the statement token may range from the non-existant (as in CONT, which has no following bytes) to the extremely complex (as in PRINT, in all its variations), each statement generally has responsibility for choosing what to do with these bytes.

With the exception of assignment-type operations (LET, READ, INPUT, etc.), file designators (PRINT #), and complex statements (FOR...TO...STEP), what follows the statement byte is generally a series of one or more expressions, separated by commas, equal signs, semicolons, etc. Thus it comes as no surprise that there is a major subroutine in Atari BASIC entitled "Execute Expression," which can evaluate virtually any numerical or string expression.

As a simple example, let us examine the mechanism of POKE. The syntax is properly "POKE <aexp>,<aexp>" (where <aexp> means Arithmetic EXPression). So POKE's statement execution routine simply calls Execute Expression for the first value, saves it away someplace safe, skips the comma (it *knows* the comma is there...the syntaxer said so!), calls Execute Expression for the second value, and stores the second into the memory location designated by the first. Now, in truth, POKE calls a variation on Execute Expression which is guaranteed to return a 16-bit (or 8-bit, as required) value; but the concept holds for most statements.

It is really beyond the scope of this article to try to explain the intricacies of Execute Expression. It will suffice to point out that it must worry about operator precedence ("*" before "+", etc.) and parentheses and subscripted variables and functions (SIN, RND, etc.) and more.

And that's about it. Except to note that when a statement is finished it usually simply returns to Execute Control, which checks for another statement on the same line and/or moves its pointer to the first statement of the next line.

Much of the point of this discussion has been to show why it is hard to fool BASIC into believing that it has a new statement to use. Even with the source, it is no easy task to make sure that the correct syntax for a new statement is entered into the syntax tables (which are actually a miniature language in their own right), the name tables, and the execution tables (to say nothing of writing the code to execute the statement). With Atari BASIC locked in ROM, the task is really impossible since BASIC makes use of no RAM-based pointers or indirect jumps throughout this process.

So how can we add features to Atari BASIC? Several ways:

1. Try the USR function as suggested last month. This really is the simplest, most straightforward, most guaranteed-to-work.

2. Make your own special device handler (a la "M:" in **COMPUTE!**, January, 1981, issue #20). Open a channel to it (OPEN #1,...). PRINT something to it. When your driver gets control, it can actually go in and look at the BASIC tokens and decide what to do from there. Cryptic, but it works.

3. If you are interested in commands, as opposed to statements, you can intercept BASIC's call to "E:" (for the next input line) and examine the line yourself (presumably as does Monkey Wrench). This implies that you must check syntax, find variables, convert ASCII to floating point, etc., in your routine. Tedious, but obviously feasible and usable.

As you can, no doubt, tell, I am much in favor of method 1. It is by far the easiest to do and requires the least knowledge of BASIC's internals.

Is there yet another way? A month ago I would have said *"no!"* But, now, I have discovered a crack in the door. It is complicated, prone to programmer error, fairly inflexible, and of doubtful value for anyone but professional software developers. To explain it would take a couple of more columns, and I'm simply not willing to write that much on a topic of dubious value. If you feel you absolutely *must* know, write me (care of OSS). If enough people write, I *may* make up a pamphlet and sell it at an outrageous price. Are you sure you can't live with method 1?

Easy Horizontal Joysticks

If you have an application (a polite way of saying game) that needs a joystick that moves only horizontally (or only vertically, if you are willing to hold your joystick turned 90 degrees from "normal"), then have I got a trick for you! Try this program, with joystick number 0 plugged in:

```
10 PRINT PTRIG(0)-PTRIG(1),
20 FOR J=1 TO 50 : NEXT J
30 GOTO 10
```

Now push the joystick in all directions. Neat? Pushing it left gives you a value of -1 and right gives you +1. And, of course, you can use A= PTRIG(2)-PTRIG(3) to read joystick number 1, etc.

Why does it work? Because the paddle triggers happen to use the same pins on the connector that the horizontal switches in the joystick use. I discovered that by reading the technical manual; so, you see, there is probably still buried gold in those books.

Unfortunately, no such happy coincidence exists for reading the vertical joystick switches. Incidentally, use of this trick does not affect STRIG in any way.

Dissonances

The algorithm Atari gives for figuring out what *actual* frequency will result from the divider FREQ in (for example) SOUND 0,FREQ,tone,volume is as follows:

$$\text{actual frequency} = \frac{63921}{\text{FREQ}+\text{FREQ}+2}$$

This means that, at values for FREQ around 85 (the middle of the Atari's frequency range), the minimum actual frequency step is about 4 Hz. While adequate for solo parts, this kind of frequency resolution can really grate on your ears when there are three or four voices active. To illustrate the real meaning of this, try the following one-liner:

```
FOR F=255 TO 0 STEP -1 : SOUND 0,F,10,15 :
    NEXT F
```

The resultant sound is a smooth glide until you get near the top end, when you begin to really hear the steps.

For those of you with a keen ear and/or a strong sense of music, cheer up. Atari, once again, gave us a solution. The entire Atari audio system is controlled by hardware register AUDCTL ($D208). Normally, the audio channels are clocked by an oscillator running at 63921 Hz. But, the user may specify that channels zero and two (which Atari calls one and three in the *Technical Manual*... oh well) are to be clocked by a 1,789,790 Hz oscillator. If you change 63921 to 1789790 in the formula above and plug in 255 (the highest value) for FREQ, you will see that the *lowest* note thus playable is around 3000 Hz!

But we have yet another solution available via AUDCTL: instead of an 8-bit counter for a single audio channel, we use a pair of channels to produce a 16 bit counter. (Unfortunately, we then are limited to two sound channels.) The modified formula then becomes:

$$\text{actual frequency} = \frac{1789790}{\text{FREQ}+\text{FREQ}+14}$$

Since FREQ now has values from 0 to 65535, it's obvious we have many more actual frequencies available to us. I present herewith two sample programs using this technique. Program 2 shows two voices doing very smooth glides. Program 3 shows a 9-octave chromatic scale (C, C#, D, D#, etc.). This compares to the 3-octave scale available

ALTERNATE REALITY SOFTWARE

Presents for the Atari

THE I CHING

THE ANCIENT CHINESE BOOK OF DIVINATION

- The complete text of the world's oldest book on disk
- 40K program
- 73 disk files (155,000 bytes)
- Occupying 1211 disk sectors
- High Resolution Graphics
- Animation
- Music and Sound Effects
- Instructional text material
- Calculates and Displays Hexagrams
- Displays Judgement, Image, Moving Lines for primary & secondary Hexagrams
- $44.95

order from:

Alternate Reality Software
2111 W. Arapahoe Drive
Littleton, Colorado 80120

Dealer inquiries invited
Atari is a T.M. of Atari, Inc.

ZAP

THE BIG MATH ATTACK™

Challenging new math program . . .
Sharpen your skills by entering the correct answer before the equation 'lands' on your city! Provides hours of educational entertainment.

POW

Features:

- Full color
- High resolution graphics
- Animation
- Sound
- Four math functions (+, -, ×, ÷)
- Two levels for each function

Recommended for grades 1-6. Available for the ATARI & APPLE II.

ATARI 16K (cass.) $20.00
ATARI 24K (disk) $25.00
Requires ATARI BASIC cartridge
APPLE II (disk) DOS 3.2/3.3 .... $25.00
Requires Applesoft Basic in ROM.

Ask for it at your local computer store.

or order direct:

T.H.E.S.I.S.
P.O. Box 147
Garden City, MI 48135
(313) 595-4722

Please add:
$1.50 shipping/handling
$1.50 C.O.D.
WRITE FOR FREE CATALOG
DEALER INQUIRIES WELCOME

ATARI®

Software for Personal Computers

A collection of 10 challenging programs created to provide a unique entertainment value — and 2 personal/business programs with broad functional value. In disk and/or cassette as indicated.

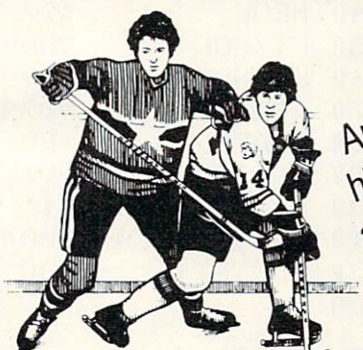
GAMES	• HELICOPTER BATTLE	Req. — 16K RAM/Cassette	$ 9.95
		— 16K RAM/Disk	14.95
	• HORSE RACING	Req. — 16K RAM/Cassette	9.95
		— 16K RAM/Disk	14.95
	• KENO	Req. — 8K RAM/Cassette	9.95
		— 16K RAM/Disk	14.95
	• LIGHTNING BOLTS and REACTION	Req. — 16K RAM/Cassette	9.95
		— 24K RAM/Disk	14.95
	• THE MAD MARBLE	Req. — 8K RAM/Cassette	9.95
		— 16K RAM/Disk	14.95
	• MUSIGAME (2 Games)	Req. — 16K RAM/Cassette	9.95
		— 24K RAM/Disk	14.95
	• SUPERMASTER	Req. — 8K RAM/Cassette	9.95
		— 16K RAM/Disk	14.95
	• TAG	Req. — 16K RAM/Cassette	9.95
		— 16K RAM/Disk	14.95
	• TRACTOR BEAM	Req. — 8K RAM/Cassette	9.95
		— 16K RAM/Disk	14.95
	• WAR AT SEA	Req. — 16K RAM/Cassette	14.95
		— 24K RAM/Disk	19.95
BUSINESS	• CCA Data Management System	Req. — 48K RAM/Disk	99.95
	• LETTER WRITER	Req. — 24K RAM/Disk	19.95

® Trademark of ATARI, Inc.

DIVISION OF CUSTOM ELECTRONICS, INC.
SOFTWARE

238 Exchange St., Chicopee, Massachusetts 01013
(413) 592-4761
Mastercard & VISA Accepted
• Dealer And Distributor Inquiries Invited
• Closed Mondays — Open Daily 'Til 5:30 — Fridays 'Til 8

HOCKEY



An outstanding, high-speed video action game for the Atari 400/800 *

HOCKEY is for 2, 3 or 4 players. Carry the puck, pass, shoot, steal the puck, intercept passes, and block shots on goal! High-resolution color graphics. All machine language.

HOCKEY comes on cassette or diskette. Price is $29.95. Order from your local dealer or from

Gamma Software
P.O. Box 25625, Los Angeles, CA 90025,
(213) 473-7441

California residents add 6% sales tax. Add $2.00 for shipping and handling.

Dealer inquiries invited.

Game requires joysticks and minimum of 16K RAM.

*Indicates trademark of Atari, Inc.

via the standard SOUND commands.

Finally, if you simply need lower notes than you can get with SOUND, TRY POKEing AUDCTL to one. This has the effect of lowering *all* notes by

Program 1.

```
equates and commentary

0000            1000            .PAGE "equates and commentary"
                1010 ;
                1020 ; STEAL A RESET
                1030 ;
                1040 ; listing for COMPUTE! magazine, April, 1982
                1050 ;
                1060 ;
                1070 ; there are two parts to this listing:
                1080 ;   1. what happens when this is first loaded
                1090 ;      (which initializes everything)
                1100 ;   2. what happens when the user pushes
                1110 ;      SYSTEM RESET.
                1120 ;
                1130 ; Most of 1 is understandable.  Most of 2
                1140 ; is magic.  If it works, don't knock it.
                1150 ;
02E7            1160 MEMLO    =    $2E7         ; BOTTOM OF USABLE MEM
                1170 ;
00FF            1180 LOW      =    $FF
0100            1190 HIGH     =    $100
                1200 ;
                1210 ; EQUATES INTO BASIC ROM
                1220 ;
BD72            1230 SETDZ    =    $BD72        ; ENSURE OUTPUT TO CONSOLE
0092            1240 MEOL     =    $92          ;FLAG: LINE MODIFIED
BD99            1250 FIXEOL   =    $BD99        ; UNMODIFY
00B9            1260 ERRNUM   =    $B9          ; AT LEAST BASIC THINKS SO
B940            1270 ERROR    =    $B940        ; HANDLE ERRORS
00BD            1280 TRAPFLAG =    $BD
DA51            1290 INITBUF  =    $DA51        ; SAFETY
0011            1300 BRKFLAG  =    $11
BD41            1310 CLOSEALL =    $BD41        ; close IOCBs 1-7
000C            1320 DOSINIT  =    $0C          ; see article
                1330 ;

SETUP THE RESET VECTOR

0000            1340            .PAGE "SETUP THE RESET VECTOR"
                1350 ;
                1360 ; We move the OS DOSINIT vector to point to ourselves
                1370 ;
                1380 ; ***** NOTE: change next line to suit!!! *****
0000            1390            *=   $1F00
                1400 CHANGEDOSINIT
1F00 A50C       1410            LDA  DOSINIT
1F02 8D231F     1420            STA  RESET+1
1F05 A50D       1430            LDA  DOSINIT+1 ; Self modifying code...nasty
1F07 8D241F     1440            STA  RESET+2
1F0A A922       1450            LDA  #RESET&LOW
1F0C 850C       1460            STA  DOSINIT
```

```
1F0E A91F   1470          LDA  #RESET/HIGH
1F10 850D   1480          STA  DOSINIT+1 ; We have changed the pointer
            1490 ;
            1500 ; Here we move MEMLO...
            1510 ;   we arbitrarily use 256 bytes of space
            1520 ;
            1530 MOVEMEMLO
1F12 A900   1540          LDA  #RESETTOP&LOW
1F14 8DE702 1550          STA  MEMLO
1F17 A920   1560          LDA  #RESETTOP/HIGH
1F19 8DE802 1570          STA  MEMLO+1
1F1C 60     1580          RTS
            1590 ;
            1600 ; FROMBASIC  is just a second entry
            1610 ;   entry point into the initialization...
            1620 ;   for initializing from BASIC
            1630 ;
            1640 FROMBASIC
1F1D 68     1650          PLA                   ; get cnt of parms off stack
1F1E F0E0   1660          BEQ  CHANGEDOSINIT ; good...no parms
1F20 D0FE   1670 OOPS     BNE  OOPS             ; otherwise, tough!
THE ACTUAL RESET TRAP
1F22        1680          .PAGE "THE ACTUAL RESET TRAP"
            1690 ;
            1700 ; On reset, DOS normally gets
            1710 ; called to reinitialize itself...
            1720 ; we use this to our advantage by
            1730 ; reinit'ing both DOS and BASIC
            1740 ;
            1750 RESET
1F22 200000 1760          JSR  0                ; second two bytes get replaced
            1770 ;                           by the address of real DOSINIT
1F25 A2FF   1780          LDX  #$FF
1F27 9A     1790          TXS                   ; BASIC likes it this way
1F28 8611   1800          STX  BRKFLAG          ; ensure no BREAK key pending
1F2A 2041BD 1810          JSR  CLOSEALL         ; so everybody agrees
1F2D 2072BD 1820          JSR  SETDZ
1F30 A592   1830          LDA  MEOL
1F32 F003   1840          BEQ  RST2
1F34 2099BD 1850          JSR  FIXEOL
            1860 RST2
1F37 2051DA 1870          JSR  INITBUF
            1880 ;
1F3A 20121F 1890          JSR  MOVEMEMLO ; to protect this code
            1900 ;
            1910 ; NOW we fool BASIC into thinking
            1920 ; an error occured
            1930 ;
            1940 ;
1F3D A9FF   1950          LDA  #255             ; (or your choice of errors)
1F3F 85B9   1960          STA  ERRNUM
1F41 4C40B9 1970          JMP  ERROR            ; do it
            1980 ;
            1990 ; THE FOLLOWING EQUATE IS USED TO SET
```

```
                2000 ; RESETTOP on a page boundary
                2010 ;
2000            2020 RESETTOP =   *+$FF&$FF00
                2030 ; SET UP LOAD AND GO
                2040 ;
1F44            2050          *=   $2E0
02E0 001F       2060          .WORD CHANGEDOSINIT
02E2            2070          .END
THE ACTUAL RESET TRAP

=02E7 MEMLO       =00FF LOW              =0100 HIGH      =BD72 SETDZ
=0092 MEOL        =BD99 FIXEOL           =00B9 ERRNUM    =B940 ERROR
=00BD TRAPFLAG    =DA51 INITBUF          =0011 BRKFLAG   =BD41 CLOSEALL
=000C DOSINIT      1F00 CHANGEDOSINIT     1F22 RESET      1F12 MOVEMEMLO
=2000 RESETTOP     1F1D FROMBASIC         1F20 OOPS       1F37 RST2
```

Program 2.

```
10 AUDCTL=53768:DBL=120
20 AUDF1=53760:AUDC1=53761
30 SOUND 1,10,10,15:SOUND 3,10,10,
   15
40 POKE AUDC1,0:POKE AUDC1+4,0
50 POKE AUDCTL,DBL
60 FOR J=10 TO 15:POKE AUDF1+2,J:
   POKE AUDF1+6,20-J
70 FOR I=0 TO 255:POKE AUDF1,I:POKE
   AUDF1+4,255-I:NEXT I
80 NEXT J
   ...VERY SMOOTH GLIDES...
```

Program 3.

```
10 AUDCTL=53768:DBL=120
12 OSC=1789790/2
20 AUDF1=53760:AUDC1=53761
30 SOUND 1,10,10,0
40 POKE AUDC1,0:POKE AUDC1+4,0
50 POKE AUDCTL,DBL
60 P2=2^(1/12)
70 NTE=16:REM C IN THE REAL BASS
80 FOR I=1 TO 109
90 FREQ=INT(OSC/NTE-7+0.5):F0=INT
   (FREQ/256)
92 F1=FREQ-256*F0
100 POKE AUDF1,F1:POKE AUDF1+2,F0
102 POKE AUDC1+2,175
103 PRINT "NOW PLAYING ";INT(NTE+
    0.5);" HZ"
105 FOR J=1 TO 100:NEXT J
110 NTE=NTE*P2
120 NEXT I
130 GOTO 70
  ...9 OCTAVE CHROMATIC SCALE...
```

approximately two octaves. Unfortunately, you do not get to have some channels high and others low. Example:

SOUND 0,60,10,12 : SOUND 1,45,10,12 : POKE 53768,1 : FOR I = 1 TO 500 : NEXT I

Again, investigate *De Re* for even more details on some of the more complex aspects of the sound system. (Want to hear your Atari "MOO" like a cow?)

A Final Caution

I have had a couple of people write or call me complaining that, when they tried my assembly language routines, they didn't work. Honest, they *do* work. The listings published in the magazine are the ones I actually used. Sometimes the problem simply resolves to a typo on the part of the user. But sometimes it turns out to be an address conflict.

I do most of my work with OS/A+ and BASIC A+ (naturally. But I use Atari BASIC to check out programs for these pages), and I usually use memory addresses which are convenient to me. Since I get tired of putting everything in page six, I sometimes use $1F00 or some such as an origin. You *can* use these addresses in your system with the Atari Assembler/Editor if you change MEMLO to, say, $3000 (my usual choice, and achievable with the LOMEM command of EASMD). However, it may be more convenient to use SIZE to look at where your source, etc. is and then change the origin to reflect your memory configuration.

Of course, you can always assemble into the dreaded page six or assemble directly to disk (or cassette). But, in any case, don't use an origin ("*= xxxx") which conflicts with SIZE in your system. I purposely give you source listings so that you can see how things work; take the time to type them in and it will probably prove easier in the long run. ©

A High Resolution Digital-To-Analog Converter For The PET

G. Eric Matthews and
Pamela L. Carter
Department of Physics
Wake Forest University
Winston-Salem, NC

It is relatively simple to interface a microprocessor compatible digital-to-analog converter (D/A) to the Commodore PET. For most applications, an 8 bit signal is adequate. An 8 bit D/A interface can be implemented easily via the User Port. However, in many applications a very accurate voltage is necessary, and an 8 bit signal may be inadequate. A 12 bit signal has 16 times the resolution of an 8 bit signal and therefore allows much finer control over the output voltage. Connecting a 12 bit interface to the User Port can be somewhat complicated since the computer only offers 8 data lines there. However, the PET has a Memory Expansion Port (located on the right side of the cabinet) that includes fully buffered, partially decoded address lines [Ref. 1]. Use of the expansion port offers the additional advantage of allowing multiple interfaces to be connected and used at once merely by assigning each a unique address. The interface described here utilizes the signals available at this port.

Since the computer has only an 8 bit data bus, interfacing a 12 bit D/A is more difficult than implementing an 8 bit device. This can be accomplished through double buffered data latching. A latch holds the signal until the program signals the latch to accept a new value and to pass it on to the next part of the circuit. Signalling the latch to hold or pass data is accomplished from BASIC through POKE commands. The actual interfacing technique follows.

The Hardware

The first step in the design of an analog output port is to determine the accuracy of the conversion that is to be performed. If the desired analog accuracy is ± 1%, or 1 part in 100, then a 7 bit conversion is adequate, with a resolution of better than ½% (one part in 64). ± 0.1% accuracy requires 10 bits to be within the desired range. When the accuracy level has been determined, the digital-to-analog converter chip (DAC) can be chosen. The characteristics of the DAC should meet the requirements of the circuit in which it is to be used. In this interface an Analog Devices AD7541 (CMOS 12-Bit Monolithic Multiplying DAC) was chosen [Ref. 2].

The next step is to determine the addresses to be devoted to receiving the data for the latches. Checking the computer memory to find available space is necessary in order to avoid interfering with other operations of the computer. The most straightforward way of passing a 12 bit number to the DAC is by assigning one address to the upper eight bits of the DAC and assigning a second address to the lower four bits of the DAC. Therefore at least two addresses must be devoted to the interface. For simplicity, only partial decoding of the address bus (the high order eight bits) has been used. The addresses chosen are $90XX and $91XX (where an X means "don't care"). These two addresses do not conflict with the Toolkit or Wordpro. A POKE to $90XX enables a latch for the upper eight bits and a POKE to $91XX enables the corresponding latch for the lower four bits.

A schematic diagram of the circuit employed is shown in figure 1. The chip used for address decoding is the 74154 (4 Line Decoder) [Ref. 3]. This chip decodes four binary-coded inputs to one of 16 mutually exclusive outputs when each of the two strobe inputs are low. Address decoding with the PET is simplified by the availability of partially decoded address lines on the Memory Expansion Port: $\overline{\text{SEL9}}$ is low when the high order four bits of the address bus corresponds to $9. Applied to the strobe inputs of the 74154 are $\overline{\text{SEL9}}$ and A11 for the chosen address. To the four binary-coded inputs A,B,C,D are fed the Ø2 clock, A8,A9,A10 respectively. As configured, output 1 will be low if and only if the Ø2 clock is high, $\overline{\text{SEL9}}$ is low, and A8-11 are low (as during a POKE to $90XX). Output 3 will be low when the Ø2 clock is high, $\overline{\text{SEL9}}$ is low, A8 is high, and A9-11 are low as during a POKE to $91XX). These two outputs provide the enabling pulses for the latches. Note that this circuit does not decode the R/W line. Care should be taken, therefore, that no PEEKs to the above addresses be included in any program using this circuit as this would result in spurious data being passed to the DAC.

The latches used here are 7475 Quad Latches [Ref. 4] and function as temporary storage for binary information. Information at the data inputs is transferred to the outputs when the enable is high. Therefore the two independent (low) output pulses from the 74154 must be inverted before being applied to the 7475 enable pins. These two inverted pulses correspond to the enabling addressed $90XX and $91XX. The output of the latch follows the input as long as the enable is high. When the enable goes high the output is held constant no matter the state of the input until the enable goes high again. The purpose in double buffering the data latches is to pass all 12 data bits at the same instant to the DAC. If the latches were only single buffered, then the eight bit data latch (two 7475's) and the four bit data latch (one 7475) would be enabled with separate POKE commands, and an instant of time would exist when the data input to the DAC is in an intermediate state between the previous input and the present input. The data in this state is erroneous and produces a glitch in the voltage output. Double buffering avoids this glitch by passing the data to the DAC in the following way.

Address $90XX enables the first eight bit latch. This passes the eight most significant bits to the first latch. Then POKEing address $91XX enables the four bit latch simultaneously with the second eight bit latch. This enable passes the four least significant bits to the four bit latch and transfers the eight upper bits from the first latch to the second. Therefore all 12 data bits appear at the inputs of the DAC at the same time producing a smooth transition of voltage [Ref. 5].

The AD7541 consists of an R-2R ladder and 12 CMOS current switches which perform the D/A conversion. Adding a 741 operational amplifier at $I_{out\ 1}$ and connecting the output of the 741 to $R_{Feedback}$ completed this simple, but very powerful, circuit.

The Software

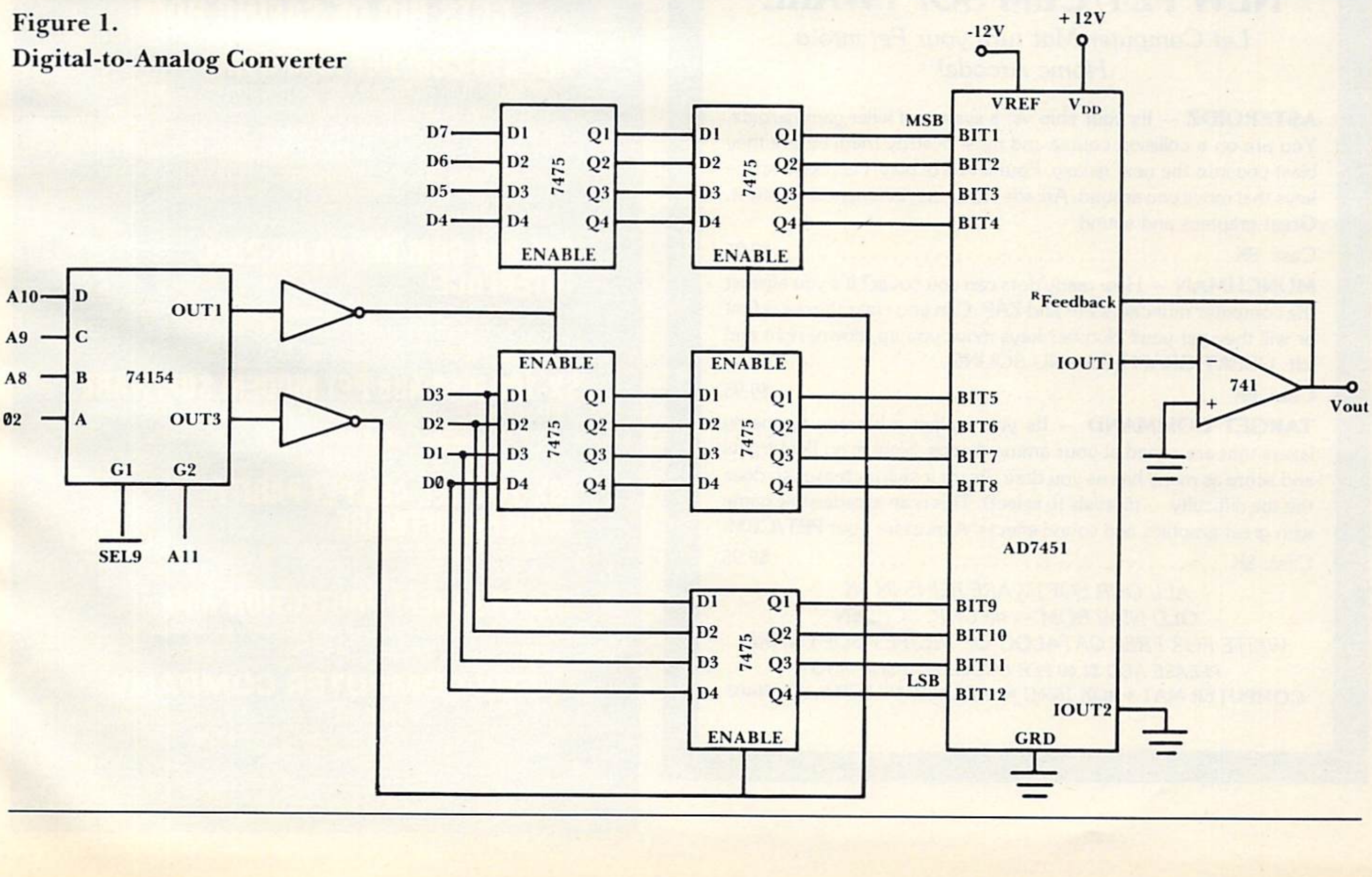
The following program allows the input of any number from 0 to 4095. The D/A outputs the corresponding voltage from 0 to 11.25 volts.

```
100 INPUT "V=";V
110 U=INT(V/16):REM-UPPER 8 BITS
120 L=V-16*U:REM-LOWER 4 BITS
130 POKE 36864,U:REM-HIGH 8 BITS
          GO TO $90XX
140 POKE37120,L:REM-LOW 4 BITS
          GO TO $91XX
150 GOTO 100
```

Additional Comments

After construction of the D/A interface, it was seen that double buffering the lower four bits rather than the upper eight bits of the data for the DAC would require one less chip and also would allow the use of the interface as an eight bit D/A. For use

Figure 1.
Digital-to-Analog Converter



as an eight bit D/A, a POKE command at the beginning of the program would set the lower four bits to 0 and then a single POKE would output the data to the upper eight bits.

Some difficulty has been encountered in finding connectors compatible to the PET Memory Expansion Port. The authors prefer the Relia-Tac connectors (Amphenol 221-1660 receptacle strips with 220-502 contacts) for this purpose.

The only modification required to use this circuit with other 6502 based machines is a change of the address decoder. Partially decoded address lines are not universally available, so one may need to use an additional 74154 to decode the high order four bits of the address bus. If the addresses $90XX and $91XX are occupied in your machine, you will need to change the decoder to an unused address.

It should be emphasized that the decision to decode eight address lines was rather arbitrary. The authors chose that figure because it is the maximum that could be decoded with a single 74154.

Finally, note that if you wish to implement additional analog outputs, more address decoders are unnecessary. Outputs 5, 7, 9, etc. of the 74154 as configured correspond to addresses $92XX, $93XX, $94XX, etc.

A high resolution analog output is straightforward to implement for the PET microcomputer. The circuit is easy to construct using readily available components. Using the Memory Expansion Port rather than the User Port allows one to add other analog outputs by merely changing the address assigned to the interface. The potential uses of the interface are restricted only by the imgination of the designer. For example, the availability of two analog outputs allows high resolution plotting via an X-Y recorder.

This circuit also demonstrates the ease with which the Memory Expansion Port can be utilized for interfacing. We share with Marvin L. DeJong the hope that there are other "hardware nuts" out there interested in such circuits.

References:

1. *CBM Professional Computer User Manual, Commodore Business Machines, Inc., Santa Clara, June 1979, pp. 82-83.*
2. *Analog Devices Spec Sheet for AD7541 CMOS 12-Bit Monolithic Multiplying DAC.*
3. *Signetics Data Book, Signetics Corporation, 1974, pp. 2-144-145.*
4. *Signetics Data Book, Signetics Corporation, 1974, pp. 2-76-77.*
5. *Short, Kenneth L., Microprocessors and Programmed Logic, Prentice-Hall, Inc., Englewood Cliffs, 1981, pp. 403-312.*

©

Renumber VIC-20 BASIC Lines The Easy Way

Charles H. Gould
Computer Science Department
Florida Institute of Technology

Until we have a ROM available, such as the Toolkit, for the VIC-20, here is a simple way to renumber lines in that crowded program you are writing. Simply type in the lines given below, and: RUN 10000. Better still, put this program on a separate tape, load it whenever you start a new program development; when complete, erase these lines. The program uses only 198 bytes, so 95% of your RAM is still there.

As shown, the renumbered lines start at line 10, and increment each line number by 10. *But* it does not change GOTO or GOSUB line numbers. You must manually change these after renumbering, and before running. An easy way to keep track of the GOTO and GOSUB terminal addresses is to insert "REM line #" at the end of each such statement, renumber, correct GOTO and GOSUB references to the new line numbers, and erase the REMs.

Line 9990 simply protects your program from entering the renumber routine until you want it to. In line 10010, Y7 is the starting line number, and in line 10050, the Y7 = Y7 + 10 sets the increment. Either can be changed. Line 10020 tests to see if we have renumbered up to, but not including, line 9990. Line 10030 changes the line number. Line 10040 searches for the next line. Y6 is the normal start of BASIC text (-1). The BASIC text lines are stored in RAM memory in this way: the first and second (LO and HI) bytes are a link to the next line; the third and fourth bytes are the line number (see line 10030 below); the BASIC statement is then given using tokens; the last byte of the line is a null (ASCII 0). After the last BASIC line, two more nulls are inserted to indicate end of program.

So, until we get support utility ROMs, let's make do.

```
9990 END
10000 REM RENUMBER
10010 Y6=4096: Y7=10
10020 IFPEEK(Y6+3)=6ANDPEEK(Y6+4)=39T
   HENEND
10030 Y8=INT(Y7/256):Y9=Y7-256*Y8:POK
   EY6+3,Y9:POKEY6+4,Y8
10040 IFPEEK(Y6+5)<>0THENY6=Y6+1:GOTO
   10040
10050 Y7=Y7+10:Y6=Y6+5:GOTO10020
```

©

Part II:

Atari Microsoft BASIC

Jerry White
Levittown, NY

In the first of this three-part series, I pointed out many of the strengths and weaknesses of Atari Microsoft BASIC (AMSB). In Part 2, I will make comparisons between the three versions of BASIC available to the Atari owner. To conserve space, the abbreviation 8K will be used for the standard Atari BASIC, and A+ will be used to indicate O.S.S. BASIC A+ (Optimized Systems Software).

It somehow seems unfair to include 8K in these comparisons. The others did not have to fit into an 8K ROM cartridge. The fact that 8K is on a ROM, and that it requires far less RAM than the others, is its greatest asset. Those who have less than 32K, or do not have a disk drive, cannot consider the others at this time. For those with at least 32K and one disk drive, AMSB and A+ can provide many useful features not found in 8K.

Sources at Atari tell me that a 16K AMSB ROM is in the planning stages, but will not be available until at least the end of this year.

One of the major arguments I've noticed when different versions of BASIC are compared, is the way strings are handled. Most Microsoft users cannot imagine how one could possibly live without string arrays. 8K and A+ users have the "unlimited length string." They have learned that this can certainly be an advantage, and that one long string can be manipulated to simulate string arrays.

For those who require more than a two dimensional array, AMSB is the only choice.

It would appear that compatibility would be the most important factor in choosing between AMSB and A+. While AMSB is not 100% compatible with other versions of Microsoft BASIC, converting from other versions of Microsoft to AMSB is not all that difficult. All the necessary information is provided in the rather large AMSB manual. It would certainly be easier to convert from one Microsoft BASIC to another, than to convert from Microsoft to Atari 8K or to A+.

On the other hand, A+ is compatible with 8K, and is a logical upgrade for current 8K users.

What is BASIC A+? A+ is a version of BASIC which is available for the Atari as well as other microcomputers. It is available from Optimized Systems Software and sells for $80.00. While A+ requires more RAM than Atari 8K, it uses about 3K less than AMSB. Overall, A+ is the fastest of the three.

A+ has all the features of 8K, many of those found in AMSB, and a few unique to itself. Among its unique commands are DIR which is used to list a disk directory on the screen. LVAR lists all variables currently in use to any device. DPEEK and DPOKE may be used to access or change double byte locations. HSTICK and VSTICK provide a simple method for reading horizontal and vertical joystick positions. BGET and BPUT are used to GET and PUT blocks of data very quickly. Using only these two commands, an entire screen may be placed into a string. Similarly, RGET and RPUT are used to input and output fixed length records.

A+ is the friendliest of all when it comes to using Player Missile Graphics (PMG). Briefly, the PMG command is used to turn off, or enable single, or double line resolution P/M Graphics. PMCLR is used to clear or erase players and/or missiles. PMCOLOR, PMWIDTH, PMADR, and PMMOVE statements are self-explanatory. The MISSILE command provides an easy way for a parent player to "shoot" a missile. The BUMP command is used to detect collisions.

A+ is still being improved. The folks at O.S.S. tell me that RENUM command will be available by the time this article is printed. A+ is efficiently written, well documented, and moderately priced.

AMSB has the most accurate math package and its own unique features. Some, such as multidimensional, alphanumeric arrays, have already been discussed. AMSB also has integer, single, and double precision numeric variables and constants, and hexadecimal constants.

AMSB has the most commands from which to choose. Three of its unique commands I found particularly interesting and useful were TIME$, VERIFY, and COMMON.

TIME is set and reset by setting TIME= JIFFIES. A jiffy is 1/60 of a second. Once TIME has been set, the current time in the format of HOURS:MINUTES:SECONDS is automatically stored as TIME$. A demonstration program using TIME$ as well as other AMSB features will be the subject of the third and final part of this series.

VERIFY is used to match the program in memory with a program on cassette or diskette. If the two programs are not identical, a TYPE MISMATCH ERROR will let you know.

COMMON provides the ability to make variables in two programs the same in value as well as in name. Chaining programs takes on an entirely new dimension when the COMMON command is available. ©

*An EPROM is an Erasable Programmable Read Only Memory. It is similar to the ROM chips inside the computer that hold BASIC, except that it can be erased, usually by exposing it to ultraviolet light for a period of time. It is useful if you have a program, such as Micromon (**COMPUTE!**, January, 1982, #20, pg. 160) which you use quite often. Instead of having to load it into the computer each time, it could be saved onto an EPROM, pushed into one of the empty sockets in your machine, and then used by a SYS to its starting location. This is sometimes called* firmware, *falling, as it does, between soft- and hardware.*

Review:

The "Branding Iron" EPROM Programmer For PET/CBM

David A. Hook
Barrie, Canada

The Branding Iron is a versatile EPROM burner for Pet 2001, CBM 4000 and 8000 series computers. The software provided functions with Original ROM, Upgrade ROM, or ROM 4.0 (both 40-column and 80-column).

Software and hardware allow programming of the three PET-compatible EPROMs: Intel 2716 or TI 2516 (both 2K) and TI 2532 (4K). The single, 5-volt, version is the correct type for the PET.

You may: program an EPROM, verify the contents of the EPROM just programmed, or copy an EPROM's contents into the PET's memory.

The Branding Iron consists of a 5"x 5"x 1" module and two connectors. One attaches to the User Port and the other to the 2nd Cassette Port. Ribbon cables are connected to the main unit. A 250 mA transformer provides the power and is to be plugged into a nearby AC socket.

The main module is attractively housed: no bare circuit board. Rubber feet should prevent scratches to your table. The 24-pin socket is not a "zero insertion-force" type. However, it has been gentle to the fragile IC-leads so far. There is a read/program switch and a status light to show programming activity.

The two cables are approximately 24 inches long. Both connectors are marked to show the "up" side. Neither included polarizing keys to prevent upside-down insertion. This 10-cent cost should not have been omitted. (The etched markings on my User Port connector were not painted).

Installation

Four typewritten pages of instructions are included. They indicate the location of the proper cassette port for each model PET/CBM. Most large keyboard, 40-column, machines make this connection inside the case. If your machine was upgraded from a built-in cassette, this port will be external.

I chose to plug in the transformer at a different wall socket than the PET.

Operation

The first program on the cassette provides several screens of instructions. The proper syntax and function for each command is shown.

You are then offered a choice of four programs, or a return to the start of the instructions. Then hit 'RETURN' and the tape will seek and load the chosen program. If you are working from disk, add ',8' to the 'LOAD' command on the screen.

The proper 'SYS' command has also been printed on the screen. Move the cursor here and hit 'RETURN'. The software has now been linked to the normal machine language monitor. Four extra commands are now available: T,P,V and C.

The documentation also gives the memory addresses used by each program. This is helpful – you won't accidentally use these yourself.

'T' Command: selects the type of EPROM. Defaults to a 2716 or 16K bit ('.T 16'). Enter '.T 32' for a 2532. No need to re-enter for each EPROM, unless you're switching types.

'P' Command: programs the EPROM from PET memory, given a start/end address in hexadecimal. Follow the manual directions for the 7-step procedure.

An address counter increments on the screen during programming. Any errors or defective memory locations are displayed on the PET's screen. Two minutes later it's done.

It isn't necessary to program the entire contents of the EPROM at any one session! This feature is quite significant. It is shown in the demo program, but not emphasized. Because of this, it is important to program from PET memory carefully.

If you want to store into the first byte of the EPROM, be sure to have this byte at a "round" hex address in the PET. (e.g. $2000, $3000, $4000, etc.).

Correspondingly, to program the fifth 256-byte "page" of the EPROM, you might enter: '.P 2400 24FF'. I have programmed as few as 21 bytes in a session.

'V' Command: compares the EPROM's contents with a given range of PET memory. It's done in an instant. Any discrepancies are reported, giving: memory address, PET memory contents, and EPROM contents at that address.

'C' Command: the EPROM contents are transferred to the user-designated range of PET memory. This too is almost instantaneous.

You may then examine or edit, using the normal PET monitor commands.

I've programmed both 2K and 4K EPROM's with this device. All features have worked flawlessly.

It's very versatile: all PET/CBM hardware, software for all PET ROMs and all compatible EPROMs. All this for a reasonable price.

With all the utilities now available for the PET, EPROMs to expand user-memory are a valuable enhancement. The PET's character-generator is another candidate for an EPROM.

Producer: K-Z Systems, Rt. 2, Box 473
Manchester, MO 63011
Availability: AB Computers, 115 E. Stump Road
Montgomeryville, PA 18936
Cost: $79, plus shipping

©

A look at the first VIC three-in-one program package from The Code Works, creators of the "magnetic magazines," CURSOR for PET and IRIDIS for Atari.

Review: VIXEL #1

Eric Giguere
Peace River, Canada

VIXEL #1, for VIC, is the first in a series of software packages published by The Code Works, following in the tradition of CURSOR for the PET and IRIDIS for the Atari. For $12.95 you receive three excellent programs that can keep you tied to your VIC for hours. Also included in the package is an informative booklet which gives you the instructions and explanations of the programs. Here is a brief summary:

FIRE!

The first program is called FIRE!. It is a game in which you pilot a helicopter and try to douse a skyscraper fire with the water in your tanks. Sounds easy, eh? Well, there is more to it than that. You only have three minutes to extinguish the fire or the people you are trying to save will die. Also, you may only hold 1000 gallons of water in your tanks at once and, when you run out, you have to waste precious time refilling the tanks, as the fire mercilessly continues to spread. It isn't easy to put out the fire, but it can be done. My record for putting it out was eight seconds, and, although I admit it was really a fluke, I challenge anyone to beat that!

The game itself is very creative. It uses no custom characters to form the helicopter or fire. Instead it uses the VIC's graphic symbols (with some creativity). It also uses a short machine language program to make the fires flicker, but, apart from that, there is nothing special about the program itself that any knowledgeable programmer couldn't figure out.

DRAW

One of my friends has an Apple, and the thing I admire most about it is its HI-RES graphics, where you can draw a line from any point on the screen to any other. Apparently, this feature will be incorporated in the Commodore *[a new VIC add-on, expected soon]*.

DRAW has partially solved that problem. This program allows you to draw in HI-RES within a 13x14 character-wide box in the middle of the screen. Using function keys and the standard VIXEL control keys (described in the instruction book), you can draw, erase, or move the cursor around at your pleasure. And, when you're finished drawing, you have the option of SAVEing the whole thing for later viewing.

Basically, what the program does is create custom characters "on the fly" using a machine language program. It creates a character which looks like part of your drawing. It is a good idea, exploited to the utmost, and I find it very useful as a way to get children interested in the VIC.

RACE

You control a race car and are trying to earn as many points as possible before the computer's car crashes into you. You pick up points by passing over dots on the track and, after having passed fifty dots, the computer's car starts laying down diamonds after it, with each diamond being worth five points. If you happen to collide with the computer's car, do not fear. You have two other cars in reserve (when these are used up the game ends), though you start over with a new screenful of dots.

RACE is divided into two programs: the first, called RACE.CHSET (for RACE CHaracter SET), creates the special characters used by the game and then automatically loads the second program, the game itself. RACE is very enjoyable; a good finale to VIXEL#1.

The Cover

I neglected to mention that, at the start of the tape, there is actually a fourth program called COVER. This program uses the custom characters to create a likeness of the VIXEL mascot and move him across the screen, chalking out the words "VIXEL #1," tapping his feet and blinking his eyes all at the same time.

It shows the VIC owner how he can use the VIC's ability to create custom characters for his own special purposes. After this demonstration, the user is prompted to press the SPACE bar, after which the program lists the programs available on that particular tape. All the user has to do now is LOAD the program of his or her choice.

In my view, VIXEL #1 is a good investment for the VIC-20 owner. I feel that the programs need only one major improvement: that The Code Works change the selection of keys used to control the programs. Perhaps they use the I,J,K, and M keys instead of the present W,A,D, and X. I always manage to hit the SHIFT LOCK key instead of the A key, and this is rather frustrating as it prevents the other keys from being read. It can cause problems when you are playing RACE.

Apart from this minor problem (probably due to the fact that these programs are mainly made for use with a joystick) VIXEL #1 scores an A+ with me, and I can hardly wait for the release of VIXEL #2.

VIXEL #1, The Code Works, Box 550, Goleta, CA, 93116. $12.95. For VIC-20 with tape drive.

©

CAPUTE!:
Corrections And Amplifications

1. "VIC: Alternate Screens," *Home and Educational Computing!*, Fall, 1981, pg. 14: change line 500 POKE 648,S:POKE 36866,T

2. "Discovering Atari's 'Hidden Graphics'," **COMPUTE!**, December, 1981, #19, pg. 98: in the first program, change line 50 SE. X,RC,6 and, in the second program, change line 110 IF X=219 THEN Z=97:X=0:Y=4:G.30 and line 120 IF Z=123 THEN Z=225: X=0:Y=6:G.30

3. "SuperFont," **COMPUTE!**, January, 1982, #20, pg. 110: Mr. Brannon has made the following improvements to his program. Change line 1110 FOR I=0 TO 7: A=INT(PEEK(CHSET+C*8+I)/2) and change line 1320 Z=0: FOR I=0 TO 2: FOR J=0 TO 1+(I>0): A=PEEK(CHSET+C*8+Z): Z=Z+1

4. "MICROMON," **COMPUTE!**, January, 1982, #20, pg. 160: the author sent in one additional change (to those listed in the article which adapt the program to the 8032's screen). Change $XD18 from $08 to $10. Our thanks also to John Stout for mentioning this correction. In addition, several readers have mentioned problems in attempting to relocate the main program up to $6000. The transfer memory function must be used first to move the program to $6000. Both New Locator functions attempt to resolve all internal JMP's and JSR's which are not (after Transfer) correct any more. Nevertheless, the moved code should be disassembled to find any remaining out-of-range references.

5. "Named GOSUB With Variable Passing," **COMPUTE!**, February, 1982, #21, pg. 69: two lines should be changed. Change 270 K=INT(10*RND(1)): L=INT(10*RND(1)) and also change line 6000 DATA 169,10,133,104,169,0,141,0,10,32,75,214,169,76,141,245,1688.

6. "Timekeeping," **COMPUTE!**, February, 1982, #21, pg. 173: On page 173, replace B1 with B$ and, in the program below, replace the *60 in line 120 with *3600. On page 174, replace the 902 in line 775 with 920. ©

New Products

Educational Aids In The Classroom

Educational Activities, Inc. announces the availability of their new diagnostic, prescriptive, tutorial reading program, DIASCRIPTIVE READING. It is designed to aid the reading teacher, eliminate paper work, and provide individualized instruction for each student.

The program, which was successfully classroom tested at Lindenhurst Junior High School, Lindenhurst, New York, covers six skill areas – vocabulary, sequence, main idea, fact/opinion, details, and inference.

DIASCRIPTIVE READING is contained on 7 diskettes for TRS-80, Apple, and PET or on 15 cassettes for TRS-80 or PET. The price for either format is $245.00.

Educational Activities, Inc.
Microcomputer Department
P.O. Box 392
Freeport, NY 11520

* * * * * * *

The company has also announced the immediate availability of two new sound filmstrip programs developed to meet the need for computer literacy and computer career awareness in schools.

Computer Programming – BASIC for Elementary Grades is designed to instruct elementary age children in computer programming in the BASIC language. All the conceptual presentations have been pretested successfully on a broad range of youngsters. Piloted in a special computer programming course held for elementary students, the principles and applications of the major BASIC commands and statements proved to be well within the capabilities of the average middle-to-upper elementary age child.

The second filmstrip series, *Careers in Computers,* informs students about the many job opportunities in the high growth computer industry. The program is geared toward high school students and young adults who need to know what types of jobs are available, educational requirements, experience, and personal characteristics demanded by the job, as well as general salary descriptions, and possibility of advancement. This program not only discusses career opportunities, but is also a good introduction to the computer industry.

Computer Programming – BASIC for Elementary Grades consists of 5 full-color filmstrips, 5 audio cassettes, and a teacher's handbook. The price is $97.00. *Careers in Computers* consists of 4 full color filmstrips, 4 audio cassettes, and a teacher's guide. The price is $79.00.

For further information call, toll free, 800-645-3739 (outside New York State) or 516-223-4666 (inside New York State) or direct your inquiry to:

Educational Activities, Inc.
P.O. Box 392
Freeport, NY 11520

Understanding Your VIC From TIS

Los Alamos, NM – Total Information Services, Inc. (TIS), a publisher of books and software for Commodore and Ohio Scientific computers, announces a book for the new Commodore VIC-20. *Understanding Your VIC Volumn 1: Basic Programming* is a 148 page, 8½" by 11" softcovered book that uses a step-by-step approach to help the beginner quickly and easily learn about the VIC. The book is full of exercises and examples. Many exercises show the expected results so that the reader has immediate feedback on errors. To start the beginning programmer off on the right foot, a chapter on program design describes the use of psuedo code and data dictionaries to refine programming problems. There are chapters on the VIC color and sound features that use these program design techniques to build demonstration programs. This easy-to-use, learn-by-doing book is available from Commodore dealers or directly from TIS, Box 921, Los Alamos, NM 87544. Retail price is $11.95 plus $2.00 for first class shipping and handling. The color and sound demonstration programs are both available on a single cassette for $7.95 plus $1.00 for shipping.

VisiSchedule Program Extends Product Line From VisiCorp

VisiCorp (formerly Personal Software Inc.) announced the VisiSchedule program. The new program increases to nine the number of interrelated personal computer software products in the VisiSeries™ family.

The VisiSchedule program allows the user to create an on-screen schedule of projects and tasks. The displayed schedule

ATARI HOME COMPUTERS

ATARI 800

16K ... $679
32K ... $749
48K ... $819

ATARI 400

16K.... $329
32K.... $478
48K.... $555

410 Recorder	$76.00
810 Disc Drive	$449.00
822 Printer	$269.00
825 Printer	$629.00
830 Modem	$159.00
820 Printer	$269.00
850 Interface	$169.00
New DOS 2 System	$29.00

PACKAGES

481 Entertainer	$83.00
482 Educator	$130.00
483 Programmer	$57.00
484 Communicator	$344.00

ATARI HOME COMPUTER PROGRAMS

Home Office	
CX404 ATARI Word Processor	$119.00
CX8102 Calculator	$29.00
CX412 Dow Jones Investment Evaluator	$99.00
CX4109 Graph It, Joystick optional	$17.00
CX4104 Mailing List	$20.00
CX4115 Mortgage & Loan Analysis	$13.00
CX406 Personal Financial Management System	$59.00
CX4103 Statistics 1	$20.00
CX8107 Stock Analysis	$20.00
CXL4015 TeleLink 1	$23.00
Home Study	
CX4101 An Invitation to Programming 1	$20.00
CX4106 An Invitation to Programming 2	$23.00
CX4107 Biorhythm	$13.00
Conversational Languages (ea.)	$46.00
CX4121 Energy Czar	$13.00
CX4114 European Countries & Capitals	$13.00
CX4108 Hangman, Joystick optional	$13.00
CX4102 Kingdom	$13.00
CXL4007 Music Composer	$47.00
CX4123 Scram, uses joystick	$20.00
CX4112 States & Capitals	$13.00
CX4110 Touch Typing	$20.00
Home Entertainment	
CXL4013 Asteroids	$35.00
CXL4004 Basketball	$27.00
CX4105 Blackjack	$13.00
CXL4009 Computer Chess	$33.00
CXL4012 Missile Command	$35.00
CXL4008 Space Invaders	$35.00
CXL4011 Star Raiders	$42.00
CXL4006 Super Breakout	$33.00
CXL4010 3-D Tic-Tac-Toe	$27.00
CXL4005 Video Easel	$24.00
Programming Languages and Aids	
CXL4003 Assembler Editor	$47.00
CXL4002 ATARI BASIC	$47.00
CX8126 ATARI Microsoft BASIC	$70.00
CXL4018 PILOT	$72.00
CX405 PILOT	$105.00

CX30 Paddle	$18.00
CX40 Joy Stick	$18.00
CX853 16K RAM	$89.00
Microtek 16K RAM	$75.00
Microtek 32K RAM	$159.00
Ramdisk (128K)	$539.00
One year extended warranty	**$50.00**

THIRD PARTY PROGRAMS

ATARI Program Exchange:

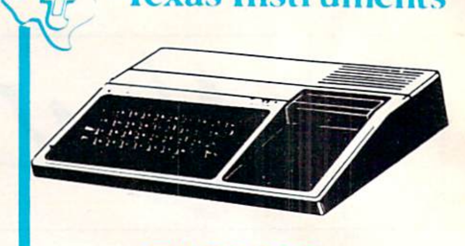
Eastern Front '41	$25.50
Avalanche	$15.50
Outlaw	$15.50
747 Landing Simulation	$15.50
Babel	$15.50
Dog Daze	$15.50
Downhill	$15.50
Attack!	$15.50
Blackjack-Casino	$15.50
Reversi II	$15.50
Domination	$15.50
Solitare	$15.50
Disk Fixer	$15.50
Supersort	$15.50
Data Management	$15.50
Chameleon	$15.50
Instedit	$15.50
Insomnia	$15.50
My First Alphabet	$25.50
Mapware	$18.00
Number Blast	$11.50
Family Cash Flow	$15.50
Weekly Planner	$15.50
Bowler's Data Base	$13.00
Banner Generator	$11.50

Visicalc	$169.00
Letterperfect (Word Processor)	$109.00
Ricochet	$14.50
Crush Crumble & Chomp (cassette or disk)	$24.00
Star Warrior (cassette or disk)	$29.00
Rescue at Rigel (cassette or disk)	$24.00
Datestones (cassette or disk)	$16.00
Invasion Orion (cassette or disk)	$18.50
Mission Asteroid	$22.00
MouskATTACK	$31.00
The Next Step	$34.00
Softporn	$27.00
Wizzard & Princess	$29.00
K-BYTE Krazy Shoot Out (ROM)	$39.00
Protector (Disk 32K)	$32.00
Jaw Breaker (on line disk)	$27.00
Ghost Hunter (cassette)	$24.00
Ghost Hunter (disk)	$30.00

COMING SOON

PAC MAN (May) $35, CENTIPEDE (June) $35
CAVERNS OF MARS (April) $32

Texas Instruments



TI-99/4A $359

PHP 1600 Telephone Coupler	$179.00
PHP 1700 RS-232 Accessories Interface	$179.00
PHP 1800 Disk Drive Controller	$239.00
PHP 1850 Disk Memory Drive	$389.00
PHP 2200 Memory Expansion (32K RAM)	$319.00
PHA 2100 R.F. Modulator	$43.00
PHP 1100 Wired Remote Controllers(Pair)	$31.00
32K Expansion	$329.00
PHP Printer Solid State	$319.00
PHM 3006 Home Financial Decisions	$26.00
PHM 3013 Personal Record Keeping	$43.00
PHD 5001 Mailing List	$60.00
PHD 5021 Checkbook Manager	$18.00
PHM 3008 Video Chess	$60.00
PHM 3010 Physical Fitness	$26.00
PHM 3009 Football	$26.00
PHM 3018 Video Games I	$26.00
PHM 3024 Indoor Soccer	$26.00
PHM 3025 Mind Challengers	$22.00
PHM 3031 The Attack	$35.00
PHM 3032 Blasto	$22.00
PHM 3033 Blackjack and Poker	$22.00
PHM 3034 Hustle	$22.00
PHM 3036 Zero Zap	$18.00
PHM 3037 Hangman	$18.00
PHM 3038 Connect Four	$18.00
PHM 3039 Yahtzee	$22.00

PRINTERS

Centronics 739-3	$619.00
Centronics 739-1	$519.00
Diablo 630 Special	$1799.00
Epson	
MX70	$359.00
MX80	$469.00
MX80FT	Call
MX100	Call
NEC	
8023	$549.00
7730	Call
7720	Call
7710	Call
Okidata	
82A	$499.00
83A	$769.00
84	$1129.00
Citoh Starwriter	
F10-40 CPS	$1469.00
F10-55 CPS	Call
Paper Tiger	
445G	$699.00
460G	$899.00
560G	$1129.00
Talley	
8024-7	$1399.00
8024-L	$1629.00

HOW TO ORDER: Phone orders invited or send check or money order and receive free shipping in the continental United States. PA and NEV. add sales tax.

computer mail order west

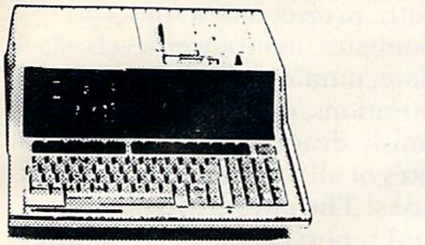
800-648-3351

Franco
Hables
Espanol

IN NEVADA, CALL (702) 588-5654

P.O. BOX 6689, STATE LINE, NEVADA 89449

HEWLETT PACKARD



HP•85 $1999

80 Column Printer	$799.00
HP•125	$1999.00
HP•83	$1699.00
HP•85 16K Memory Module	$169.00
5¼" Dual Master Disc Drive	$1929.00
Graphics Plotter (7225B)	$2079.00
NEW! HP•87	$1999.00
Hard Disk w/Floppy	$4349.00
Hard Disk	$3440.00
"Sweet Lips" Plotter	$1149.00

HP41CV Calculator $259

41 C	$189.00
11 C	$119.00
12 C	$129.00
34 C	$117.00
38 C	$119.00
HP•41 Printer	$340.00
HPIL CALCULATOR PERIPHERALS	
IL Modual	$104.00
Digital Cassette	$449.00
Printer/Plotter	$419.00
Card Reader	$164.00
Optical Wand	$99.00

CALL FOR SOFTWARE INFORMATION

XEROX®

Xerox 820	
System I 5¼"	$2450.00
System II 8"	$2950.00
CPM 5¼"	$169.00
Word Processing	$429.00
Super Calc	$269.00

CALL FOR MORE INFORMATION

commodore BUSINESS MACHINES

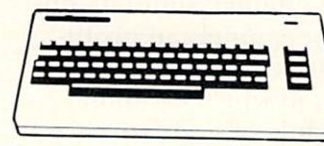
CBM 8032

$1069

SOFTWARE

Word Pro 5 Plus	$319.00
WordPro3 Plus	$199.00
WordPro4 Plus	$299.00
Commodore Tax Package	$589.00
Visicalc	$169.00
Medical Billing	$449.00
The Source	$89.00
OZZ Information System	$289.00
Dow Jones Portfolio	$129.00
Pascal	$239.00
Legal Time Accounting	$449.00
Word Craft 80	$289.00
Power	$79.00
Socket-2-Me	$20.00
Jinsam	$Call
MAGIS	$ Call
The Manager	$209.00
Softrom	$129.00
Real Estate Package	$799.00
BPI Inventory Control	$319.00
BPI Job Costing	$319.00
BPI Payroll	$319.00
BPI General Ledger	$329.00
Creative I SAM	$79.00
Creative General Ledger	$229.00
Creative Accounts Receivable	$229.00
Creative Inventory	$229.00

4032	$969.00
4016	$769.00
8096 Upgrade Kit	$399.00
Super Pet	$1599.00
2031	$529.00
8050	$1299.00
4040	$969.00
8300 (Letter Quality)	$1799.00
8023	$769.00
4022	$599.00
Pet to IEEE Cable	$37.00
IEEE to IEEE Cable	$46.00
Tractor Feed for 8300	$240.00
8010 Modem	$229.00



VIC 20 $259

COMPLETE

Vic 6 Pack Program	$44.00
VIC1530 Commodore Datassette	$69.00
VIC1540 Disk Drive	$499.00
VIC1515 VIC Graphic Printer	$339.00
VIC1210 3K Memory Expander	$32.00
VIC1110 8K Memory Expander	$53.00
VIC1011 RS232C Terminal Interface	$43.00
VIC1112 VIC IEEE 488 Interface	$86.00
VIC1211 VIC 20 Super Expander	$53.00
VT232 VICTerm I Terminal Emulator	$9.00

VIC1212 Programmers Aid Cartridge	$45.00
VIC1213 VICMON Machine Language Monitor	$45.00
VIC1901 VIC AVENGERS	$23.00
VIC1904 SUPERSLOT	$23.00
VIC1906 SUPER ALIEN	$19.00
VIC1907 SUPER LANDER	$23.00
VIC1908 DRAW POKER	$23.00
VIC1909 MIDNIGHT DRIVE	$23.00
VT106A Recreation Pack A	$44.00
VT107A Home Calculation Pack A	$44.00
VT164 Programmable Character/Gamegraphics	$12.00

New VIC Software

Household Finance	$27.00
VIC Games	$19.00
VIC Home Inventory	$13.00
VIC Rec/Ed II	$13.00
VL101 Introduction to Computing	$19.00
VL102 Introduction to BASIC Programming	$19.00
VM110 VIC 20 Programmers Reference Guide	$15.00

Terminal	$13.00
Un Word	$13.00
Grafix Menagerie	$11.00
VIC PICS	$15.00
Ticker Tape	$13.00
Banner Headliner	$13.00
RS 232	$39.00

Modems

Novation Auto	$239.00
D Cat	$169.00
Cat	$159.00
Hayes	
Smart	$239.00
Livermore Star	$119.00

Terminals

Televideo	
910	$579.00
912C	$699.00
920C	$749.00
950	$939.00
Call for computers	
Zenith Z19	$749.00
Adds	$549.00

Monitors

Amdex 12" B&W	$139.00
12" Green	$149.00
13" Color	$349.00
Sanyo 12" B&W	$259.00
12" Green	$269.00
13" Color	$449.00
TI 10" Color	$349.00

Above are cash prices, add 3% for Master Card and Visa purchases

computer mail order east

800-233-8950

IN PA. CALL (717) 327-9575

477 E. THIRD ST., WILLIAMSPORT, PA 17701

Patricio Hables Espanol

responds immediately to "What if?" schedule detail changes and shows the effect of the change on all subsequent events. The Visi-Schedule program immediately updates its information on-screen when a change occurs; and it incorporates all pertinent data including costs, manpower, resource leveling and scheduling constraints.

The timechart and other reports produced by the Visi-Schedule program provide valuable information to any business involved in planning projects and managing resources. Managers, executives, engineers and manufacturing personnel can profitably use VisiSchedule to track any project in which deadlines and a schedule of events are important.

The program has been designed for ease of use. With its "moving cursor menu," the user need only point with the cursor to each command. The user is then guided through each step, developing the project as it is being entered; no preplanning is necessary.

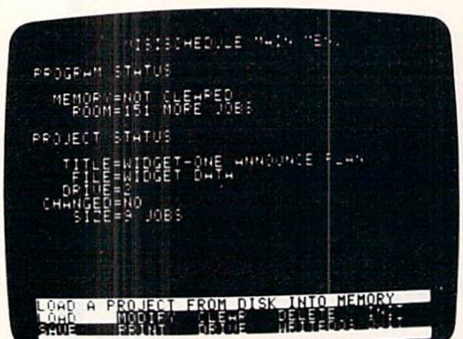
Like other members of the VisiSeries product family, the VisiSchedule program may send data to the VisiCalc, VisiTrend/Plot or VisiFile programs.

A calendar representation of start and stop dates, slack time, holidays, and deadlines for up to 160 different tasks may be displayed and automatically printed out. Many different computer-generated reports in an almost unlimited combination of information may be produced. Such information includes critical path, project milestones, cost estimates, manpower levels, slack time, number of successors, durations, earliest start dates, late finish, deadlines and prerequisites of all or some of the project tasks. The interactive time chart and reports allow a user to investigate the tradeoffs among manpower, dollars and time.

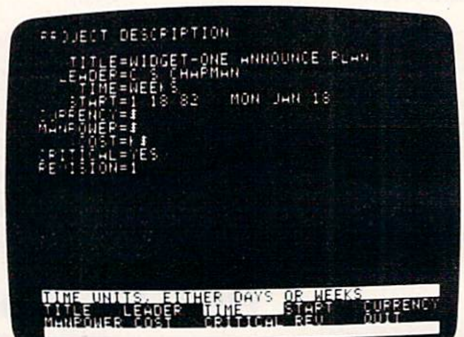
The VisiSchedule program requires an Apple II or II Plus computer with a minimum of 48K memory and two disk drives. A printer is strongly recommended. A 48K computer can handle up to a 50-task project; adding the Apple Language System or Microsoft RAMcard increases this capacity to a 160-task project.

Price and Availability: Suggested US price of VisiSchedule is $300. It has been available since mid-February.

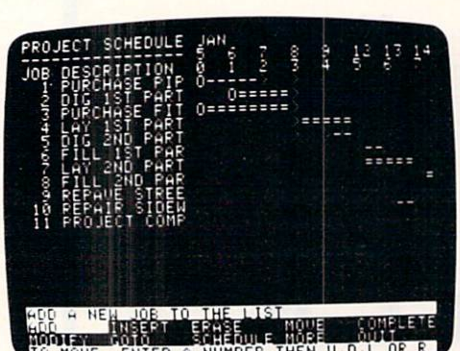
VisiCorp
2895 Zanker Rd.
San Jose, CA 95134
408-946-9000



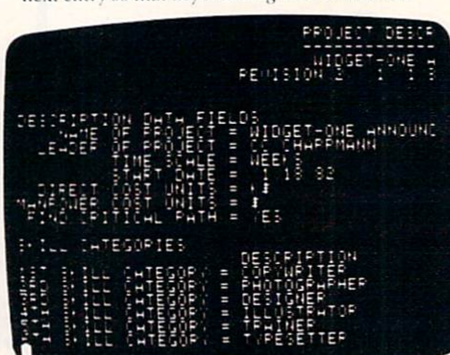
The VisiSchedule™ program automatically loads details from the previous use of the program. The user can re-load a project schedule already set up, or create one from scratch. Current date, person preparing report, current status of last project worked on can be checked.



The Modify Menu allows the user to create or change the overall description of the project. Such items as title of project, leader, currency, cost levels and revision number can all be changed. Modify is where the user sets up manpower, skills, costs, holidays and the work week. The moving cursor anticipates the next entry so that keyboarding time is reduced.



This displays the work schedule itself. The user may scroll in all four directions to view the entire project. Simple menu commands add, delete or rearrange jobs. As each job is entered, it is displayed on screen. Once jobs are entered, manpower peaks may be auto leveled, costs and manpower levels are displayed or the project schedule adjusted in many ways.



The VisiSchedule program prints a wide range of useful reports, including summary of the entire project, critical path, project milestones, manpower and cost estimates, job descriptions, tabular reports and sorted reports. The reports stay current because you can complete jobs, slip the project or make any changes to show what is really happening.

Use Graphics And Sound For Blackjack Program

Recreational Software Services announces development of a new interactive BLACKJACK program especially developed for the COMMODORE VIC-20. Uniquely keyed to the VIC-20's extended graphic and sound capabilities, and adapted for efficient play and "card-counting" – develop your own system to "beat the dealer". Available on cassette with full instructions, ready to LOAD and RUN. $7.95 plus $1.00 shipping and handling to Recreational Software Services, 38 Franklin Street, Northampton, MA 01060. Other recreational software available – write for free catalog.

T.M.

Oryx software

What does the Oryx, an African antelope, have in common with reliable service? They are both almost extinct.

Oryx Software wants to bring trustworthy service back to life. We're here to stay!

GUARANTEED LOWEST PRICES!
To celebrate our opening, we are matching any advertised price. Just show us the ad.

ORDER TOLL FREE - Outside WI
1-800-826-1589

SOFTWARE

Graphpower	299.00
Tax Preparer '82	127.00
Real Estate Analyzer	127.00
Creative Financing	127.00
Word Processor 1	425.00
Financial Projection	550.00
Mail Management	250.00
Micropro Wordstar	248.00
Mailmerge	90.00
Spellstar	169.00
Supersort II	159.00
Context Connector	180.00
Easywriter (Pro)	195.00
Easymover	45.00
Tellstar II	70.00
Easymailer	160.00
Datadex	249.00
Microplan Basic	419.00
Executive Secretary	212.00
Nevada CoBol	129.00

ACCESSORIES

Apple II Dual Thermometer w/software	260.00
Mountain Expansion Chassis	625.00
CPS Multifunction	199.00
Mountain Music System	395.00
100,000 Day Clock	375.00
Mountain Clock	252.00
Supertalker SD 2000	199.00
A/D + D/A	299.00
CCS 12K ROM/PROM Board	99.00
CCS A/D Converter Board	99.00
CCS Serial Asynch. Board	139.00
CCS Calendar/Clock Board	99.00
VOTRAX Type-n-Talk	375.00
ADALAB Package	495.00
DOS Dater	49.00
ABT Numeric Keypad	125.00
ABT Softkey	150.00
ABT Barwand	195.00
ABT Cash Box	295.00

. . . AND MANY MORE!

Call for price information:

Printers - Epson, Okidata, Microline, NEC, C. ITOH, Qume, Diablo, Anadex, IDS, MPI, Microtek, Printek.

Monitors - Amdek, Sanyo, Hitachi, NEC, Kaga/TSK.

Modems - Novation, Hayes.

Disk Drives - Percom, Corvus, Morrow, CCS, etc.

Plotters - Watanabe.

Quoted prices valid for stock on hand and subject to change without prior notice.

Ask for our **FREE** catalog.

Please:
- Wisconsin residents - add 4% sales tax
- Foreign orders - please add 15%
- Add $2.50 for shipping per software and small item. Call regarding others.

We Welcome:
- Visa, MasterCharge - (Add 4%)
- Checks (Allow 1-2 weeks for clearing)
- COD (Add 4%)

TOLL FREE - Outside Wisconsin:
1-800-826-1589

For Technical Info & in Wisconsin:
715-848-2322

Oryx Software
P.O. Box 1961 • Wausau, WI 54401

TAKE A BYTE OUT OF YOUR SOFTWARE COST!

SAVE 20% OFF RETAIL

JOIN SOFTWARE GALORE!*
THE FASTEST GROWING MEMBERSHIP BUYING SERVICE FOR APPLE, TRS-80, ATARI, PET, IBM, CP/M SOFTWARE.

*Member: Better Business Bureau

We would like to give you an opportunity to come on board and become a **MEMBER** of **SOFTWARE GALORE.** The largest national **SOFTWARE BUYING SERVICE.** For a limited time only, **introductory** memberships are available to you at **50% off.** Now only **$7.50** (Reg. $15.00 value) and will be good for the period of one year. **Only members may buy from SOFTWARE GALORE.**

You'll find **we stock** the latest releases as well as older hard to find software products that most computer stores no longer carry. We feature these manufacturers:

Accent Software
Acorn Software
Activision
Avant-garde Creations
Adventure International
Arcade Plus
Astar International Co.
Automated Simulations
Avalon Hill Game Co.
Beagle Brothers
Big Five Software
Broderbund Software
Budgeco
California Pacific
Cavalier Computer
Computer Magic Ltd.
Context
Continental Software
Dankin 5 / Level 10
Datamost
Datasoft
Delta Software
Denver Software
Edu-ware
Edu-ware East
Gebelli
Glostronics
Hayden Software
Highlands Computer Service
Howard Software
Infocom
Innovative Design Software
ISA
IUS
Lazer Systems
LJK Enterprises
Micro/Lab International
Microsoft
Muse
On-Line Systems
Optimized Systems Software Inc.
Personal Software
Piccadilly
Phoenix Software
Quality Software
Riverbank Software
Sensible Software
Sentient Software
Sierra Software
Sirius Software
Sir-Tech Software
Softape
Software Emporium
Software Publishing Corp.
Spectrum Computer
Stoneware
Strategic Simulations
Sublogic
Synergistic Software
TG Products
The Book Company
The Cornsoft Group
The Logical Choice
Turnkey Software
United Software of America
Versa Computing
Videx Inc.
Voyager Software

Other "Discount" outlets "Blowout" a few items to get your business then charge you 100% list price for everything else.

At **SOFTWARE GALORE,** you'll save a full **20% OFF RETAIL** on everything you buy and be entitled to these additional services:

- An Additional 5% off on your first single item purchase.
- Access to the largest inventory of software!
- Low handling/shipping charges.
- Overnight shipments by **U.P.S.**
- 90 day guarantee on all software.
- Personalized, efficient service.

Software Galore backs its commitment of service and quality to its members with a 90 day unconditional warranty. You'll never get stuck with a lemon from us.

Catalogs and inquiries call our **Toll Free Numbers:**
1-800-772-2666 x 989 (in Calif.) 1-800-423-6326 (direct national)

FOR IMMEDIATE ORDERS CALL 213-827-1851

Enclosed is ☐ 7.50 (for 1 yr.) ☐ 20.00 (for 2 yrs.) sign me up immediately and send a catalog, membership pack and order forms.

☐ Please send me a catalog and application only. I am under no obligation to buy.

Name: ☐ Mr. ☐ Ms.________________

Street:________________

City:________ State:________ Zip:________

Type of Computer Used:________________

Mail to: SOFTWARE GALORE P.O. Box 10005, Marina del Rey, CA 90291
Or Call: 1-800-772-2666 x 989 (in Calif.) 1-800-432-6326 (direct national)

Dial/Data Gives Access To Financial Markets

Remote Computing Corporation, who developed Dial/Data – a new software program that will give microcomputer users immediate access to financial markets, is located at:

1044 Northern Blvd.
Rosylyn, NY 11576
(516)484-4545

Accounting Software For The Apple III

EASy™ (Executive Accounting System) for Apple III is now being shipped to computer stores nationwide. It is the latest addition to a line of business software products developed by The Denver Software Company.

EASy™ is for the Apple III personal computer using floppy diskettes. It is an integrated general ledger, accounts receivable, and accounts payable system for medium-sized businesses. It meets the accounting needs of business managers, department heads, professionals, consultants, and retailers, EASy™ is designed for people with no computer experience and limited accounting experience.

EASy™ generates detailed financial and management reports in a clear format. It also prints invoices, statements, and checks. It is entirely menu driven, user-friendly, and expandable. Add-on modules will be available for inventory, word processing, budgeting, graphic analysis, and payroll. It's upgradable. EASy™ will be available for the Apple III Profile hard disk system.

Denver Software Company
14100 East Jewell Ave.
Suite 15
Aurora, CO 80012
(303)750-9980

Multi-purpose Interface For CBMs

Teaching Tools: Microcomputer Services announced in our February 1982 issue, a new multi-purpose interface for PET/CBM computers providing:

1) video monitor connector
2) sound adaptor
3) an audio tape recorder control

The price of the interface is $109.95; for the interface and RF modulator, $149.95.

Contact them for more details:

Teaching Tools:
Microcomputer Services
P.O. Box 50065
Palo Alto, CA 94303
(415)493-3477

SSM Introduces Four Function Serial/Parallel Interface Board For The Apple II

SSM Microcomputer Products Inc., a manufacturer of board level products for the Apple II, has introduced an enhanced version of its AIO serial/parallel interface board.

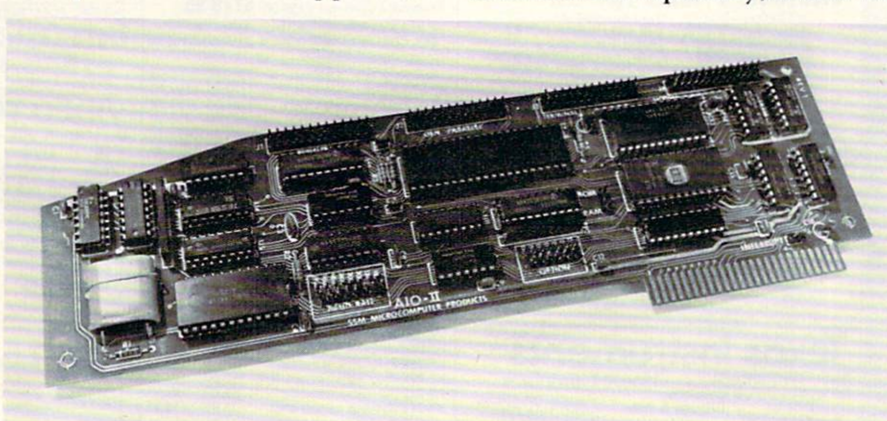
The new version, designated the AIO-II, delivers significantly improved performance, and combines several new features. This highly flexible, full function serial/parallel interface eliminates the need for any other I/O boards by combining two boards into one compact unit.

Among the enhancements are the AIO-II's ability to perform four independent interface functions, including serial modem, serial terminal/printer, parallel Centronics-compatible printer, and a general purpose parallel port.

The AIO-II permits simultaneous output to both one serial and one parallel device using the standard Apple control code protocols. Additionally, the AIO-II's advanced design techniques eliminate the need for "phantom" slot assignments, special set-up requirements, or hardware modifications.

The novice user can easily install the AIO-II with jumper options, while more experienced users may override the jumper set-up with control codes identical to Apple parallel card connections. All operations are transparent, requiring no additional user-written software.

The AIO-II's on-board firmware provides all necessary drivers, including terminal communications capability, and it will operate in any Apple II expansion slot (except slot 0). Optional drivers are available which will support Apple Pascal and Microsoft Softcard™.



The AIO-II package includes manual, jumpers and wiring information to support a wide variety of printers, including Epson, Anadex, Centronics, IDS, Okadata, NEC, Diablo, Qume, and more.

SSM Microcomputer Products, Inc.
2190 Paragon Dr.
San Jose, CA 95131
(408)946-7400

The Source Offers New Services For Investors

The Source, a nationwide electronic information service for individuals, businesses and organizations, has added two major new data bases to assist investors:

Media General Financial Service – with comprehensive, detailed background information on more than 3,100 stocks, updated weekly.

Commodity News Service – which tracks trends, news and prices of all major commodity markets and futures exchanges.

Source Telecomputing Corp.
1616 Anderson Rd.
McLean, VA 22102

Communication And Cataloging Now For CBM

HESCOM is a machine language program that can transfer data and programs between two PETs, two VICs or a PET and a VIC. You can load into the PET/CBM a program from a disk and transfer it to the VIC at 7000 bytes per second – three times the speed of the disk. After modifying the program on the VIC, you can send it back to the PET for saving to a disk or listing on a printer. HESCOM subroutines can also be called in programs to transfer single or whole blocks of memory between two machines. Thus, you can use an existing disassembler for the PET and with one change, disassemble the ROMs in the VIC. Use the VIC as a peripheral to the PET – a program running on the PET could display hi-res color graphics on the VIC, produce four-voice VIC sound or even get input from joysticks connected to the VIC. Price is $49.95 on tape, $52.95 on disk, plus $2 for postage.

HESCAT is a complete and fast diskette cataloging system for a PET/CBM, comprised of five programs in BASIC and machine language. You can catalog diskettes almost as fast as you can insert them. Use HESCAT to organize your diskette library. Using a full or partial

APPLE • ALTOS • ATARI • MAXELL • DYSAN • EPSON • CCS • SHARP • CASIO • HP • VERBATIM • MEMOREX • SOROC • CORVUS • ADDS

MICRO BUSINESS WORLD INC.

Information Line (213) 996-2252
TOLL FREE MAIL ORDER LINES
(800) 423-5886 Outside Calif.

TI-99/4A home computer

$355.00
MSL 649.95
Save 294.95

NEC MODEL 8023A PRINTER

$499.00
MSL 745.00
Save 246.00

SHARP BUSINESS COMPUTER 64K SYSTEM

Complete System Including Printer

- 5.25″ diskette
- 284K byte per disk drive
- 2 disk drives per unit
- Expandable up to 4 units
- Including SHARP's FDOS (Floppy
- Disk Operating System)

MSL 5995.00 OUR PRICE $3995.00 Save $2000.00

HEWLETT PACKARD

	List	Our Price
HP-125 Microcomputer	3750.00	2750.00
HP-85 Microcomputer	2750.00	1995.00
HP-83 Microcomputer	2250.00	1777.00
16K Exp-Memory Module	295.00	255.00
Graphics Plotter 7225	2450.00	2075.00
Personality Mod. For 7225	750.00	665.00
2631B Impact/Printer/ Hvy Dty	3950.00	3250.00
Option 020 For 2631B	150.00	125.00
8 Drives to Choose from 82902S	1300.00	1125.00
9895A 8″ Dual Drive	6850.00	5500.00
Graphics Tablet 9111A	2050.00	1678.00
HP-11C Slim-Line Advanced	135.00	119.00
HP-12C Slim-Line Financial	150.00	129.00
HP-41 CV New 2.2 Bytes Mem	325.00	250.00
HP-41-C Calculator	250.00	185.00
Card Reader For 41CV/C	215.00	162.00
Printer For 41CV/C	385.00	289.00
Optical Wand For 41 CV/C	125.00	97.00
Quad Ram Equals 4 Mem. Mods	95.00	81.00
Memory Modules For 41C		25.00
HP-97 Programble Printer	750.00	595.00
HP-67 Programble Calculator	375.00	295.00
HP-34C Programble Scientific	150.00	117.00
HP-38C Programble Bus. R/E	150.00	117.00
HP-32E Adv. Scientific	55.00	48.00
HP-37E Business Mgmt.	75.00	57.00

commodore

	List	Our Price
8032-32K 80 Col Crt	1495.00	1097.00
4032-32K 40 Col Crt	1295.00	989.00
4016-16K 40 Col Crt	995.00	788.00
8050-Dual Disk 950K	1795.00	1344.00
4040-Dual Disk 343K	1295.00	989.00
C2N-Cassette Drive	95.00	77.00
4022-80 Col Printer	795.00	639.00
8024-Mannsman Tafley	1995.00	1545.00
25CPS-Starwriter	1895.00	1399.00
CBM-IEEE Modem	279.00	225.00
Voice Synthesizer	395.00	325.00
VIC 20	299.00	255.00
PET to IEEE Cable	39.95	34.00
IEEE to IEEE Cable	49.95	39.00

XEROX

	List	Our Pr
Xerox 820-1 5-1/4″ Disk	2995.00	2399.00
Xerox 820-2 8″ Disk	3795.00	2995.00
CP/M Operating System	200.00	169.00
Word Processing Software	500.00	429.00
Super Calc	295.00	249.00

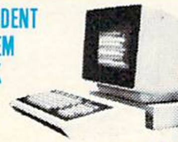
ATARI PERSONAL COMPUTERS

	List	Our Price
400 16K Bytes of Ram	595.00	337.45
800 16K Bytes of Ram	995.00	699.00
410 Program Recorder	90.00	77.00
810 Disk Drive	600.00	457.00
825 Printer (80 c.l Centronic 737)	999.95	769.00
820 Printer (40 col impact)	450.00	353.00
830 Acoustic Modem	200.00	155.00
850 Interface Module	220.00	192.00
Atari Visicalc	200.00	164.00

TEXAS INSTRUMENTS

	List	Our Price
TI 99/4A Console New Improved	950.00	369.00
10″ Color Monitor High Resolution	399.95	339.95
32K Memory Module	399.95	312.95
Speech Synthesizer	149.95	127.95
Disk Memory Drive	499.95	390.95
RF Modulator	49.95	42.50
Telephone Coupler (Modem)	224.95	185.95
Printer (Solid State)	399.95	315.95

NEC STUDENT SYSTEM 64K

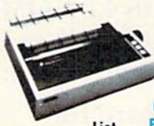


- NEC PC 8001 A Computer
- NEC PC 8012 A Expansion
- NEC PC 8031 A Dual Drive

12″ Grn. Phs. Video Monitor

MSL 3565.00
Save $1170.00
Our Price $2395.00

PRINTERS



	List	Our Price
EPSON		
MX 80 FT	$ 745.00	$ 549.00
MX 80 IMPACT	645.00	447.00
MX 70 IMPACT	500.00	369.00
MX 100	995.00	725.00
ANADEX 9501	1650.00	1299.00
NEC		
5510	3195.00	2445.00
5515	3295.00	2545.00
3510	2495.00	1795.00
3515	2545.00	1849.00
OKIDATA		
MICROLINE 80	545.00	395.00
MICROLINE 82	649.00	549.00
MICROLINE 83	1050.00	769.00
PAPER TIGER		
445G with Graphics	795.00	695.00
460G with Graphics	1,394.00	899.00
560G New full size	1,695.00	1,139.00
DIABLO (LETTER QUALITY)		
630 R102 bi-directional tractors	2,965.00	2,450.00
1640K109 keyboard tractors	4,000.00	2,899.00
630 RO Receive Only	2,710.00	2,250.00
1650K 136 keyboard tractors	4,000.00	3,100.00

MONITORS

	List	Our Price
BMC		
12″ Grn. Phs KQ (15 Hz)	219.00	165.00
12″ Grn. Phs EO (18 Hz)	$249.00	185.00
12″ Grn Phs (20 Hz)	279.00	209.00
12″ Colour Composite Hi Res.	439.00	319.00
Grn. Phs 12″	275.00	165.00
NEC Grn. Phs. 12″	225.00	179.00
Zenith 12″	159.00	119.00
LE MONITOR (SANYO TYPE MONITOR)		
• 9″ B & W	179.00	159.00
• 9″ Green Phs	189.00	165.00
• 12″ B & W	209.00	185.00
• 12″ Green Phs.	229.00	199.00
•• 12″ Green Phs.	199.00	159.00
• Anti Reflective Screen		
•• T.V. Grade Screen		
SANYO MONITORS		
13″ Color (new) high quality	$550.00	399.00
12″ Green Phs.	360.00	269.00
12″ B & W	340.00	239.00
15″ B & W	370.00	255.00
9″ B & W The Best Seller!	235.00	159.00
9″ Green Phs.	275.00	199.00

1 Year Extended WARRANTY
$99.00
INQUIRE

$ AVINGS
ERVICE
ELECTION
ATISFACTION

XEROX 820 WORD PROCESSING SYSTEM (INCLUDES SOFTWARE)

MSL 3495.00
OUR PRICE $2649.00
Save 846.00

TERMINALS

TELEVIDEO	List	Our Price
910	$ 699.00	$ 599.00
912C	950.00	699.00
920C	995.00	795.00
950C	1,195.00	949.00

DISKETTES SOLD IN BOXES OF 10

(Min. Purchase) $100

PRICE PER DISKETTE	List	Our Price
DYSAN		
104/1 5″ SOFT SECTOR	6.00	3.99
104/1D =″ DBL. DEN.-SOFT SEC.	6.40	4.60
3740/1 8″ SOFT SECTOR	7.25	4.75
3740/1D 8″ DBL. DEN.-SOFT SECTOR	10.75	6.90
MAXELL		
MD-1 5″ SOFT SECTOR/DBL. DEN.	5.00	3.88
MD-2 ″ SOFT SECTOR/DBL. SIDE./DBL. DEN.	7.00	4.90
FD-1 8″ SOFT SEC./DBL. DEN.	6.50	4.50
FD-1 8″ SOFT SEC./DBL. SIDE/DBL. DEN.	8.50	5.95

DRIVES

CORVUS	List	Our Price
5 MBYTES	3,750.00	$3,050.00
10 MBYTES	5,350.00	4,449.00
20 MBYTES	6,450.00	5,325.00

EXPANSION BOARD

Q STAR
16K RAM BOARD
LIST 199.00 OUR PRICE 129.00

SOFTWARE FOR APPLE II

	List	Our Price
Language System with Apple Pascal	495.00	399.00
BPI General Ledger System	395.00	319.00
Visidex	200.00	159.00
Visicalc	200.00	159.00
Desktop Plan II	200.00	159.00
Microlab Database System	150.00	119.00
Stoneware DB Master	229.00	189.00
Muse Supertext II	150.00	119.00
Softape Magic Window	99.00	72.00

commodore COLOR COMPUTER VIC-20

Vic-TV Modual $19.00
Vic Cassette $69.00
Vic Disk Drive .... $ Call
Vic 6 Pack program $44.00

MSL 299.00
OUR PRICE $255.00
Save 44.00

SOFTWARE FOR COMMODORE

	List	Our Price
Ozz-The Information Wizard	395.00	$289.00
Wordcraft 80	395.00	289.00
Irma-Info Retrieval & Mgmt. Aid	495.00	389.00
Dow Jones Portfolio Mgmt.	149.00	119.00
Pascal Development Pkg.	295.00	219.00
Ebs-Receivables, Inventory	750.00	569.00
Bpi-General Ledger	395.00	389.00
Word Pro 3-40 Column	250.00	175.00
Word Pro 4-80 Column	375.00	259.00
Word Pro 4 Plus	450.00	319.00

MODEMS

	List	Our Price
NOVATION CAT MODEM	189.95	140.00
NOVATION D-CAT	199.00	150.00
NOVATION APPLE-CAT	389.00	329.00
HAYES MICROMODEM	379.00	295.00
HAYES SMARTMODEM	279.00	239.00

CALCULATORS

CASIO	List	Our Price
HR-10 Paper Feed	49.95	39.00
HR-12 Paper Feed	54.95	42.00
FR-100 Paper Feed	79.95	59.00
FR-1210 Paper Feed	129.95	99.00
PQ-20	29.95	23.00
LC-785	12.95	10.00
LC-3165	12.95	10.00
FX-68 Scientific	29.95	23.00
FX-81 Scientific	19.95	17.00
FX-3600P Scientific	39.95	29.95
FX-602P "Computer Talk" 88 Memories Programming Upper & Lower Case Dot Matrix 512 Step	129.95	99.00
FX-702P Solves Problems with Alpha-Numeric Clarity, uses Basic Language	199.95	159.00

TELE. ANSW. DEVICES

PHONE MATE	List	Our Pr
900 Economy	119.95	86.00
905 Remote Dual Tape Sys.	199.95	144.00
910 60 Calls Per Side	159.95	115.00
920-C Vox	199.95	144.00
925 Remote Call Monitor	239.95	173.00
930 Remote Twin Cass.	299.95	216.00
950 Remote	339.95	245.00
960 Remote Built In Tele Microcomputer Controlled	399.95	288.00

MICRO MAIL ORDER BUSINESS WORLD WAREHOUSE
18720 Oxnard, #108
Tarzana, CA 91356

OUTSIDE CA CALL TOLL FREE 1 (800) 423-5886 IN CA (213) 996-2252

Telex: 182852
Answer: MICKO TZNA

*California residents add 6% sales tax.
**Add 3% Shipping & Handling — Add 3% surcharge for credit cards. Orders cannot be shipped unless accompanied by payment, including shipping, handling and tax where applicable.

Name (Please print) ____________
Address ____________
City ________ State ________ Zip ________

Qty	Make	Model	Description	Price	Total

TOTAL ORDER $ ________
TAX IF APPLICABLE* ________
SHIPPING & HANDLING** ________
TOTAL ENCLOSED $ ________

☐ Certified Check or M.O. ☐ Allow 2 weeks clearance for personal check
☐ Bank Wire Transfer
☐ VISA ☐ MasterCard

CREDIT CARD # ____________
Exp. Date ________ Signature ____________

HP • D.C. HAYS MICROCOMPUTERS • PAPER TIGER • GTE • DIABLO • BMC • CENTRONIC • NEC • HAZELTINE • MEMOREX • AMPEX • PERSONAL SOFTWARE • ADDS • XEROX • DIABLO

GTE • PERSONAL SOFTWARE • PAPER TIGER • D.C. HAYS MICROCOMPUTERS • NOVATION • CANON • HAYDEN • HITACHI • VERBATIM • AMPEX • HAZELTINE • NEC • CENTRONIC • BMC

MAXELL • DYSAN • EPSON • CCS • SHARP • CASIO • HP • VERBATIM • MEMOREX • SOROC • CORVUS • PERSONAL SOFTWARE • CCS name, in seconds you can find on what diskette a certain program is on. For example, "SORT" would match filenames "SORT" and "QUICK-SORTER"; "T?N" would match "TOP TEN", "TIN" and "TNT". When filenames are found, the display can be paused and resumed at your convenience. Reports available include an alphabetized list of all your filenames, a summary listing of each diskette, or a list of all filenames, grouped by the disk they are on. For each, you can select the entire report or choose a range. All sorting is done in machine language. HESCAT is menu-driven with excellent human engineering. Price is $39.95 on disk plus $2 postage.

Human Engineered Software
3748 Inglewood Blvd. Rm. 11
Los Angeles, CA 90066

Reference Guide For Computer Publications

Micro... Publications in Review is a monthly publication, a quick reference, of titles of articles in the 70+ Micro-Mini Computer and Technology Publications. It has a magazine format and is intended to keep the reader abreast of this explosive industry through (1) reprints of the Table of Contents and (2) a subject index consisting of 26 major disciplines with each having from 6 to 40 classifications. Approximately 70 publications are covered with over 800 articles per issue.

Where possible, a fourth breakdown by computer and a fifth by language and/or operating system is classified. These later breakdowns will only appear in special issues.

Any title can be classified in one or two classifications and the breakdown is manually coded for computer indexing, since in most cases the title does not offer the necessary information for any keyword search.

Vogeler Publishing Inc.,
455 Crossen Ave.
Elk Grove Village, IL 60007
312-228-0951

PET Terminal Emulator

The PET Terminal Emulator is a combination of hardware and software which converts a Commodore computer with 4.0 ROMs into a sophisticated terminal emulator. This system gives you:

– A serial interface board for PET to serial conversion

– A machine language program on a PROM to handle communication

– Communication speeds between 150 and 4800 baud

– Choice of three different major manufacturer terminal configurations

– Uses almost none of the PET's RAM (only 512 bytes)

– Permits PET BASIC program residing in RAM during terminal operation

– Automatically relocates itself to operate in conjunction with other machine language programs

– Adds terminal function keys and special characters to character set

– Ability to redefine keys on the keyboard for greater flexibility

This system is optimized to give the highest possible speed when the PET is operating as a terminal and to permit taking advantage of the many programs which are designed to use the screen characteristics of terminals in their operation.

The board, called the Serial Connection, takes the signal from the Commodore computer and translates it to a serial signal that can be used with any standard RS 232C modem or acoustical coupler. It has the capacity to send and receive at speeds between 150 and 4800 baud.

The machine language program which is stored on a programmable read only memory (PROM) handles the communication between the PET and another computer. Since there are an increasing number of programs that take advantage of the screen handling features of terminals for data entry and printing on the screen this program permits the user to choose to emulate either the ADDS Regent 100, the Lear Sigler ADM 31, or the Televideo 950. These are three very popular terminals, and users should find it advantageous to use one of the three in almost all situations.

Since the machine language program is stored on PROM it uses very little of the Commodore computer's RAM. In addition, once the machine language program is initialized it is protected from PET BASIC programs that may be loaded into the PET. PET BASIC will not write over it, and it does not interfere with the operation of PET BASIC programs if it is initialized before the PET BASIC programs are loaded into the computer.

The documentation for the PET Terminal Emulator is very thorough; more than thirty pages. And there is a table of contents and an index. The manual is written for ordinary users. Each step in installation and use is carefully and explicitly described.

The PET Terminal Emulator sells for $175. It can be ordered from:

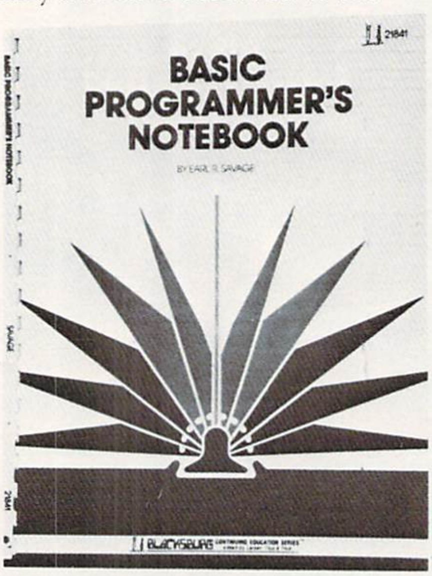
Amplify, Inc.
2325 McBride
Iowa City, IO 52240

BASIC Programmer's Notebook

BASIC Programmer's Notebook, a


commodore
Check Our New Lowest Prices!
EQUIPMENT:
VIC-20 (Includes RF Modulator) 265
CBM 4016 CPU (40 Col. Screen, 16K RAM) 790
CBM 4032 CPU (40 Col. Screen, 32K RAM) 990
CBM 8032 CPU (80 Col. Screen, 32K RAM) 1090
CBM 8096 CPU (80 Col. Screen, 96K RAM) 1590
CBM Micro Mainframe (Super PET) 1690
NEW VIC Single Disk Drive (170K per 5¼ Diskette) 470
CBM 2031 Single Disk Drive (170K per 5¼ Diskette) 570
CBM 4040 Dual Disk Drive (170K per 5¼ Diskette) 990
CBM 8050 Dual Disk Drive (½ Meg per 5¼ Diskette) 1340
VIC Tractor Feed Printer (80 Col. 30 CPS) 325
CBM 4022 Tractor Feed Printer 625
8023P Dot Matrix Printer (136 Col. 150 CPS) 790
NEW Tally 8024-7 (7x7 Matrix) Printer 1275
Tally 8024-9 (7x9 Matrix) Printer 1425
8300P Letter Quality Printer (40 CPS) 1790
CBM 8010 Phone Modem 225
NEW CBM C2N Cassette Deck (New Style) 65
CBM CPU/IEEE Cable 35
CBM IEEE/IEEE Cable 40
SOFTWARE:
Wordcraft 80 Wordprocessor 295
Wordpro 4+ Wordprocessor 325
OZZ Data Base System 295
Visicalc 175
Tax Preparation System 395
IRMA Information Retrieval 395
Dow Jones Portfolio 115
PUBLICATIONS:
CBM User Guide 7.95
CBM Basic 4.0 Reference Manual 9.95
CBM Disk Manual 7.95
CBM Printer Manual 7.95
MOS Hardware Manual 6.95
MOS Programming Manual 6.95
The PET Revealed 19.95
Library of PET Subroutines 19.95
Commodore Software Encyclopedia 4.95
VIC SOFTWARE:
VT 106A Recreation Six Pack 43.95
Includes Car Chase, Blue Meanies, Space Math
Slither/Super Slither, Biorhythm Capability
VT 107A Home Utility Six Pack 43.95
Includes Personal Finance I, Personal Finance II,
VIC Typewriter, Expense Calendar, Loan & Mortgage
Calculator, Home Inventory
VIC 1901 VIC Super Alien 29.95
VIC 1904 Super Slot 29.95
VIC 1907 Super Lander 29.95
VIC 1908 Draw Poker 29.95
"CALL ABOUT SUPPLIES"
COMING SOON:
More VIC Peripherals and Software
CBM 8250 Dual Disk Drive
Data Acquisition and Control Devices
All items Insured
COD – UPS
Prepaid Orders Shipped Free
In stock items shipped within 48 hours
MASTERCHARGE OR VISA ADD 3%
GA. RESIDENTS ADD 4% SALES TAX
MART
3300 Buckeye Rd., N.E.
Suite 641
Atlanta, Ga. 30341
404-458-0729
Call 9 AM-5 PM EST M-F
WORD PROCESSING SPECIAL
CBM 8032
CBM 4040
CBM 8300 (Letter Quality, DIABLO 630, 40 CPS, Bi-directional)
WORDCRAFT 80
• Includes all cables
$4200
CALL ABOUT EQUIPMENT AND/OR SOFTWARE SUBSTITUTIONS

110-page textbook by Earl R. Savage, presents an array of techniques and subroutines by which both novice and experienced programmers can write programs easier and better. Rewritten from notes collected over the years, the Notebook offers shortcuts to replace lengthy techniques, statements sequences to make programs more professional in appearance, and techniques for increasing their effectiveness and efficiency. Instructions and explanations given make it possible to do many things with the computer which may not have occurred to the



user. No complete programs are given, only many subroutines and program fragments that allow the user's imagination and sense of purpose to develop programs tailored to specific needs. The advanced programmer will find an array of ideas that have been worked out to save time.

Although the program statements are written in the Radio Shack Level II BASIC, they can be used in various types of computers either exactly as written or with only minimal changes to adapt them to most other BASIC dialects.

Each topic and each subroutine is stated clearly and explained in detail; how and why it works, when and how to use it. Flowcharts clarify the logic further when needed.

The book is divided into four major parts. Part 1 contains a variety of suggestions for improving the quality, size, and speed of one's programs. Part 2, the largest section, contains 50 Notes and many variations of them. Each Note is comprised of a listing of the statements as they would appear in a program, an analysis of the statements, suggested uses of the subroutine or technique, and variations of the listing with a flowchart if appropriate. Part 3 gives detailed information in utility programs which can be a real asset in program writing and guides the user in the wise selection of those aids. Part 4 presents hardware aids to increase the efficiency of the entire computer operation including, for example, various ways to shorten saving and loading time, memory expansion, and others.

BASIC Programmer's Notebook, Earl R. Savage, 110p., 8½" x 11", plastic bound, soft cover, Book No. 21841, $14.95 plus $1.00 shipping and handling.

Virginia residents add 4% sales tax. VISA and Master Cards accepted. For purchase or further information, contact:

Group Technology, Ltd.
P.O. Box 87
Check, VA 24072
703-651-3153

Daisy Wheel Printer From Leading Edge

Starwriter F-10, a daisy wheel printer, includes such features as: low profile design (6" high) to fit easily into your system, industry-standard ribbon cartridges, 40 or 55 cps models, standard parallel or RS232-C interfaces (including ETX/ACK, X-ON/S-OFF) protocols, extensive built-in word processing functions that allow easy adaptability, uses plastic or metal wheels,

low noise operation. A choice of friction feed or optional bidirectional tractor feed is available.

Leading Edge Products
225 Turnpike Street
Canton, MA 02021
(800)343-6833

More Atari Memory

Mosaic Electronics has announced the 16K/32K RAM board. Designated Part #H216, the Mosaic 16/32K RAM, adds 16K to an Atari computer system. After the Atari user has exhausted the potential of 16K, upgrade to 32K is very easy using the $60 upgrade kit #H212. Atari 400 owners can use their existing 16K RAM to upgrade to 32K for $120 total. The Mosaic 16/32K RAM is of particular interest to owners of the Atari 400 with 16K, the Atari 800 with 16K, and the Atari 800 with 32K.

For more information write to:

Tanya Hickman, Customer Service
Mosaic Electronics
P.O. Box 748
Oregon City, Oregon 97045

Software Packages For The Atari From Roklan

Twelve new software packages have been developed by Roklan Corporation of Rosemont, Illinois. Three categories of programs are available: pro-

gramming utilities, business applications and games.

Among the utilities designed for programmers are a Copy/ Verify – Format/Certify program and an Absolute Disk Editor. A Telecommunications package allows computers to "talk" to each other and transfer data efficiently. In addition, a 6502 simulator program has been developed for Apple computer systems.

Among the business applications packages are FinPac, a financial calculations package containing programs for all financial calculations, and Real Estate, a real estate investors aide which will also be useful to brokers and appriasers.

Game programs to be introduced include several microcomputer versions of popular arcade games. Among them are Deluxe Invaders, Midway's Gorf, and Wizard of Wor.

Roklan Corporation
10600 Higgins Road
Rosemont, IL 60018

Symtec Light Pen For The Atari And VIC Home Computer

Symtec announces a light pen for the Atari 400/800 and the VIC-20 home computers. This model is identical to the Symtec pens used by the American Heart Assoc., the Army, Bell and display systems. The pen is easily programmable and can activate any software command the Atari or VIC can handle. Programming instructions are included for the technical owners and user programs will be announced. The barrel is stainless steel and has a touch-ring that sends a signal to the computer with the touch of a user's finger or by tapping a metal clip. The pen sees where it is being pointed at with an accuracy of + or − 1 pixel. The pen sells for $150.00. For more information contact:

Software Etc.
20828 Vermander
Mt. Clemens, MI 48043
(313)792-3391
or on the Source at TCX060

Software For Elementary Students

Orange Cherry Media, a division of Multi Dimensional Communications, Inc., announces its line of microcomputer software for the educational and consumer marketplace.

Fifty-nine programs have

been marketed which will include software in the areas of language arts, reading, communication skills, science, and mathematics. The programs have been specifically programmed for all popular models of Commodore PET, Radio Shack (TRS-80), and Apple microcomputers.

Similar to Orange Cherry's audio visual lines, the microcomputer software has been designed for the elementary market, grades K-8.

According to Carol Vazzana, Vice President of Marketing at Orange Cherry Media, "Our software relates directly to curriculum objectives and takes youngsters on adventures that enliven lessons in subject areas."

Programs are available on separate cassette tapes and disks for each of the various models of PET, TRS-80 and Apple computers. Many of these programs provide students and instructors with a choice of subject matter. The learner's responses are recorded to determine the reading and comprehension levels of the subject matter.

Orange Cherry Media
7 Delano Drive
Bedford Hills, NY 10507
(914)666-8434

Programs For The Classroom

Two microcomputer programs to aid classroom instruction were released in the second 1981-82 issue of CourseWare Magazine. The instructional package includes program documentation, teacher and student guides.

Decimal Estimation presents a student with a multiplication problem such as:

42.31 x .1602 = .677780620

The user moves the decimal by pressing the R key to move right or the L key to move left. The student places the decimal point to his/her satisfaction. RETURN is pressed to register the response and the computer accepts or rejects the response. A running timer is displayed and, after each success, average time per problem.

Alphabetize gives the student experience in ranking lists of between three and eight words. The teacher supplies a vocabulary. The computer selects at random the chosen number of words which appear on the screen. The student interchanges pairs of words until the list is correctly alphabetized. A score is kept which represents elapsed time to complete the task.

Versions of the instructional package are published for Apple, PET and TRS-80 microcomputers. They were mailed to subscribers in the US, Canada, Europe, Taiwan and Japan. CourseWare Magazine is located at:

4919 North Millbrook #222
Fresno, CA 93726

Disks And New Educational Programs From Teacher's Pet Software

All Teacher's Pet Software programs are now available on two disks with menus and automatic program loading from the menus. One disk contains all math and logic programs, the other all language and management programs. When a student finishes the programs, a single keypress will either restart the program or reload the menu. These programs will run on any 40 column BASIC 3.0 or 4.0 PET using a 2040, 4040, or 2031 disk drive.

Teacher's Pet Software has just released 4 new decimal and fraction math programs designed for students in grades 4 through 8. In all programs, up to 4 students can take turns working individual problems, and be scored separately.

Decimal Multiplication provides 5 levels of multiplication problems from single digit to 5 digit multiplicands with up to 4 digits after the decimal place. The program follows the paper and pencil procedure, making it a very effective tutorial. This is the only program a teacher needs to reinforce the common multiplication algorithm, and to teach the multiplication of decimal and whole numbers.

Comparing Fractions provides practice in comparing fractions, providing pictures of the fractions and a thorough, interactive tutorial teaching students a simple and fool-proof method for comparing two fractions.

Comparing Fractions II extends the practice of Comparing Fractions to more difficult fractions to let students test their mastery, and provides the same tutorial for review. Both programs have "help" and the directions available at any time in the program.

Recognizing Fractions provides the opportunity for practice in naming fractions given a picture, in drawing the picture (on the screen) given the fraction, or in a random mix of the two. The teacher can select the mode, or let the student choose.

Teacher's Pet Software has available over two dozen other programs in math, language, logic, and management suitable for elementary and junior high school students. Many of the programs are being used for remediation in high schools. These programs were written by Glenn Fisher, an elementary school teacher, computer consultant, and author of articles on computer use and programming techniques. A free brochure describing all programs is avail-

able from:

Teacher's Pet Software
Glenn Fisher
1517 Holly St.
Berkeley, CA 94703

Edu-Ware Announces Rendezvous Release

Edu-Ware has announced the April release of Rendezvous™, a space shuttle flight simulation by a NASA senior scientist, Wesley Huntress. In four phases Rendezvous simulates an actual space shuttle flight from Earth Liftoff, through Orbital Rendezvous and Approach, to Alignment and Docking with a space station. The system is available in Applesoft and Atari BASIC, and will retail for $39.95.

Rendezvous' high-resolution graphics and animation capture the drama and difficulty of space navigation. Shuttle liftoff and flight conditions are governed by gravitational forces, thrust, and the player's own piloting skills. Distance and engine burn time (measured in kilometers and seconds) complicate orbital calculations.

Throughout the final alignment and docking sequence the player views the rotating, 3-dimensional space station as it would appear through a space craft's anterior window. 3-D Solid-Structure Graphics™ bring to life the massive bulk and solidity of the station and docking bay. In a precisely timed operation, the player must maneuver his craft around the station and into the narrow, tunnel-like bay.

Once docking is achieved, a sophisticated algorithm scores performance.

Maneuvers are implemented with either keyboard or joystick controls.

Rendezvous is recommended for individual players, 13 and over. Suitable for classroom application in General Science, Earth Science, and Physics.

Edu-Ware Services, Inc.
P.O. Box 22222
Agoura, CA 91301

"My First Alphabet" Claims Atari $25,000 Award

Fernando Herrera, winner of the $25,000 Atari Star Award for the creation of a computer program, was inspired to write My First Alphabet to assist his visually-impaired son, Steve, in learning to read. The colorful, animated alphabet is a beneficial learning aid to children between two and eight years of age. More than $100,000 worth of prizes are awarded annually to creators of computer software in the contest sponsored by Atari, Inc. Herrera, an architect by training, recently switched careers as the result of his interest in home computers. He now manages a computer store on Long Island, New York.

ALL ATARI® HARDWARE 15%-25% OFF LIST PRICE

Atari 800 16K	710.00
Atari 400 16K	359.00
Atari 410 Cassette	80.00
Atari 810 Disk	480.00

ATARI® ACCESSORIES 10%-20% OFF LIST PRICE

8K Memory Board	40.00
16K Memory Board	80.00
Joysticks (pair)	19.00
Paddles (pair)	19.00

To order: Call 617 964-3080
Ask for mail order, or write:

BBI Mail Order

P.O. Box 365
Newton Highlands, MA 02161
(617) 964-3080

PLUS 10%-20% OFF ALL ATARI® SOFTWARE ALSO 3RD. PARTY HARDWARE AND SOFTWARE AT COMPARABLE SAVINGS

Use the card in the back of this magazine to order your **COMPUTE! Books**.

ATARI

ATARI 800 (16K)	**669**
ATARI 400 (16K)	**319**
ATARI 410 Recorder	**74**
ATARI 810 Disk Drive	**439**
ATARI 820 Printer	**249**
ATARI 822 Printer	**269**
ATARI 825 Printer	**589**
ATARI 830 Modem	**149**
ATARI 850 Interface	**164**
ATARI cx853 16K Ram	**79**
ATARI Software	**Call**

COSMIC COMPUTERS UNLIMITED

228 N. PROSPECTORS
DIAMOND BAR, CA.
ZIP 91765

(714) 861-1265

PRICES SUBJECT TO CHANGE. SHIPPING EXTRA. CALIFORNIA RESIDENTS ADD SALES TAX. ATARI IS A REGISTERED TRADEMARK

Kelly's Computing SPECIALS

ATARI — A Warner Communications Company

ADVENTURE INTERNATIONAL			
☐ Adventure #1, 2 & 3 (Disk)	$39.95	*NOW*	**$33.50**
☐ Adventure #4, 5 & 6 (Disk)	$39.95	*NOW*	**$33.50**
☐ Adventure #7, 8 & 9 (Disk)	$39.95	*NOW*	**$33.50**
☐ Adventure #10, 11 & 12 (Disk)	$39.95	*NOW*	**$33.50**
☐ Adventure's 1 - 12 (Disk)	$100.00	*NOW*	**$87.50**
☐ Rear Guard (Disk)	$24.95	*NOW*	**$19.50**
☐ Rear Guard (Cass)	$19.95	*NOW*	**$16.50**
☐ Galactic Empire (Cass)	$19.95	*NOW*	**$16.50**
☐ Star Treck 3.5 (Cass)	$19.95	*NOW*	**$16.50**
AUTOMATED SIMULATIONS			
☐ Temple of Apshai (Disk)	$39.95	*NOW*	**$33.50**
☐ Temple of Apshai (Cass)	$39.95	*NOW*	**$33.50**
☐ Star Warrior (Disk)	$39.95	*NOW*	**$33.50**
☐ Star Warrior (Cass)	$39.95	*NOW*	**$33.50**
☐ Date Stones of Ryn (Disk)	$19.95	*NOW*	**$16.50**
☐ Date Stones of Ryn (Cass)	$19.95	*NOW*	**$16.50**
☐ Crush, Crumble & Chomp (Disk)	$29.95	*NOW*	**$24.50**
☐ Crush, Crumble & Chomp (Cass)	$29.95	*NOW*	**$24.50**
☐ Invasion Orion (Disk)	$24.95	*NOW*	**$19.50**
☐ Invasion Orion (Cass)	$24.95	*NOW*	**$19.50**
☐ Ricochet (Disk)	$19.95	*NOW*	**$16.50**
☐ Ricochet (Cass)	$19.95	*NOW*	**$16.50**
☐ Mouse Attack (Disk)	$34.95	*NOW*	**$28.50**
☐ Mouse Attack (Cass)	$34.95	*NOW*	**$28.50**
AVALON HILL GAME CO.			
☐ Empire of the Overmind (Disk)	$35.00	*NOW*	**$29.50**
☐ Empire of the Overmind (Cass)	$30.00	*NOW*	**$24.50**
☐ Tankticks (Disk)	$29.00	*NOW*	**$23.50**
☐ Tankticks (Cass)	$24.00	*NOW*	**$19.50**
EDU-WARE			
☐ Compu-Read (Disk)	$29.95	*NOW*	**$24.50**
☐ Compu-Math-Fractions (Disk)	$39.95	*NOW*	**$33.50**
☐ Compu-Math-Fractions (Cass)	$29.95	*NOW*	**$24.50**
☐ Compu-Math-Decimals (Disk)	$39.95	*NOW*	**$33.50**
☐ Compu-Math Decimals (Cass)	$29.95	*NOW*	**$24.50**
GEBELLI			
☐ Match Racers (Disk)	$29.95	*NOW*	**$24.50**
K-BYTE			
☐ K-Razy Shootout (Cart)	$49.95	*NOW*	**$41.50**
ON-LINE SYSTEMS			
☐ Wizzard Princess (Disk)	$32.95	*NOW*	**$27.50**
☐ Jaw Breaker (Disk)	$29.95	*NOW*	**$24.50**
QUALITY SOFTWARE			
☐ Star Base Hyperion (Disk)	$22.95	*NOW*	**$18.50**
☐ Star Base Hyperion (Cass)	$19.95	*NOW*	**$16.50**
☐ Ali Baba and the Forty Thieves (Disk)	$32.95	*NOW*	**$27.50**
STRATEGIC SIMULATIONS			
☐ The Shattered Alliance (Disk)	$39.95	*NOW*	**$33.50**
SYNERGISTIC			
☐ Warlock's Revenge (Disk)	$35.00	*NOW*	**$28.50**
ATARI			
☐ Asteroids (Needs Joy Sticks) (Cart)	$44.95	*NOW*	**$38.50**
☐ Star Raiders (Needs Joy Sticks) (Cart)	$49.95	*NOW*	**$42.50**
☐ Space Invaders (Needs Joy Sticks) (Cass)	$44.95	*NOW*	**$38.50**
☐ Super Breakout (Needs Paddles) (Cart)	$44.95	*NOW*	**$38.50**

SEND FOR FREE PRICE LIST

INCLUDE $2.00 FOR SHIPPING AND HANDLING. CALIFORNIA RESIDENTS ADD 6% SALES TAX. ALL ORDERS SHIPPED FROM STOCK WITHIN 48 HOURS. WE ACCEPT **MASTERCARD AND VISA.** C.O.D.'S ADD $5.00. PRICES SUBJECT TO CHANGE WITHOUT NOTICE.

NAME ______________________

STREET ______________________

CITY ______________ STATE ______ ZIP ______

CARD # ______________ EXP. DATE ______

SIGNATURE ______________________

3515 BRYCE WAY, RIVERSIDE, CA 92506

ORDERS (714) 787-7002

Home Budget Programs For The Atari

Personal Finance
From: COMPUTARI
9607 Athlone
Dallas, TX 75218

Personal Finance offers several different programs, such as check entry, budget entry/review, check search, tabulations, expense/budget charts, check reconciliation and a secondary menu which offers eight utility programs. A four-color bar graph program provides the same type of information as does the Tabulations program – expenses versus budget by month, year-to-date, or by expense category over twelve months. In Check Search, you can search through your checks and records by payee, expense category, tax deductible status, or check number.

This program retails for $49.95, operates with 24K and an 810 disk drive. It is an autoboot, menu driven package.

****

Budgetmaster
From: Sunrise Software
P.O. Box 25621
Cleveland, OH 44125
(216)581-9359

Budgetmaster will:

1) Teach you how to set up a personal budget.

2) Allow you to set up your categories, budget amounts, and input your expenses, without additional equipment or materials.

3) Make changes in category names, budget amounts, expense inputs, and otherwise manipulate data for 12 months at a time to tape or disk.

4) Analyze your spending patterns and your budgeting skills to assist you in making future budget decisions.

5) Interface with most printers, giving you a hard copy of your displays whenever you wish.

6) Save your data and category names on either cassette or diskette.

Budgetmaster is currently available on disk for $34.95. Check Balancer may also be purchased with Budgetmaster for a total sum of $39.95, and will interface with Budgetmaster. Another version of Check Balancer will operate independently and will retail in the $14-$17 range if bought separately. Cassette versions are available for both for slightly less.

These programs require an Atari 400 or 800 with 32K RAM.

* * * * * * *

Personal Finance System
From: Dynacomp, Inc.
1427 Monroe Ave.
Rochester, NY 14618

This is a single diskette, menu-oriented system – a collection of ten interdependent programs that are used to create, correct, maintain, sort and display your personal finance records. All information is stored in the form of concise records in a master data file. 600 records per disk.

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. It will produce monthly bar graphs of your expenses by category.

The program requires 24K of memory. It retails for $34.95.

Reston Publishing Introduces New Software/Book Products

Reston Publishing Company, a subsidiary of Prentice-Hall Inc., unveiled three new entries in its personal computer book line.

The three new computer book/software packages are: Paint, package available for use on the Atari 400 or 800 personal computer; Multiploy, an educational game for the Apple II Plus computer; and Space Knights, a game of warring space pirates.

Paint is an educational software program written by the Capital Children's Museum of Washington, DC. Utilizing the color capabilities of the Atari, the students are able to create "paintings" of their own on the computer using hundreds of color textures. The paint brush is controlled by the joy stick of the computer. The Paint package consists of an educational book with the complete software – both work side-by-side to guide the students through every step.

Multiploy, by Paul Coletta, is an educational game developed for the Apple home computer. The combination of arcade-type action and mathematical operations (addition, subtraction, multiplication, and division) provide a fun environment for learning. There are three different skill levels for each operation, making this a game for any age group. The computer automatically ranks the players at every skill level so they can follow their own progress as they climb to the top level.

Multiploy will be available in May. Cost is $19.95.

Space Knights, especially designed for sci-fi computer gamers, is written by David L. Heller and Robert Kurcina. Using the Atari 400 personal computer, this new concept in software links adventure, science fiction and the computer's color, sound, and graphic capabilities. The result is a series of role-playing games for all ages. Players follow Jake and Lisa through a series of action-packed adventures in an attempt to do better than they do on their adventure. The encounters are successively harder to provide the challenge.

Reston Publishing Co.
1148 Sunset Hills Rd.
Reston, VA 22090
(703)437-8900

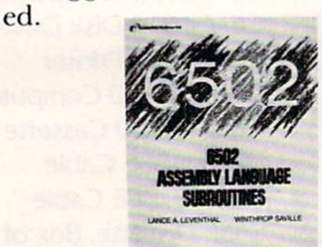
6502 Assembly Language Subroutines

Osborne/McGraw-Hill has just released the first title in a new series of Assembly Language Subroutines.

6502 Assembly Language Subroutines by Lance A. Leventhal and Winthrop Saville presents an overview of assembly language programming for the 6502 microprocessor and a collection of more than forty useful subroutines. These routines can be used as subroutines in actual applications and as guidelines for complex programs.

Leventhal and Saville provide code for common routines including: code conversion, array manipulation, arithmetic, bit manipulation, summation, sorting and searching. Also included are examples of I/O routines, interrupt service routines, and initialization routines for common family chips such as parallel and serial interfaces, and timers. These subroutines can run on microcomputers manufactured by Apple, Commodore, Ohio Scientific, Atari, and on other 6502-based micros.

This book, as well as forthcoming books in the series, is aimed at the user who wishes to utilize assembly language immediately. All routines have been debugged, tested and documented.



The book has 596 pages and sells for $12.99 paperback. For more information:

Kendal Anderson
Osborne/McGraw Hill
630 Bancroft Way
Berkeley, CA 94710
(415)548-2805

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. ©

Advertisers Index

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?

4 5 6 7 8 9 10 11 12

Place
Stamp
Here

COMPUTE! Magazine
Post Office Box 5406
Greensboro, NC 27403

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 236 BROOMALL, PA

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE! Magazine
515 Abbott Drive
Broomall, PA 19008

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 2312 GREENSBORO, NC

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE! Magazine
Post Office Box 5406
Greensboro, NC 27403

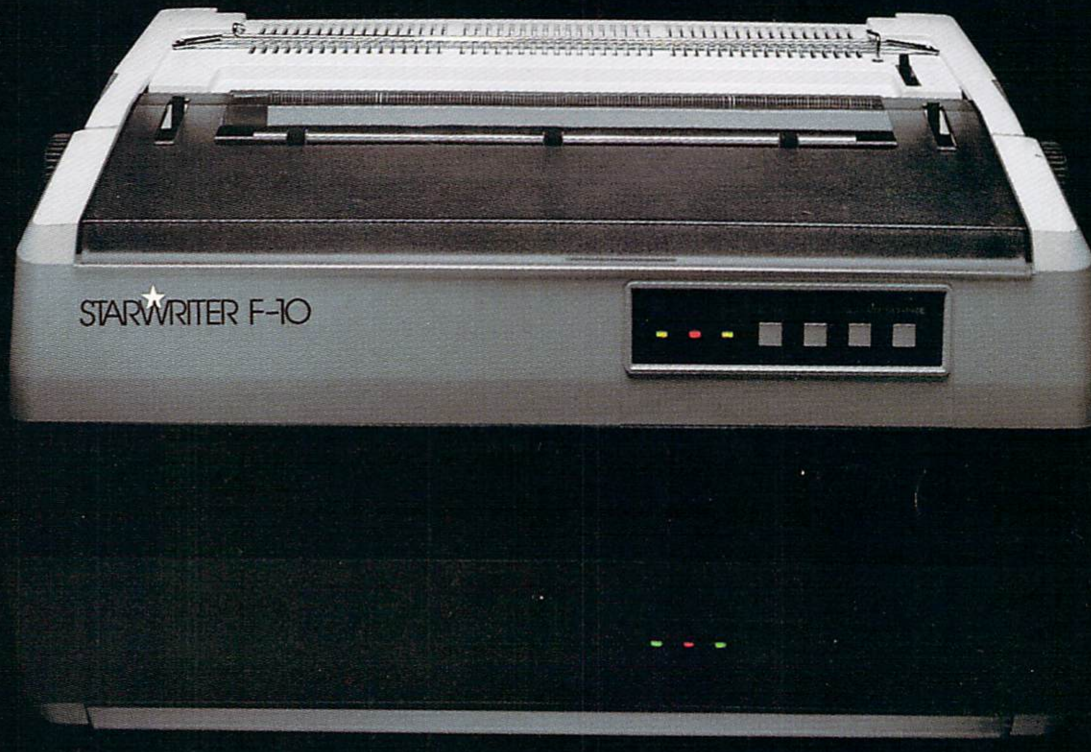
NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 236 BROOMALL, PA

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE! Magazine
515 Abbott Drive
Broomall, PA 19008

SAVE 6 INCHES, 10 POUNDS, AND $800.



On the new, slicked-up, trimmed-down Starwriter F-10.

It's C. Itoh's latest generation of letter-quality printers.

It cranks out flawless copy at 40 cps; and its full 15" carriage lets it double in brass for both letter processing and business applications. You can plug it into almost any micro on the market (serial or parallel) simply by plugging it in. And then make it keep on trucking with inexpensive, easily available Diablo compatible daisy wheels and ribbons.

In its serial mode, it can print just about anything (including boldface, underlines, subscripts and superscripts), and snap the carriage back to start the next line in less than a second. In its line mode, it prints in both directions, for even faster throughput.

(While making about as much noise as a cat walking on Kleenex.)

It's a nice, portable 30 pounds—about 10 pounds lighter than the Starwriters before it. And it stands exactly as tall (or precisely as small) as a dollar bill.

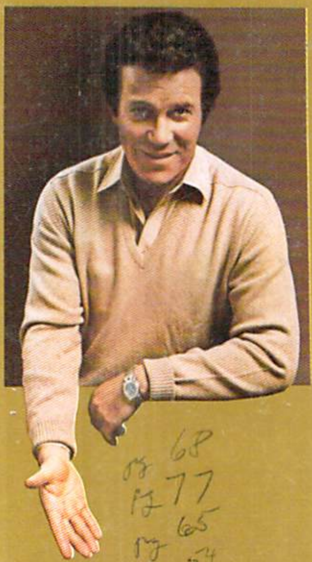
Speaking of which:

Incredibly, the Starwriter F-10 sells for about the same preposterously low price as its predecessors. Which is to say, about $800 less than a lot of other printers that don't even come close to measuring up. Or even better...

Measuring down.

Distributed Exclusively by Leading Edge Products, Inc., 225 Turnpike Street, Canton, Massachusetts 02021. Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.

LEADING EDGE.

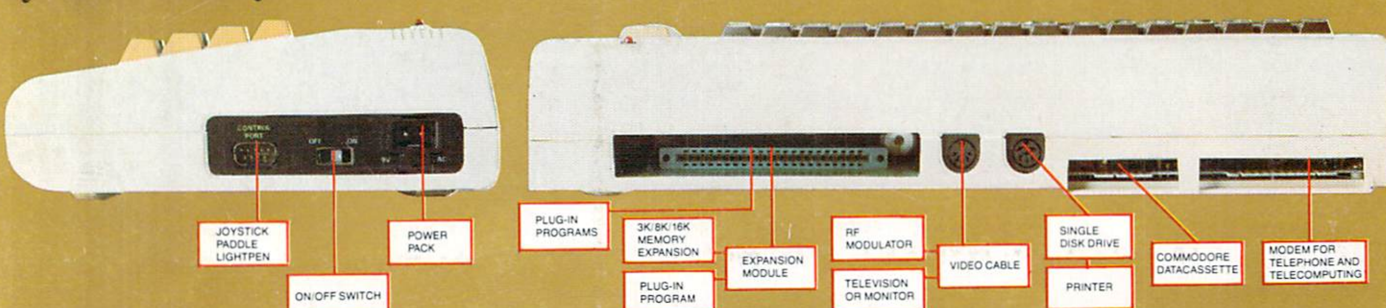


COMMODORE VIC-20®

"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	$458.95	$499.00	$399.00
Maximum RAM Memory	32K	16K	16K	32K
Keyboard Style	Full-Size Typewriter Style	Flat Plastic Membrane	Half-Size Typewriter Style	Calculator Style
Number of Keys	66	57	40	53
Programmable Function Keys	4	0	0	0
Graphic Symbols On Keyboard	62	0	0	0
Displayable Characters	512	256	64	256
Microprocessor	6502	6502	TI990	6809
Accessible Machine Language	YES	YES	NO	YES
Upper/Lower Case Characters	YES	YES	NO	NO
Operates with all Peripherals (Disk, Printer and Modem)	YES	NO	YES	YES
Full Screen Editor	YES	YES	NO	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 Jan. 1, 1982. Includes BASIC cartridge required for programming.

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 VIC-CO

Please send me more information on the VIC-20.

Name ____________________

Address ____________________

City ____________ State ________ Zip ________

Phone ____________________