The New Wave Of Personal Computers:
An In Depth Look At The New Commodore 64, Sinclair Color Spectrum, Epson Portable Computer, And More

COMPUTE!

$2.50
August
1982
Issue 27
Vol. 4, No. 8
63379 £1.85 in UK

The Journal For Progressive Computing™

Household Budget Manager
Control And Analyze Credit, Expenses, And Income

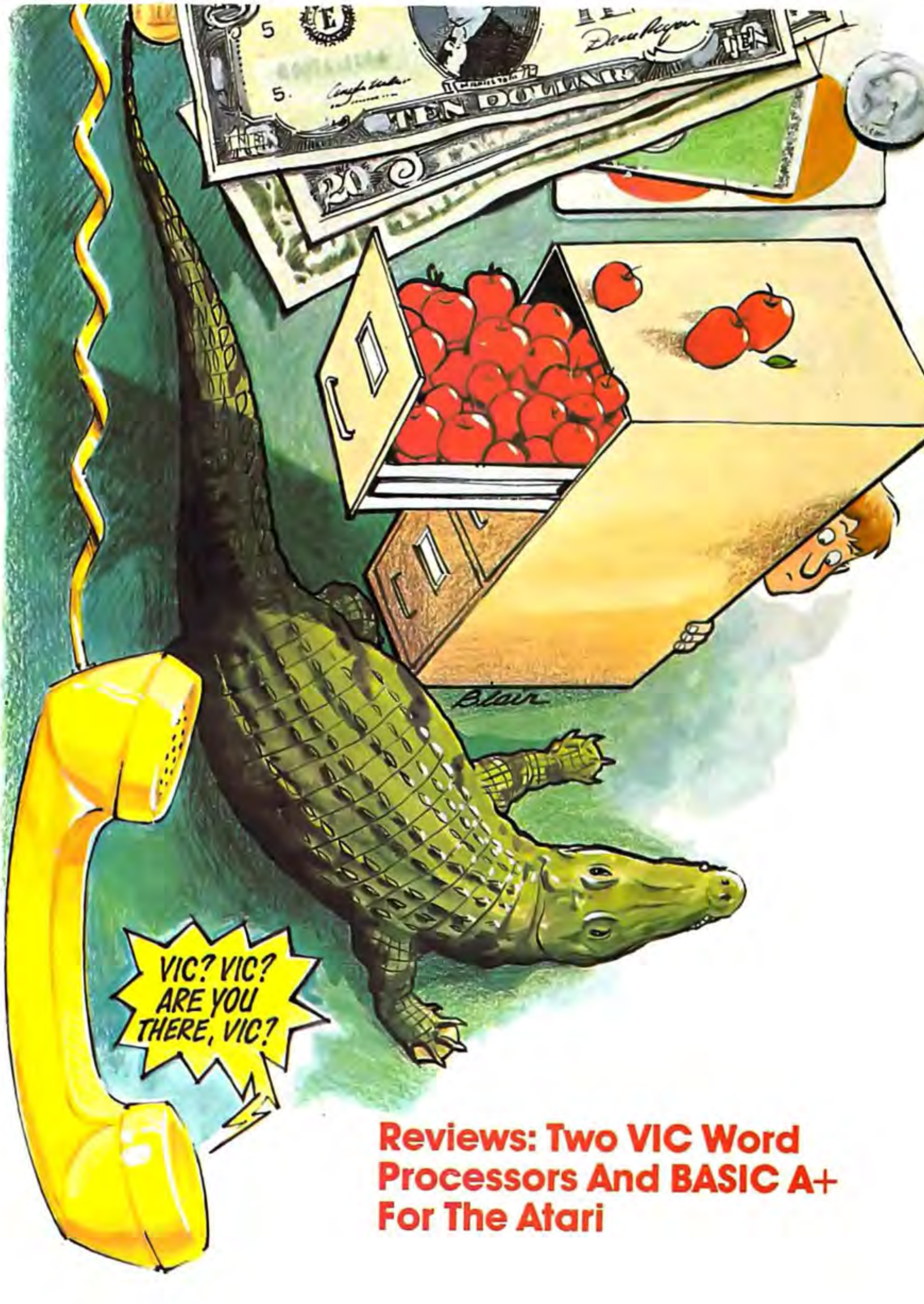
VIC-20 Communications: The RS-232 Interface

Apple Data Manager: An Intelligent Filing Cabinet

Home Energy Monitor For Radio Shack Color Computer

Guess That Animal
An Educational Commodore/Atari Game

Inner BASIC For PET/Commodore



Reviews: Two VIC Word Processors And BASIC A+ For The Atari

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

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

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

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

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

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

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

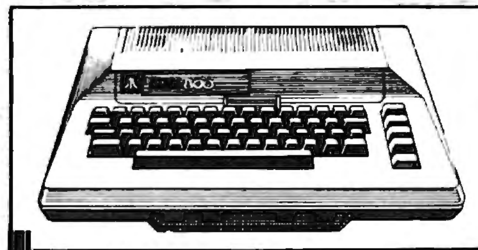
TM WordPro is a Registered Trademark of Professional Software, Inc. WordPro was written by Steve Punter.
All specifications subject to change without notice.

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

Tomorrow's Technology Today

Commodore Computer

VIC 20 Personal Computer	$ 299.95
VIC 1515 Graphic Printer	$ 395.00
CBM 4016	$ 995.00
CBM 4032	$1295.00
CBM 8032	$1495.00
CBM SuperPet 9000	$1995.00
CBM 8050 Dual Disk Drive	$1795.00
CBM 4040 Dual Disk Drive	$1295.00
CBM 2031 Single Disk Drive	$ 695.00
CBM 4022 Tractor Printer	$ 795.00
CBM 8010 IEEE Modem	$ 279.00
CBM Datasette	$ 74.95
CBM-IEEE Interface Cable	$ 39.95
IEEE-IEEE Interface Cable	$ 49.95



Atari

Atari 400 16K	$ 399.00
Atari 800 16K (incl. BASIC cartridge)	$ 899.00
Atari 410 Recorder	$ 99.95
Atari 810 Disk Drive	$ 599.95
Atari 822 Thermal Printer	$ 299.95

NEC Spinwriter Printer

NEC 7730	$3085.00
NEC 7710	$3085.00
NEC 7720	$3610.00
NEC 3510	$2290.00
NEC 3500RD	$1895.00

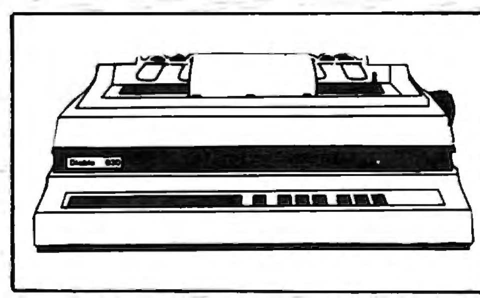
Tractor Feed Options are available

Amdek Monitors

Video 100G (Limited Quantity)	$ 179.00
Video 300G	$ 200.00
Color 1	$ 449.00
Color 2	$ 999.00

Diablo 630 Printer

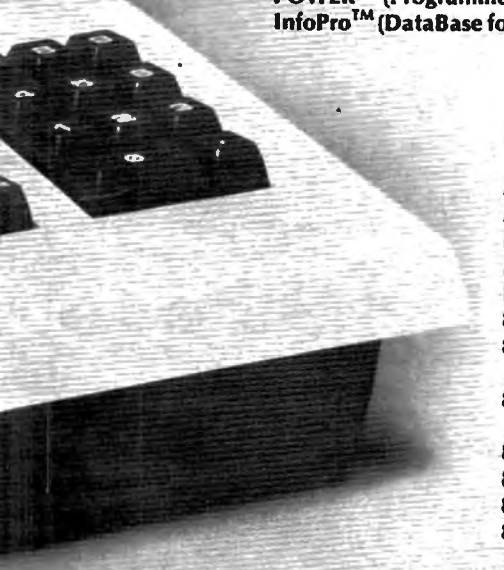
Diablo 630	$2710.00
Tractor Option	$ 350.00



Professional Software

WordPro™ — A Family of CBM Word Processing Programs

WordPro 2 Plus	$ 199.95
WordPro 3 Plus	$ 295.00
WordPro 4 Plus	$ 450.00
WordPro 5 Plus (for CBM 8096)	$ 450.00
The Administrator (DataBase for CBM)	$ 650.00
POWER™ (Programmer's Utility ROM)	$ 89.95
InfoPro™ (DataBase for CBM)	$ 295.00



Epson Printers

MX-70	$ 299.95
MX-80 w/graphtrax	$ 645.00
MX-80 FT	$ 745.00
MX-100 FT	$ 945.00
INTERFACE CARDS	
8141 RS-232 Interface Board	$ 75.00
8145 RS-232 Interface Board w/2K Buffer	$ 149.00
8151 RS-232 Interface Board w/X/ON-X/OFF	$ 170.00
8161 IEEE Interface Board	$ 55.00
8131 Apple Interface Card	$ 85.00
8232 Apple Interface Cable	$ 35.00
8220 TRS-80 Cable	$ 35.00

CMD Mupet

MC-800A Mupet Controller (Multi-User Controller for CBM Computers)	$ 995.00
CM-100 Channel Module	$ 250.00
Printer Module	$ 350.00

Qume

Spring 9/45	$2495.00
Tractor Option	$ 210.00

WordPro, POWER and InfoPro are registered trademarks of Professional Software Inc.

NEECO
679 Highland Ave.
Needham, MA 02194

(617) 449-1760
TELEX: 951021

MON-FRI 9:00 - 5:30 E.S.T.
MASTERCARD • VISA

Table of Contents

August 1982 Vol. 4, No. 8

Title	Author	Page	Guide To Articles And Programs: Computer Specific	Guide To Articles And Programs: Multiple Computers
Features				
The New Wave Of Home Computers	Tom R. Halfhill,	18		•
Household Budget Manager	Richard Kalagher,	39		•
Word Games	August Shau,	54		•
A First Look At The Commodore 64	Tom R. Halfhill,	60	P	
Don't Forget Testing	Dave Johnson,	66		•
Education And Recreation				
Atari Sketchpad	Todd Mowbray,	72	AT	
Chemistry Lab	Joanne Davis,	75	AP	
Guess That Animal	Daniel Hastie,	84		•
Reviews				
Two VIC Word Processing Programs	Harvey B. Herman,	93	V	
BASIC A+	Charles Brannon,	95	AT	
Columns And Departments				
The Editor's Notes	Robert Lock,	6		•
Ask The Readers	Robert Lock, Richard Mansfield, and Readers,	10		•
Computers And Society	David Thornburg,	14		•
The Beginner's Page: Structured Programming	Richard Mansfield,	34		•
Friends Of The Turtle	David Thornburg,	69		•
The World Inside The Computer: Build Your Own Computer Friend	Fred D'Ignazio,	78		•
Learning With Computers: The PILOT Language	Glenn Kleiman,	88		•
Telecommunications: Choosing A Modem Part II	Michael Day,	114		•
Machine Language: Shreds And Patches	Jim Butterfield,	126		•
Insight: Atari	Bill Wilkinson,	145	AT	
The Journal				
VIC Communications: The RS-232 Interface	Jim Butterfield and Jim Law,	99	V	
The Keyprint Compendium	Charles Brannon,	103	P	
Screen Saver	David Wine,	107	P	
Atari Video Graphics And The New GTIA Part II	Craig Chamberlain,	108	AT	
Energy Monitor	Linton S. Chastain,	116	C	
Animation And P/M Graphics	Tom Sak and Sid Meier,	119	AT	
Apple Manager: An Alphanumeric Data Manager	Robert J. Beck,	130	AP	
Pet Auto Repeat	Art Hunkins,	139	P	
VIC Curiosities	Doug Ferguson,	140	V	
A Light Pen For Under $10	William Hale,	141	V	
Substring Search Utility	Edward C. Smith,	142	At	
Electric Eraser	Louis F. Sander,	153	P/V	
System Clock For The Atari	Bill Zimmerman,	156	AT	
Inner BASIC	Jim Butterfield,	158	P	
Copy 2031 Files	G. H. Watson,	160	P	
VIC-Key	Thomas Henry,	164	V	
The FORTH Page: Speed Search	Richard Mansfield,	169		•
COMPUTE!'s Listing Conventions		**168**		
CAPUTE!: Modifications Or Corrections To Previous Articles		**173**		
New Products		**175**		
Advertisers Index		**192**		

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. Second class postage paid at Greensboro, NC 27403 and additional mailing offices.

ISSN 0194-357X.

A Small System Services, Inc. Publication
COMPUTE!

The Editor's notes...

Robert Lock, Publisher/Editor

Software Warranties Revisited

In recent editorials we've raised the question of software and product warranties. In February, we quoted an interesting letter written by John Navas II, a concerned subscriber. We're still quite interested in *your* feedback. One note of inquiry we received was from the Federal Trade Commission. They too, it seems, are concerned about the same questions. We would like to keep the forum going, and welcome input from both buyers and sellers. Here's an update from John:

Thank you for quoting my letter on software warranties in your February editorial, and for inviting comment on this important issue. Since then I have learned that some firms justify "as is" warranties as a defense against potentially large product liability damage claims. The following is my rebuttal to that justification:

As a businessman I understand concerns for the potential risks of product liability and consequential damages. Such concerns, however, do not justify a disclaimer of all *warranties. To do so confuses product* warranty *with the separate issue of product* liability. *It is a simple matter to write a product warranty which limits liability to the price paid for the product.*

With such a warranty, a responsible business should have little to fear, particularly when there is not clear negligence or knowing concealment of product defects. Presumably **COMPUTE!** *readers share my lack of sympathy for negligent conduct or failure to disclose known defects to prospective purchasers.*

All I ask of suppliers is that they be diligent in pre-sale quality control, that they promptly remedy without charge all product defects identified within a reasonable post-sale period and that they provide a written warranty to that effect. If a supplier disclaims all warranties, including implied warranties, it forces its customers to rely solely on trust.

Unfortunately, my own experience has shown that such trust can be unwarranted, and that there is no reliable way to anticipate how a supplier will behave. Several programs that I have purchased recently from reputable suppliers proved within the first few weeks of use to have serious defects. In at least one case I discovered that the supplier had previously known about some of the defects. Only once have I been able to get a defect repaired without charge. Sometimes suppliers have advised me that no corrections would ever be made. Sometimes "new" versions would become available, but only at a substantial additional charge. Either way, suppliers explicitly relied upon their "as is" warranty disclaimers.

Some suppliers do provide reasonable software warranties. One example limits liability to the price paid, notes that software is not warranted to be error-free, but entitles the original purchaser to replacement or repair of defects without charge (or to a full refund) within the warranty period. These suppliers deserve our business.

Sincerely,
John Navas II

Personnel Updates

June has been an interesting month for corporate presidents. The presidents of Atari, Inc., and Commodore, Inc., are both leaving their respective positions. Peacefully too, from all we can tell. Atari's president is leaving to establish a personal computer venture... Commodore's to establish an innovative computer retailing plan. We wish them both well in their newest endeavors. We'll keep you posted on their replacements.

Documentation Update: Krell LOGO

It's not our policy to mention specific vendors on this page, but in this case I think it's warranted. A July column panned Krell's LOGO documentation, and it slipped through our editorial review in a fashion that I don't think was quite fair. We have no qualms about describing the realities of a given situation, but we always verify and double check. In this case we didn't. If we had, we would have discovered that Krell has substantially increased their package documentation (prior to our comments, by the way). Our apologies for not pointing this out last month. ©

Introducing the PERCOM Alternative to ATARI Disk Storage

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

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

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

- Operate in either single- or double-density storage mode using Atari DOS 2.OS. In double-density you can store almost 184 Kbytes (formatted) on one side of a 40-track diskette.
- Connect your Percom RFD first-drive system directly to your computer or connect into your system through your Atari 810 Disk Drive.
- Add an RFD first-drive system with its versatile four-drive controller, then connect as many as three more low-cost RFD add-on drives.
- Write application programs that can query and set up your system to operate a different type drive at each cable position — that can even change configuration as the program executes.
- Get quality and state-of-the-art capability at competitive prices. Percom first-drive RFD systems are priced from only $799, first add-on drive is only $459. Cables included.

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

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

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

PERCOM

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

PRICES AND SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE.

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

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

Staff of COMPUTE!

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

Associate Editors

Jim Butterfield, Toronto, Canada
Harvey Herman, Greensboro, NC
Fred D'Ignazio, c/o COMPUTE! P.O. Box 5406, Greensboro, NC 27403
David Thornburg, P.O. Box 1317 Los Altos, CA 94022

Contributing Editors

Marvin DeJong, Dept. of Mathematics-Physics The School of the Ozarks Pt. Lookout, MO 65726
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

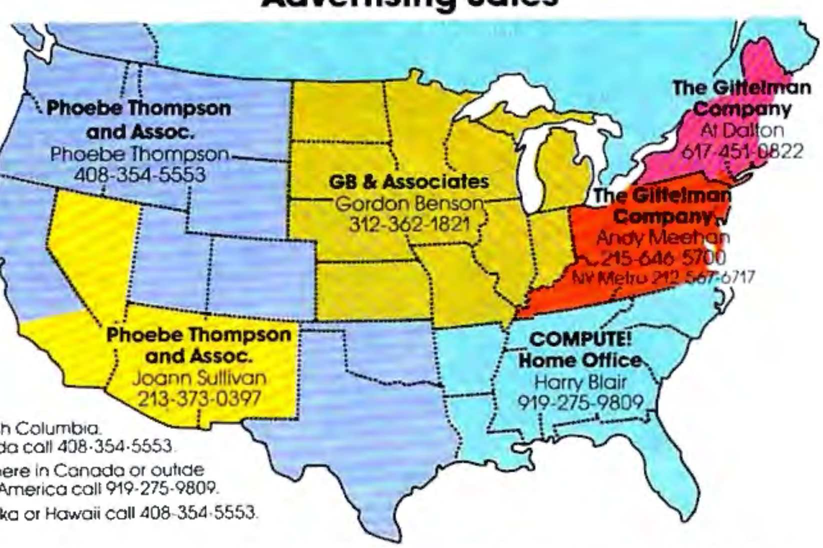
Corporate office:
625 Fulton Street,
Greensboro, NC 27403 USA

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 Assist ınt

Advertising Sales



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

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
AL DALTON

The Gittelman Company
Summit Office Center
7266 Summit Avenue
Fort Washington, PA 19034
ANDY MEEHAN

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

Coming In The September Issue Of COMPUTE!

Turtle PILOT For Atari, Commodore, and Apple

Software Voice Synthesis For Apple and PET

A $20 VIC Digitizer

PET Pointer Sort

Dramatic Atari GTIA Demonstrations

Turtle PILOT For Atari, Commodore, and Apple

ADD POWER™ TO YOUR COMMODORE COMPUTER

$89.95

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

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

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

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

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

Ask The Readers

Robert Lock, Richard Mansfield, And Readers

COMPUTE! *welcomes questions, comments, or solutions to issues raised in this column. Write to: Ask The Readers,* **COMPUTE!** *Magazine, P.O. Box 5406, Greensboro, NC 27403.* **COMPUTE!** *reserves the right to edit or abridge published letters.*

Atari Tape Suggestions

I have an Atari 800 with a cassette recorder. I have been having one heck of a time getting good CSAVE's on my tapes. I use good tapes and try my best to properly record, but it has gotten to the point where I'm afraid to start on any large program because if the CSAVE is no good, CLOAD will wipe me out when I want to check if the tape is good. I would like to hear of any good solutions to the problem. I have had the recorder checked and I was told it was working normally.

Albert Jacobs

Proper maintenance of a cassette recorder is essential. The heads should be regularly cleaned and demagnetized (solvents and tools for these jobs are available at any electronics supply store). Also, some computers require special recording techniques.

The Atari stores programs as FSK tones. The only direct effect this has on you is that you should not use computer digital tape or chromium dioxide tape. Instead, use a good quality, "low noise" audio cassette. Uniform tape speed is also important. Avoid tapes which rattle or seem to stick. Some people find that it helps to fast-forward and then rewind a tape before using it for the first time.

Another thing that can help is to issue an LPRINT command (even if you don't have a printer) before you CSAVE. This insures that certain operating system flags are set correctly.

For additional information, see "Atari Tape Techniques," **COMPUTE!**, *July 1981, #14.*

VIC Upgrades

Will it ever be possible to upgrade the VIC with any of the new display, sound, and microprocessor chips in the new Max and the Commodore 64? Also, with the Superexpander cartridge in the VIC, it is easy to mix normal (but not reversed) text and graphics characters with high resolution graphics. Is it possible to program these characters? (The bit map for the graphics screen does not use all the available memory.)

D. M. Lane

Transforming the VIC into the new Max and 64 (see "The New Wave of Home Computers" in this issue) would be a major technical undertaking – it is not a simple chip substitution. Highly placed sources at Commodore have hinted, however, that there is a possibility that a special cartridge for the VIC will be manufactured. This would bring to the VIC all the synthesizer effects available on the MAX and the 64. This is, however, only a possibility.

Concerning programming characters, the Superexpander uses up nearly all of the characters, but in theory you could have up to 50 more. To do this, you'd need to see what characters the cartridge was using to bit-map the screen and then see what's left over.

Recover From NEW On VIC

If you type NEW, your program is still in the VIC. "Recovering From NEW On Apple and CBM" (**COMPUTE!**, May 1982, #24, pg. 135) showed how you can restore the program for PETs and Apples. The procedure is exactly the same for VIC except that the POKEs are different locations.

The first POKE should be POKE 4098,16 (instead of POKE 1026,4). The second POKE goes into location 4097. Determine the correct number to POKE by following the article's instructions for the second POKE. Also adjust the variable pointers as the article suggests. The locations are the same as for the Upgrade ROM set (locations 42 and 43). In the program which adjusts the variable pointers, use 4096 instead of 1024.

Minoah Tam

SuperPET Users Group

I am the proud owner of a SuperPET. 4040 disk drive, and a 4022 tractor printer. Would any SuperPET owners out there happen to know of any user group devoted to the SuperPET? If there isn't I would like to organize a SuperPET users group here in the Tri-state are.

P. V. Skipski

P. V. Skipski
4782 Boston Post Rd.
Pelham, NY 10803

VIC Super Expander Hints

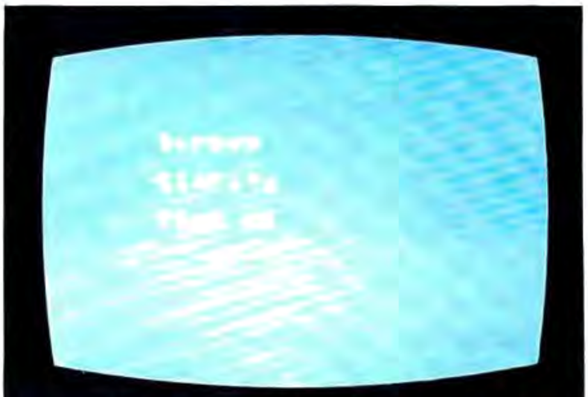
This is for other readers who have spent from $65-$80 to acquire the Super Expander for the Commodore VIC, and then found most of their games did not work properly.

THE MOSAIC RAM SYSTEMS FOR ATARI*

CLEARLY THE BEST

MOSAIC 32K RAM

THE SCREEN CLARITY TEST



THE COMPETITION

YOU CAN SEE THE MOSAIC DIFFERENCE

WHAT THE EXPERTS HAD TO SAY

A.N.A.L.O.G. 400/800 MAGAZINE said in a 32K RAM board comparison test: "The Mosaic 32K RAM showed no sign of interference and gave the best screen clarity" and "Mosaic uses what we feel are the highest quality components which may improve reliability".

INTERFACE AGE said after replacing the Atari memory board with a Mosaic 32K RAM: "Once in place there is no noticeable change in screen clarity" and "in view of its excellent performance it should be a serious choice for those Atari owners intending to expand their memory."

Each Mosaic RAM board gives you more than just the best screen clarity but also the best in reliability, flexibility, and compatibility. The Mosaic RAM systems offer you the best in every way — these features prove it.

- Works in both the Atari 400*& 800*computers.
- 4 year guarantee.
- Complete instructions.
- Test cycled 24 hours for reliability.
- Gold edge connectors for better reliability.
- Quick no-solder installation.
- Full flexible memory configuration.
- Can be used with 8K, 16K and future products.
- Designed to take advantage of Atari 800's superior bus structure.
- Designed for inter-board communication in Atari 800.
- Allows Atari 800 to have 2 slots open for future expansion.
- Always the best components used for superior screen clarity and reliability.
- Low power design for safety and reliability.
- Available companion board ($5) to allow running The Mosaic RAM systems independent of other boards.

THE MOSAIC RAM SYSTEMS FOR THE SERIOUS ATARI OWNER.

THE MOSAIC 32K RAM

For the serious Atari owner. This is the 32K RAM board you've been hearing about. It has every feature you could want from a RAM board and more features than any other board offers. Each Mosaic 32K RAM comes with complete instructions so in a few minutes you will have expanded your Atari 800 to 48K RAM. The Mosaic 32K RAM works as well in the Atari 400, but we suggest the NEW Mosaic Expander.

THE MOSAIC EXPANDER

This is the most effective way you can expand to 32K RAM for your Atari 400 computer. And at almost the 16K price! The revolutionary Mosaic Expander is a memory board with 16K RAM in place and open slots to add 16K more from the Atari 16K board that comes with your Atari 400. Each board comes with complete instructions so in a few minutes you will have expanded to 32K RAM.

*Trademark of Atari, Inc.

MOSAIC ELECTRONICS

P.O. Box 708 Oregon City, Oregon 97045
503/655-9574

On the basic 5K VIC, the memory locations (51-52 and 55-56) point to the end of memory when powered on. If you PEEK (51 or 55) you will notice that both of these locations contain zero.

Once the Super Expander is plugged in and the same two locations namely (51 and 55) are PEEKed, you will notice the value has changed from zero to 120.

What this means is that any program which is going to use its own character set must have the following instructions:

POKE 52,28 – Set String High Pointer
POKE 56,28 – Set Memory High Pointer
POKE 51,0 – Set String Low Pointer
POKE 55,0 – Set Memory Low Pointer

Also there is a major bug when the Super Expander is installed. When the RUN/STOP and RESTORE keys are hit, followed by the direct command ?FRE(X), the amount of free memory shows 3104. Since this is less than the basic 5K VIC without the Super Expander, you feel cheated.

The only way I have found to correct this loss of memory is to do a SYS64802 after the RUN/STOP and RESTORE. I have reported this bug to Commodore, but to date have not received an answer.

I hope these tips will help other people with the VIC Super Expander.

The code to look for in any game which is going to use its own character set which will not work will look like the following:

```
50 POKE 52,28: POKE 56,28
60 POKE 51, PEEK (55): CLR
70 CS=256*PEEK (52) + PEEK (51)
```

To correct this code, add the following instruction:

```
55 POKE 51,0: POKE 55,0
```

William D. Collins

Tape Dents

I would like to warn any readers who experience sudden and unexplained program failures when loading from an Atari 410 program recorder of a potential problem. The same is true of other cassette systems as well.

If you should forget to depress the STOP/EJ. key after loading in a cassette, the flywheel shaft remains in contact with the pinch roller assembly. This contact and pressure will actually put a "dent" in the cassette tape that will remain. When you try to load the tape data into the computer later on, that particular section, where the depression is, will not make contact with the playback head and the program will crash with an -ERROR 143 at line 0 indicator.

Fortunately, rewinding the tape and storing for several days will usually correct the problem. Unfortunately, it doesn't always iron itself out and if there was not an extra CSAVE or backup tape made, all data is lost!

Randy T. Agee

Earlier Atari GTIA Chips?

Thought some Atari users might be interested in learning of my experience. I've been reading reports that say that the GTIA, a more advanced graphics chip, might be in machines "purchased after January 1982." False!

The date must go back further than that. My 400 was purchased in November 1981 and, using the following test, I proved the GTIA was in my machine. Try it!

```
10 GRAPHICS 11: REM OR 9 OR 10
20 GOTO 20
```

Ed Pomelear

If you get a black screen, you've got the GTIA chip in your machine. If it's blue, you've got the older CTIA chip. For in-depth information on the GTIA, see "Atari Video Graphics And The New GTIA" in this issue.

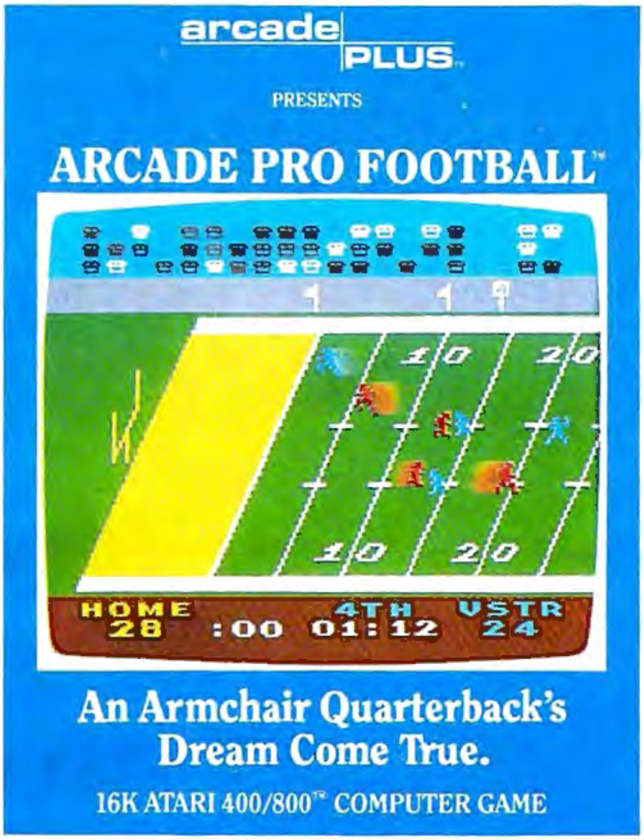
VIC Zenith Jitters

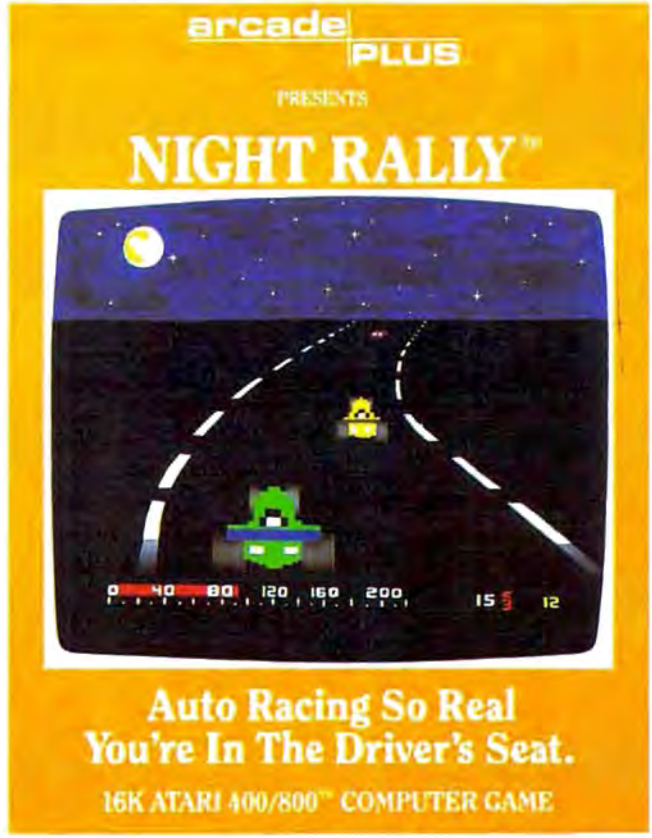
I just recently purchased a VIC and own a Zenith TV set. The computer does not work on the Zenith because of vertical hold problems. There is a single POKE command (POKE 36864,133) which corrects this problem. Some preprogrammed tapes and some cartridges prevent this command from being entered. I would appreciate it if someone could come up with a permanent fix for this. Is Commodore working out a solution?

David St. Romain

Several readers have mentioned this problem. It appears that some recent Zeniths (and reportedly some Sylvania sets, too) get a bad case of the "flutters" when attached to computers. The culprit is evidently a new circuit in some televisions which are "auto-setting." And the problem isn't limited to VIC – any computer can create these unpleasant effects with these models. Aside of the POKE solution you mention, Zenith has sent a technical notice out to its service centers with instructions on curing the problem. It involves a simple disconnection of a yellow wire and your local authorized Zenith center should be able to perform the modification.

The most recent VIC cartridge games and other recent programs released by Commodore have an "interlace mode toggle" built in. Pressing the F8 key will switch back and forth between the two screen modes and you can see which setting produces the best results on your set. Programs with model numbers between 1901 and 1908 do not have this toggle feature, but most of the later programs do. ©





IN A LEAGUE ABOVE THE REST.

When it comes to realistic, animated sports action, Arcade Plus games are in a league above the rest. State-of-the-art games designed *exclusively* for the Atari® 400/800™ Home Computers. So you know you're getting the best game for your money. Not just a *version* of a game designed for other computers.

Sweep right, screen pass left or go for the bomb in ARCADE PRO FOOTBALL.™ You call the shots and control all the action during 60 minutes of exciting gridiron play. Experience all the fun and excitement of real football including penalties, fumbles, pass catching and a crowd to cheer you on.

ARCADE PRO FOOTBALL™ pits you head to head against the *computer* or another player. No other video football game on *any* system offers you that kind of challenge.

Test your driving skills to their limits in NIGHT RALLY.™ Climb *into* the driver's seat as you race against some of the best computer drivers in the world. On some of the most challenging race courses imaginable.

Cross the finish line ahead of the pack and your initials will be immortalized in NIGHT RALLY'S™ exclusive racing Hall of Fame.

And more Arcade Plus sports games are on the way. Exciting, animated games like ARCADE BASEBALL™ (coming this summer) that are setting *new* standards in video sports game realism. And are in a league above the rest in play value. See them at your local Atari® computer dealer. Or write for our latest games catalog and dealer list.

arcade PLUS™

WE GIVE YOU A BETTER GAME.

Arcade Plus, Inc. 3916 State Street Suite 1C Santa Barbara, CA 93105

A Monthly Column

Computers And Society

David D. Thornburg
Associate Editor

Stranger Than Fiction ...

I once read that the reason truth is stranger than fiction is because fiction has to make sense.

I guess that's so, because no fiction writer could ever come up with some of the developments we have seen in the personal computer industry in the past few months. As the computer market continues to expand into mass merchandising outlets, we will be getting used to some pretty strange things.

For example, where can you find the three B's under one roof (bytes, baud, and Barbie dolls)? If you said Toys "R" Us, move to the head of the class. While this nationwide discount toy store has been selling video games for quite some time, I had once said that the beginning of the true mass market could be pinned to the date TRU decided to carry personal computers.

Nestled away among the video games and cartridges, one can find the Commodore VIC, the TI 99/4A and soon(I'm told), the Atari computers. Remember a year or so ago when I said that the TI computer was dead?

I was wrong. (Boy, was I wrong!)

Shipments of the 99/4A are up there with Apple. At a retail price (including modulator) of under $300, Toys "R" Us will probably do a brisk business in this product.

But how does a discount toy store handle esoteric products such as disk drives, expansion RAM, etc.? And, especially, how does this type of store sell these products when the sales staff doesn't know that a parallel port doesn't have parking spaces for boats?

Good question.

It is unrealistic to expect rapid turnover sales personnel to become conversant with computers overnight. As one sales person told me, "This is different from selling video games. All people ask then is whether a certain game lets them play Pac Man, or whether the game will hurt their TV. With computers we've had people in the store asking about "expandability."

To tell the truth, I'm not worried about the folks who know enough to ask about expandability. Anyone who knows that much is well on the way to making a careful purchase decision. But what about the true neophyte – the person who sees the computer as a mass merchandised electronic appliance to grace the den next to the VCR and projection TV? How is this person to make an intelligent decision on selecting a computer that retails for under $500? The traditional computer stores seem to be dropping these products, especially in areas serviced by discount houses. After all, why should someone in a MicroAge or Computerland store, for example, spend an hour making a sale, only to have the customer end up buying the machine at the local discount house?

For the first time since the revolution began, we are starting to see true product differentiation. The Apple-priced systems continue to be the mainstay of the traditional computer stores, and the low end products are being pitched to a broader audience from the traditional consumer outlets. The sad part is that the broad audience is the one that needs the most help in the purchase decision.

One solution I can think of is for the manufacturers to combine efforts and publish a "generic personal computer purchase guide" that lets people know what the personal computer is all about. Tons of these books should be shipped to all the mass outlets and handed free to anyone thinking about buying one of these machines.

If the manufacturers would also adopt a uniform format for listing specifications (similar to that of the stereo or automotive manufacturers), even the most nontechnical among us would be able to do effective product comparisons without taxing the skills or patience of a sales person who knows no more than the customer.

The manufacturers have an even greater obligation to make their system setup easy for the neophyte. When I opened my TI 99/4A, the "Read This First" manual was at the very bottom of the stack. While the setup manual was very well written, it should have been taped over the keyboard so it couldn't be missed.

I am excited about computers being bought by millions of people. I am excited to see these products in the mass market outlets. But I am afraid that the manufacturers of these micros don't realize how much help their customers really need.

Notes On Education ...

Sterling Swift Publishing Co. (1600 Fortview Rd., Austin, TX 78704) has just published a new edition

of their Apple II educational software directory ($14.95). This is a delightful replacement for their first edition, for it covers educational software offerings from 128 sources. The spectrum of software spans elementary to college education, and covers every topic from drill and practice to games. Vendor listings are divided into chapters covering traditional educational publishers, non-commercial vendors, etc. With 350 pages devoted to this important field, the Swift directory is a valuable investment for Apple using educators. I was sorry to see that the Mind Toys games from Automated Simulations were not listed, but no software directory can be expected to be perfect.

On another topic, this column has presented my views on the uses of computers in education. This view is limited by my personal bias and is, thus, incomplete. In particular, I am not a strong proponent of the use of computers as teacher replacements. It seems to me that computers can be more effectively used for other tasks. However, when one considers the use of computers with older students (those in college, for example), the idea of testing "book knowledge" with computer simulations seems quite valuable. This is especially true in the physical sciences where the computer simulation might be seen as an intermediate step from the lecture to the laboratory.

Just as Seymour Papert is well known for his view of the computer as a learning tool for the child, so Alfred Bork is known for his view of the computer as a medium of instruction. Bork's perspectives on computers as instructional tools are presented in *Learning with Computers* (Digital Press, Bedford, MA), a book-length collection of papers he published over a ten year period. Since each paper in this book is self-contained, there is a certain amount of unavoidable repetition. Much of the work by Bork and his colleagues was carried out using large computers, but he is also very interested in the use of personal computers as well as incorporating "intelligent" video disk technology into the classroom.

I have often classed Bork's view of computers in education as being quite conservative. This may be a bum rap. In fact, he believes that, once computer-based instructional materials are in widespread use, there will be radical changes in traditional educational institutions. With all courses individualized, there is no need for traditional semester boundaries, etc.

But what about teachers? If students are being taught by computers, what are the teachers going to be doing? According to Bork, their role may shift from being the deliverers of courses to the designers and developers of courses, with some time devoted to working with students who have individual learning problems. While this scenario might become reality, I find it flawed. Teachers cannot develop effective course materials in a vacuum. I know of no way to replicate the wealth of information one gains by teaching something to a class full of students.

Disagreements aside, I think that Al Bork is an articulate spokesman for his brand of classroom computer use, and that his book deserves a broad audience among educators who are using computers in any capacity. ©

Computers with more power and lower price tags than ever before will be coming on the market soon. These are the highlights of two important trade shows held recently in Chicago and Houston.

The New Wave Of Home Computers

Tom R Halfhill, Features Editor

Back in the late 1940's, when a computer weighed several tons, cost millions of dollars, and had to be air conditioned to keep its thousands of vacuum tubes from melting, somebody estimated that by the end of this century in America there might be as many as 100 computers.

In the last two years, Sinclair Research alone claims to have sold more than 300,000 of its tiny computers in the U.S. And if that's not enough to demonstrate how fast things are changing, you should have spent a week in June visiting the Summer Consumer Electronics Show (CES) in Chicago and the National Computer Conference (NCC) in Houston.

It was new microcomputer hardware that stole the show – both of them. Even IBM, the mainframe grand-daddy itself, leaned heavily toward microcomputers at its NCC booth – which, incidentally, was plopped right next to Apple Computer's. Judging just from the size and extravagance of both booths, it was impossible to tell which company was the establishment giant and which was the cocky upstart. The home/personal computer firms, banished to back aisles at the NCC until just a couple of years ago, finally have achieved their place in the sun.

Expanding Power In Ever Smaller Packages

The big news at both shows was the latest developments in a trend that started in the late 1940's: packing more power into smaller computers that cost less. Like a science fiction fantasy run wild, the Incredible Shrinking Computer is reaching almost ridiculous proportions, beyond belief even ten years ago. New computers and peripherals from Commodore, Sinclair, Epson and others will put startling amounts of computer power at the fingertips of consumers for less money than ever before. Very soon consumers shopping for home computers not only will face the question: How much power can I afford?, but also: How much power do I really need?

The Commodore Challenge

Commodore, which displayed no less than five new machines that drew "Ooohs!" and "Aaahs!" at both shows, seems to be addressing the question by offering the industry's most complete vertical selection. In prices ranging from $179.95 to $2,995, Commodore can sell you anything from a game machine with marginal BASIC programmability to a full-blown business system with built-in dual disk drives, 256K RAM, and 16-bit CP/M capability (and a wide assortment of computers in between). More importantly, each computer is among the most advanced and least expensive in its class. Commodore can do this more easily than others because it is among the microcomputer industry's few "vertically integrated" companies – that is, it does everything from designing and manufacturing its own chips to assembling the computers and writing the software.

"We want to cover the entire market," says Kit Spencer, Commodore's marketing vice president. "We're the microcomputer specialists." Some believe a hazard of this approach is that the lower-end game machines might "tarnish" the button-down image of the upper-end business computers. In other words, *Gorf* and *VisiCalc* don't mix. Atari, for example, has struggled to convince people that its computers are more than just sophisticated game machines. Commodore, though, unlike Atari, did not establish itself by selling several million video game machines, and already has a solid toe-hold in the small business marketplace, especially abroad.

What Commodore most definitely has started with its new line is a real dogfight in the under-$1,000 home computer market, not to mention the under-$500 market. Prices of competing machines suddenly dropped, though of course spokesmen denied the changes were a response to Commodore's salvos. Atari, for instance, announced at CES a $50 cut in the list price of its 400, from $399 to $349. Rumors of a new Atari computer, perhaps to be called the model 600 and falling between the 400 and 800 in terms of both features and price, turned out again to be just rumors.

Color Computer Upgrades

Radio Shack is fighting to stay in the pack, too. The TRS-80 Color Computer went on sale for $299 from its regular $399 in early summer, and a Tandy spokesman at NCC said that, although the sale was not advertised as such, it was sort of a "close-out." He said the new Color Computers being sold for the usual $399 will come with 16K RAM instead of the previous 4K (but still will lack the Extended BASIC).

The new Sinclair ZX Spectrum, a 16K color-and-sound computer selling for around $200, but not yet available outside Great Britain, will cloud the under-$500 picture even further. Meanwhile, peripherals manufacturers, hampered by the mechanical rather than solid-state nature of their devices, are battling to bring the prices of their products down to something that looks more reasonable alongside the prices of the new computers. Word is they're succeeding, and that we'll soon see disk drives and letter-quality printers at unheard-of prices. Software development isn't standing still, either, with Atari announcing a partnership with one of the entertainment industry's most admired clusters of creative geniuses for the purpose of developing new computer-based games. Here, then, is a rundown of the most exciting news from the CES and NCC shows:

Computers: Commodore

We have to start with Commodore. With five new computers scheduled to hit the market by the end of the year, it's certainly the most active company. Starting at the bottom of the line, Commodore showed its Max Machine (pre-production versions were called "Ultimax"). For $179.95, the Max is a cartridge- and cassette-driven game machine targeted at the Atari VCS, the new, more sophisticated Atari 5200, the Mattel Intellivision and the Odyssey. But Commodore also is billing Max as an entry-level computer and a music synthesizer. A plug-in BASIC cartridge will give game players a taste of computing, although only about 1K of RAM is available. Still, it's possible to save programs on cassette, and Max has an integral 66-key bubble membrane keyboard identical in layout to the popular VIC-20.

The electronics are not at all like the VIC-20's, however. While the VIC, like earlier Commodores, has a 6502 microprocessor chip for its CPU (Central Processing Unit), Max has a 6510. The 6510 is a new chip designed by MOS Technology, a subsidiary of Commodore. It is nearly identical to the 6502, but has additional input/output lines to handle the processing required by the new system. Max also boasts two more new chips: a display chip that puts 40 columns by 25 lines on the screen in text mode with 16 colors and high-resolution graphics, and a sound synthesizer chip known as SID (Sound Interface Device). SID supports three voices with a nine-octave range, and must be heard to be believed. Demo programs playing classical music sounded remarkably close to Yamaha keyboard synthesizers costing several times as much as Max.

None of the new Commodore computers replaces the VIC-20, which at $299 neatly fills the gap in the low-end market.

These same three chips – the 6510 CPU, the display chip, and SID – also are the central configuration of the new Commodore 64 (see part one of "Commodore 64: A First Look" elsewhere in this issue). Naturally, this is no coincidence. The Commodore 64 is designed to be upward compatible with Max. That is, once game-players get their initial taste of computing on the Max and exhaust its possibilities, they can upgrade to the $595 Commodore 64 and keep using all their game cartridges, joysticks, and the cassette tape drive: all will work on the 64! This means a family can buy a very sophisticated yet reasonably priced video game machine, freely invest in game cartridges, get a crack at simple programming, and later move up to a full-blown 64K RAM computer if they wish, without obsoleting their software and other accessories. In addition to its capabilities, this feature should give Max a strong edge in the video game market.

Commodore promises that both Max and the 64 should be available by the time you are reading this. Similar promises when the VIC-20 was introduced took several months to come to pass, so we'll have to wait and see.

None of the new Commodore computers replaces the VIC-20, which at $299 neatly fills the gap in the low-end market between the $179 Max and the $595 Commodore 64. VIC owners wishing to upgrade to a 64 will find that most of their peripherals will work as is, and that their programs will convert with little difficulty.

Commodore is rounding out its home computer line with the new P series, the third-generation PETs. The P128 offers 128K RAM, the same color graphics and sound capabilities as the 64, and will sell for $995. At the higher end are the B and BX series, 80-column professional computers which offer everything from built-in dual disk drives and monitors to multiprocessing, and which will sell for

Not since Gutenberg invented the printing press....

has there been so much talk about printing!

The Smith-Corona TP-1 Daisy Wheel Printer

$695.00

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

Act Now: Limited Supply, Low Cost

In the 15th century, a German craftsman and inventor named Johannes Gutenberg originated a method of printing from movable type that was used without significant change until the 20th century.

Today, **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.** Gutenberg would surely approve of the **TP-1,** 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.

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.

Don't delay. Order your **TP-1** TODAY at the low, low price of **$695.**

Micro-Printer Marketing offers same day shipping, nationwide service and invites dealer inquiries. Catalogues available. No shipping charges on prepaid orders.

Additional daisy print wheels $4.95

Additional ribbons $2.95

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

$1,695 and $2,995, respectively (see sidebar for more details on the P, B and BX series). These computers are scheduled to be available this fall.

In tune with Commodore's new corporate alignment and marketing philosophy, the Max and VIC-20 will be sold through "mass merchants" – catalog showrooms, department stores and other non-specialty retailers – while the 64 and higher machines will be sold exclusively through established Commodore dealers. However, Commodore spokesmen left open the possibility that the 64 eventually may be available through mass merchants also.

Peripherals makers are working hard to produce add-ons that don't cost twice as much as the new computers they'll be plugged into.

Other Entries

Don't get the idea that Commodore was the only company at CES and NCC with exciting new computers to show off. Sinclair offered a tantalizing glimpse of its ZX Spectrum, and Epson had plenty of its HX-20 battery-powered portables for people to fiddle with (see sidebars). Timex also debuted its $99.95 Timex Sinclair 1000, a Sinclair ZX-81 which Timex is licensed to market in the U.S. under a joint name. The Timex is identical to the ZX-81 except for its 2K of RAM, twice as much as the standard Sinclair. It is, of course, fully compatible with all Sinclair add-ons, including the plug-in 16K RAM module, also sold by Timex. According to a Sinclair representative at CES, if Timex sells a certain volume of the computers by a deadline several months hence, it wins the right to market the ZX Spectrum in this country. Otherwise, Sinclair will market the Spectrum mail-order, just as it has been selling the ZX-81s. Sinclair hopes to export the Spectrum to the U.S. by the end of this year.

Peripherals...

There should be big news in this area in coming months, with reports of upcoming Tandon disk drives for under $300 and possibly a letter-quality printer from Epson for around $400. As mentioned, peripherals makers are working hard to produce add-ons that don't cost twice as much as the new computers they'll be plugged into. However, nothing of the sort was shown at CES or NCC. What was seen from various manufacturers were prototypes of new micro-floppy disk drives,

POOL 1.5(TM)

ATARI(TM) OWNERS,

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

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

New! POOL 400 in Cartridge form for your 16K Atari 400/800 $39.95, requires joystick.

PRICE: $34.95

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

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

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

both in 3-inch and 3-1/2 inch versions. A compatibility conflict already is arising, with Sony and other Japanese companies favoring the 3-1/2 inch size. Several domestic firms are trying to standardize among themselves on the smaller disks, some say as an effort to beat out the Japanese. It might be too late, since Sony has solidified its position by signing a $300 million contract to supply Hewlett Packard with the larger micro-floppies. Regardless, either size has potential advantages for home users.

For one thing, they may be cheaper than the 5-1/4 inch drives now standard for home computers. Sony is selling the 3-1/2 inch drives for $400 each in small quantities, and for significantly less in large quantities. Computer manufacturers buying several thousand of the micro-floppies for private labeling might eventually get them on the market for less than the $500 to $700 that current 5-1/4 inch drives cost. And despite their smaller size, the micro-floppies are much faster and actually store more data than larger disks. The Sony model, for example, crams 218K single density on one side of a disk, and 437.5K double density. That's more than even 8-inch disks. A double-sided, double density version could remember close to a megabyte!

Because the micro-floppies spin at 600 r.p.m. – twice the speed of 5-1/4 inch drives – and have much less area for the read/write head to cover, access time is better than existing drives, too. Media cost is the same as 5-1/4 inch disks, and the micro-floppies are encased in more foolproof hard plastic cases. They sound like ideal companions for the new, inexpensive home computers. So when will we see them for sale? Nobody knows. Sorry. But Clive Sinclair, the brains behind the price-breaking ZX-80/81 computers, already has come up with his own answer – again. Brochures advertising the ZX Spectrum in Great Britain promise that "coming soon" is a ZX Microdrive, apparently Sinclair's own version of the micro-floppy. The Microdrive, if it lives up to the ads, will be revolutionary: palm-sized, it will store 100K per disk and transfer data at 16K bytes per second! (See sidebar on the ZX Spectrum for more details.)

More Add-Ons

Speaking of Sinclair peripherals, a British firm called HSW & S, Inc. (with offices in Oxford, PA under the name Data-Assette) was at CES showing some unusual add-ons for the ZX-80/81. Two are keyboards: one is a nifty little affair that clamps neatly on the Sinclair's membrane keyboard to provide partial-stroke keys, and the other is a full-size, full-stroke keyboard in a separate case that plugs into the computer. You can buy either for $75. There's also the ZX 99, a plug-in module that adds an RS-232C interface so the Sinclair can drive any serial printer using standard ASCII character codes (the exhibitors at CES were running a Radio Shack Line Printer VII). The module also allows software control of up to four tape drives on the Sinclair, and includes a 2K ROM tape operating system. This sells for $150. A plug-in 64K RAM board also will be available.

Commodore VICMODEM

Commodore, besides its storm of new computers, also was showing off its VICMODEM. This is a very un-modem-looking modem, a cartridge that plugs into the VIC-20 and connects it directly to modular telephones (without the familiar acoustic coupler cups). This allows the VIC, among other things, to communicate with distant computers – yes, even mainframes – and to access computing services such as CompuServe, The Source, General Videotex, and the Dow Jones News/Retrieval Service. In fact, purchase of the VICMODEM includes free membership with CompuServe and free sample access time to all these services, including the Commodore Information Network, part of CompuServe. The VICMODEM also comes with its own terminal software (necessary for running a modem), called VICTERM I. Best of all, the whole package will sell for $109.95.

Atari also introduced a telecommunications package, the Communicator II. This includes a new direct-connect modem, called the Atari 835, the terminal software on a cartridge, dubbed *Telelink II*, and a free hour of sample access time on CompuServe, The Source and the Dow Jones Service. The list price is $279.95, and Atari predicts availability in the last quarter of 1982. *Telelink II*, which allows users to store and automatically dial two frequently called information service numbers and access codes, will be sold separately for $79.95, though the direct-connect modem will not be sold separately. The Atari 830 acoustic modem still will be available.

Sinclair ZX Spectrum

Like its predecessors, the ZX-80/81 microcomputers, the new ZX Spectrum appears to be yet another price breakthrough for Sinclair Research and founder Clive Sinclair. It offers features and memory capacity heretofore unavailable in machines costing up to $1,000 – yet it is selling for the equivalent of only about $200!

Available only in Britain for the past few months, and not due for export until the end of the year, the Spectrum reportedly has been selling like hotcakes. And no wonder. The basic Spectrum offers 16K RAM, a powerful 16K BASIC language in ROM, eight colors each for the foreground, background and border (with flashing and intensity control), a tone generator programmable from BASIC with variable pitch and duration, a partial-stroke keyboard (unlike the ZX-80/81 flat membrane keyboard) with auto-repeat on all keys and one-touch BASIC keyword entry, upper and lower case, full ASCII character set, high-resolution 256 by 192 dot-addressable graphics, instant syntax checking, and high-speed LOAD and SAVE with cassette (16K in 100 seconds).

For the equivalent of less than $300, the Spectrum is available with 48K RAM. Those who buy the 16K model can upgrade to 48K for only slightly more than a 48K Spectrum would cost outright.

Even more amazing is the line of peripherals for the Spectrum. In Britain, Sinclair is now selling the ZX Printer, a thermal paper dot-matrix device, for around $100. It has a full ASCII character set (including lower case), prints 32 columns wide at 50 characters per second, connects to the Spectrum with no additional interface, and automatically prints out any screen – including graphics – with the single command COPY. Then there's the RS-232C interface soon to be available which will allow the Spectrum to hook up to a wide range of printers, terminals, modems and other devices – for under $50. And finally, Sinclair is promising a ZX Microdrive: a palm-sized disk drive that stores 100K per disk and transfers data at 16K per second. The Spectrum can handle up to eight Microdrives, totaling close to a megabyte of storage. Sinclair says it will sell for around $100.

This means that for the first time, consumers will be able to assemble a full-blown computer system – with 48K computer, hi-res color graphics and sound, printer and disk drive – for around $500!

The secret, claims Sinclair, is a new master chip combining the functions of many chips in other computers. The Spectrum has only 14 chips. At its heart is a Z-80A microprocessor running at 3.5 MHz.

This master chip, by the way, partly accounts for the delay in exporting the Spectrum to the U.S., says a Sinclair representative. The chip is being redesigned to meet Federal Communications Commission standards, probably for radio frequency interference.

Like the ZX-80/81 (which it will not replace, incidentally), the Spectrum connects to any ordinary TV and cassette recorder. It has a 32-column by 24-row screen display and redefinable character set. The keys, although similar in appearance to the calculator-style keys on the TRS-80 Color Computer, are made of a soft rubber that feels spongy to the touch. Most of the 40 keys have at least four functions, and some have six.

The Spectrum's sound consists of a single tone generator controlled by the keyword BEEP – which is an accurate description. Don't expect the kind of sound you hear from Ataris or the new Commodores, but even a beep is better than silence.

The Spectrum's extensive BASIC is an enhanced version of the current Sinclair BASIC and corrects many deficiencies. Besides graphics commands such as BASIC INK, PAPER, BRIGHT, FLASH, INVERSE, BORDER and CIRCLE, there are also a READ, DATA and RESTORE, unlike the ZX-80/81 dialect.

If the master chip is redesigned on schedule, we might see ZX Spectrums in America by the end of 1982. Depending on Timex's sales of ZX-81s, the watch company may be marketing the Spectrum also (see text). All we can do is wait.

EXPERIENCE THE MAGAZINES OF THE FUTURE . . .

MAGATARI™
for your Atari 400/800

TRC
for your TRS-80 COLOR, MODEL I, MODEL III

appletree™
for your APPLE II +

Our magnetic magazines will entertain, educate, and challenge you. On cassette or diskette, every issue is packed with quality games, home applications, utilities, and our unique 'teaching' programs. But don't listen to us, let our subscribers convince you:

"Received my first copy . . . it's great! Please rush to me one of each back issue, so I'll have a complete set." **R.G., Chicago, IL**

"(Your magazine) is everything I expected and more!" **P.L., Durand, MI**

"You folks sure know how to put together real professional programs." **W.P., Meridian, MI**

"(Your magazine) is the best thing I've seen since the disk drive!" **K.R., N.Y., N.Y.**

Subscription price for cassette is $50 per year, $30 per ½ year, and $10 for a trial issue. For diskette $75 per year, $45 per ½ year and $15 for a trial issue . . .Don't miss another issue! **Order direct or see your local dealer.**

THE COLOR ACCOUNTANT

Over 1,000 satisfied customers are using this completely integrated package to analyze their personal finances.

1. **Chart of Accounts Maintenance**
2. **Complete Checkbook Maintenance**
3. **Tax Deduction Flags**
4. **Net Worth Statement**
5. **Income Less Expense Statement**
6. **Complete Graph Package** (graphs any files)
7. **Payments and Appointments Calendar**
8. **Home Budget Analysis**
9. **Mailing List**
10. **Schedule 1040 (Long Form)**
11. **Decision Maker**
12. **Schedule G (Income Averaging)**

All programs are menu-driven and allow add/change/delete. All files and statements can be listed to screen or printer, and saved to cassette or diskette. **THE COLOR ACCOUNTANT** is $75 for cassette and $80 for diskette and comes with 35 pages of documentation. This package requires 16K except for the ATARI 400/800 which requires 24K for cassette and 32K for diskette.

the programmer's program

Do you know all the innovative ways of using BASIC commands? The staff at the Programmer's Institute have designed a set of programs explaining everything the manuals omit.

The Programmer's Program leads you step by step through the fundamentals of programming your computer. Then you explore the finer points...multiple loops, queues, stacks, game programming, sorts, trees, boolean logic, and memory savers. All these methods make your programming tasks easier and more efficient. Difficult (and very useful) programs are created before your eyes, and designed so you learn at your own speed! **The Programmer's Program is a must for every owner of a microcomputer. ($50 cassette, $55 diskette).**

Order both The Programmer's Program and 1 year's subscription, and the total cost is only $75 cassette, $100 diskette. If you order now, you will receive a Football Program and Computerized Telephone Directory ABSOLUTELY FREE with any $75 order. So don't wait . . . order today!

THE PROGRAMMER'S INSTITUTE

a division of FUTUREHOUSE™

P.O. BOX 3191, DEPT. C
CHAPEL HILL, NC 27514
MasterCard/VISA Welcome
9 AM - 9 PM, Mon - Sat

ALL SOFTWARE REQUIRES 16K

1-919-489-2198

SUNRISE SOFTWARE

LEADER IN DISCOUNT SOFTWARE — 18 - 30% OFF

ATARI

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

PET / VIC-20

UNITED MICROWARE

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

Others

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

★ ★ ★ ★ ★ ★ ★ ★ ★ ★

BONUS

★ ★ ★ ★ ★ ★ ★ ★ ★ ★

PAC MAN

FOR ONLY $30.00

with

ANY ORDER OVER $150!

★ ★ ★ ★ ★ ★ ★ ★ ★ ★

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

24 HOURS A DAY
7 DAYS A WEEK

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

NO HASSLE! GUARANTEED REPLACEMENT POLICY

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

VISA

Epson HX-20

It was a pleasant surprise – dropping by the Epson booth at NCC in Houston to look for new printers and finding a startling new computer instead. What visitors found was the recently announced HX-20, the first computer from a company known for its economical and dependable printers. And it is definitely not a "me-too" computer. Epson's new machine is destined to redefine the term "portable."

The first thing that attracts people is the HX-20's looks: it is extremely compact, yet very complete. Less than a foot wide by 8-1/2 inches deep and 1-3/4 inches thick, and weighing less than four pounds, the HX-20 manages to pack in a 24-column dot-matrix printer with upper and lower case, a 20-column by 4-row liquid crystal display screen, a microcassette drive (optional), and a full-size, full-stroke keyboard. It's not a sparse keyboard, either. Its 68 keys include such extras as upper and lower case, five programmable function keys, BREAK, one-touch MENU, PAUSE, INS/DEL, HOME/CLR, NUM, CAPS LOCK, GRPH, screen scroll, and four-way cursor movement.

Although not much larger than a TRS-80 or Sharp Pocket Computer, its full-feature design – including 16K RAM expandable to 32K RAM – puts it in a league with much larger computers tied down to wall outlets. But the most amazing thing of all about the HX-20 is its portability: powered by four nickel-cadmium batteries, the HX-20 can run up to 50 hours on a charge, and be recharged in eight hours. Naturally, this performance depends on how often the printer and microcassette drive are used – such mechanical devices really eat up power.

The HX-20 also has a built-in RS-232C interface, and Epson was demonstrating a nifty little battery-powered modem that just happens to plug into the interface and make the perfect traveling companion for the machine.

And oh, yes, the latest word from Epson is that the HX-20 should be available by September for $795.

At the center of the computer are two Central Processing Units, both eight-bit 6301 microprocessors running at 614 KHz in a master-slave configuration. It comes with 32K of ROM expandable to 40K internally, and 64K with an expansion unit that was not shown. The liquid crystal display supports upper and lower case and a full-screen editor. Although only 20 characters wide, it scrolls another 20 characters sideways before wrapping down to the next line.

More HX-20 features: Microsoft BASIC is standard, there's a built-in clock and calendar with an alarm and interval timer, a programmable tone generator covering four octaves, interfaces for a bar code reader and a standard cassette recorder (the computer is available without the on-board microcassette), an internal DIP switch for selecting international character sets, 32 special graphic characters, a numeric keypad as part of the regular keyboard, and optional programs on ROM cartridges.

Other nice touches include a low-voltage power system that maintains data in RAM even when the computer is turned off, a knob that adjusts the liquid crystal display for straight-on or angled viewing, and even dot-addressable graphics for drawing charts on the small screen and printer. Epson is promising a floppy disk drive, too.

All in all, Epson has done a stunning job in packing so many features into a box about the size and weight of a hardback book. The keyboard – where skimping usually is done on small computers – is as luxurious as those on much larger, fixed-base machines. Although the HX-20 probably will not find its niche as a home computer, mainly because its display format limits game use, it seems to be ideal for traveling businessmen or engineers who need a portable, yet powerful computer that can hook up to any telephone for communicating with the home office. Its main drawback is the small display. It's a shame that a keyboard which lends itself so nicely to word processing is tied to a screen limited to only a few words at a time. An Epson representative says it may be possible to attach a monitor or TV, but there is no mention of this in the specifications.

At any rate, Epson's debut in computers is a groundbreaking – in more ways than one.

Software...

Although the most exciting news at the CES and NCC shows was the hardware, all was not quiet on the software front. Two major creative forces in the entertainment industry were married when Atari announced a partnership with Lucasfilm Ltd. to develop new video games for arcade coin-ops, home game machines and computers. Lucasfilm produced *Star Wars*, *The Empire Strikes Back*, and *Raiders of the Lost Ark*, and did the special effects for this summer's *Star Trek II*, *Poltergeist* and *E.T., the Extra-Terrestrial*. As anyone who has seen any of those movies knows, Lucasfilm obviously is a coven of creative talent. But aside from its sparse statement on developing new games, Atari was deliberately vague about what other paths might be explored by the joint venture. "We'll be developing new forms of electronic entertainment," said an Atari spokesman, "and the term 'electronic entertainment' is quite carefully chosen." In other words, just wait and see what we come up with, folks. As far as games, Atari did suggest that video game

Commodore P, B, BX Series

With so many new computers to see at the recent trade shows, including five from Commodore alone, it was almost easy to lose a few in the shuffle. Luckily, nearly every machine was a significant advance in either technology or pricing – and often both. This was particularly true of the new Commodores.

Although the Commodore 64, an amazing value at $595, stole lots of thunder, the new P, B, and BX Series Commodores drew their fair share of interest from computer hobbyists and business users. They, too, offer unprecedented features for the money.

With their sleek, white Porsche-designed cases, these three new series look very much alike, but there are important differences separating the P from the more expensive B and BX machines.

The P128 is the home computer of the trio. It is the third-generation PET and shares many features with the Commodore 64: 40-column by 25-row screen display on normal TV sets, 16 colors, high-resolution 320- by 200-pixel graphics, the industry's most advanced synthesizer chip for programmable sound effects and music, redefinable character set, and animation with independently movable graphics blocks called "sprites" (also known as player/missiles in Atari parlance).

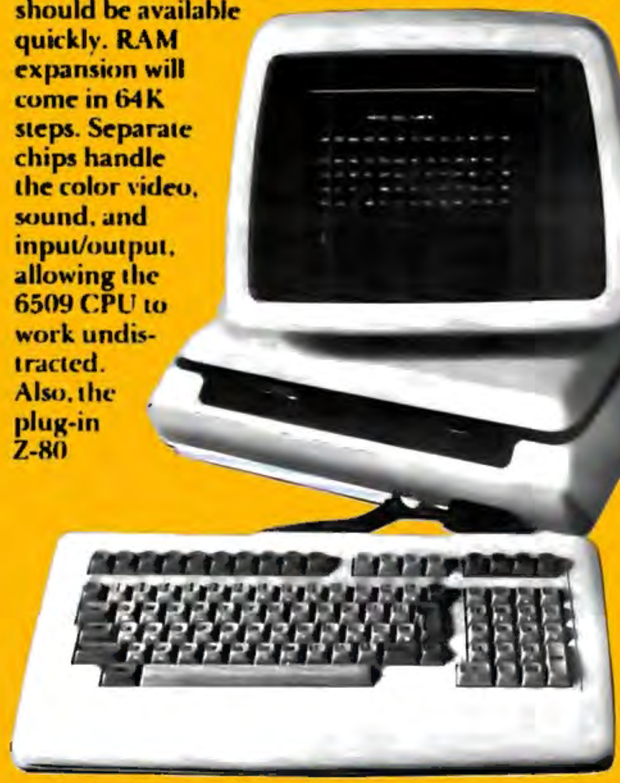
But that's where the similarities between the 64 and P128 end. The P128 comes with 128K of RAM – more than twice as much as most home computers can hold – and is expandable internally to 256K RAM and externally to 640K (896K total). And it will sell for just $995!

On top of that, there will be a plug-in Z-80 microprocessor board that will add CP/M, opening up a huge world of professional software, and even an 8088 16-bit microprocessor board for CP/M-86 capability. There's a built-in RS-232C interface for hooking up modems and printers, Commodore's traditional IEEE-488 interface to support the full range of Commodore CBM peripherals, a real-time clock, an enhanced Microsoft BASIC upward compatible with BASIC 4.0, audio system output, monitor output, and a cartridge slot for plug-in games and other software.

High-level languages such as U.C.S.D. Pascal will be supported, and the CP/M makes possible languages such as FORTRAN, COBOL and APL.

Commodore also pulled out all the stops on the P128's keyboard. It's the most complete we've ever seen on a home computer. Its 94 keys include 10 programmable function keys, a separate numeric keypad with CE, 00, ENTER and math operators, a key for each cursor arrow to support the full-screen editor, a large and easy to find RETURN, Pi, ESC, CTRL, INS/DEL, RUN/STOP, NORM/GRPH, CLR/HOME, OFF/RVS, and the full CBM business character set with PET graphics symbols.

Internally, the P128 has an eight-bit 6509 microprocessor for its Central Processing Unit (CPU). Commodore says this new chip is "functionally identical" to the 6502 found in previous Commodores, using the same instruction set. This means machine language programmers will adjust easily to the new chip, so new software for the computer should be available quickly. RAM expansion will come in 64K steps. Separate chips handle the color video, sound, and input/output, allowing the 6509 CPU to work undistracted. Also, the plug-in Z-80 and 8088 chips will work "concurrently" with the 6509 for what amounts to multiprocessing.



Game players weren't forgotten, either. The P128 supports two joysticks or four paddles. As on the Commodore 64, the sprites are totally independent of the background graphics, include collision detection and foreground/background priority, and can be three colors each. Eight of these can be moved anywhere on the screen at once. There's also a medium-resolution 200- by 160-pixel graphics mode, in addition to the hi-res and text modes.

The sound chip is the same Sound Interface Device (SID) found in the Commodore 64 and Max Machine. SID has three voices, programmable waveforms, filters, and 16-bit resolution over a

nine-octave range. (For a fuller explanation, see Part I of "First Look at the Commodore 64" elsewhere in this issue.)

In practically every category, then, the P128 is the machine that will set the pace for state of the art home computers – including price.

The B and BX Series computers are aimed more toward the business/professional users. Both are 80-column machines with built-in green phosphor monitors, dual 5-1/4 inch disk drives and detachable keyboards. The layouts of the keyboards, incidentally, are identical to the P128's.

The B128, like the P128, comes with 128K RAM and is expandable to 256K internally and 640K externally (896K total). It starts at $1,695 (including both disk drives). It also shares the P128's 6509 CPU, the SID chip, the optional Z-80 and 16-bit 8088 multiprocessor boards for CP/M and CP/M-86 capability, the RS-232C and IEEE-488 interfaces compatible with CBM peripherals, audio output, the cartridge slot, standard Microsoft BASIC upward compatible with BASIC 4.0, and the real-time clock. It does not have hi-res color graphics or joystick ports.

The BX256 computer offers all the B128 features above, except it comes with 256K RAM and the 16-bit 8088 microprocessor for CP/M-86. It will retail for $2,995.

The B and BX Series computers, then, with their built-in dual disks, tilt-and-swivel monitors, 80-column by 25-line displays, detachable keyboards, and higher prices, will compete in a different market than the P Series. The B and BX will be stiff competition in the low-end business/professional market against machines such as the IBM Personal Computer, the Apple II and III, the TRS-80 Models II and III and the new 16-bit TRS-80 Model 16. Meanwhile, the P128 will be equally strong in the high-end home computer market against the Apple II, the Atari 800, and the TRS-80 Model III.

If the P, B and BX Series Commodores hit the dealers by this fall as promised, expect to see topsy-turvy changes as the other manufacturers scramble to stay competitive. It should be quite a show.

versions of Lucasfilm movies probably could be expected. *Raiders* was mentioned specifically. But Atari also cautioned that nothing would come to fruition until at least next summer. That should leave plenty of time to fuel wild rumors.

Atari also announced three new pieces of educational software, price cuts of about 22 percent for *Asteroids*, *Computer Chess*, *Missile Command*, *Space Invaders*, and *Super Breakout*, and cuts of 33 percent for *Star Raiders* and *Music Composer*.

TI Unleashes Software

Texas Instruments announced a flood of new software for its TI-99/4A Home Computer – no less than 45 titles, including 30 cartridges. Many of these are interactive learning tools, all with color graphics and optional synthesized speech. Of particular interest to computerists is an Editor/Assembler package so hackers can get at the 16-bit TMS 9900 microprocessor that is the TI-99/4A's heart. A memory expansion unit and disk drive are required, and the assembler lists for $99.95. For the same price, TI also introduced the Mini Memory Module, a plug-in 14K RAM cartridge with a built-in battery that retains the memory even when the computer is shut down and the module removed. A full-featured word processor, *TI-Writer*, also was announced for $99.95. And for $129.95, you can get TI-LOGO II, an extended version of the respected educational language that adds music, sound effects and printer interface to the original TI-LOGO. Expect all these products by the fourth quarter of this year.

Commodore introduced a line of software to accompany its new machines, including games and versions of popular business programs. An educational program for the Commodore 64, *Visible Solar System*, used fascinating high-resolution color graphics to show the relationships between planets, their orbits and the sun. The business programs are *EasyCalc*, *EasyPlot*, *EasyTools* and *EasyScan*. *EasyCalc* is advertised as the largest spreadsheet-type program available on a micro, with a matrix of 65 columns by 999 rows. Versions for the SuperPET at $149.95 and the Commodore 64 at $99.95 will be available this summer, with conversions for the P, B and BX series to follow later. The Commodore booths also were crowded with people playing some of the 13 new cartridge games for the VIC-20, such as *Gorf*, *Omega Race*, *Wizard of Wor* and *Mole Attack*. Five Scott Adams adventure games also were shown for the VIC.

What's Next?

There you have it ... the highlights of both shows. But the real interesting part is yet to come. It will be intriguing to watch how the various computer manufacturers juggle their line-ups and prices to compete in the rapidly changing home market, especially for the under-$500 machines. This is where the manufacturers are trying to reach the vast consumer population, those who are unfamiliar with computers and are buying them as they would any other home appliance. Expect rapid changes there for the next couple years. We'll keep you posted.

A Monthly Column

The Beginner's Page

Structured Programming

Richard Mansfield
Assistant Editor

From time to time you'll hear about *structured* programming. It's not a single idea or technique, but rather a cluster of suggestions about the best way to go about programming. A "perfect" programming method will probably never be developed, but the various suggested "rules" associated with "structured programming" are worth looking into to see if there's anything there that you might find helpful.

Never Use GOTO

Advocates of structured programming suggest a variety of guidelines which programmers should follow to achieve easily written, easily modified, easily understood, and efficient programs. Some structured programming enthusiasts say you should never use GOTO in a program. GOTO jumps out of the normal "flow" of execution; it makes things unclear because its purpose within a program cannot be quickly understood. Other structured programming theorists suggest that a program should be "flowcharted" before the actual programming starts. A flowchart is to a program what an outline is to an essay: it illustrates the main ideas and shows the path of execution which the computer will follow when the program RUNs.

Related to flowcharting is a third idea associated with structured programming called "top-down" programming. In brief, this means that you make a general outline or flowchart of the main parts of your program before getting down to writing its individual subroutines and specific parts. This could mean that you first program a "main loop" which is a series of GOSUBs to subroutines written later. (With the opposite approach, "bottom-up" programming, you write the subroutines first and then tie them together at the end with a main, governing, "top" routine.)

"Modular" programming (breaking a program into smaller parts and solving each separately, using many subroutines); organizing DATA lines or files so that they are clearer and structured in a way that reflects the structure of the program which uses them; indenting program lines so that you can *see* a loop in a LISTing – all of these can be found under the umbrella idea: structured programming.

```
10 FOR I = 1 TO 10
20 PRINT I
30 PRINT I * I
40 FOR J = 1 TO 500
50 NEXT J
60 NEXT I
```

An "unstructured" listing.

```
10 FOR I = 1 TO 10
20     PRINT I: REM PRINT NUMBER
30     PRINT I * I: REM NUMBER SQUARED
40         FOR J = 1 TO 500: REM DELAY LOOP
50         NEXT J
60 NEXT I
```

One aspect of structured programming is indenting loops so that they are easily seen.

How you program is up to you. There are two general approaches, and each has its passionate advocates:

1. Improvise as you go.
2. Plan everything first.

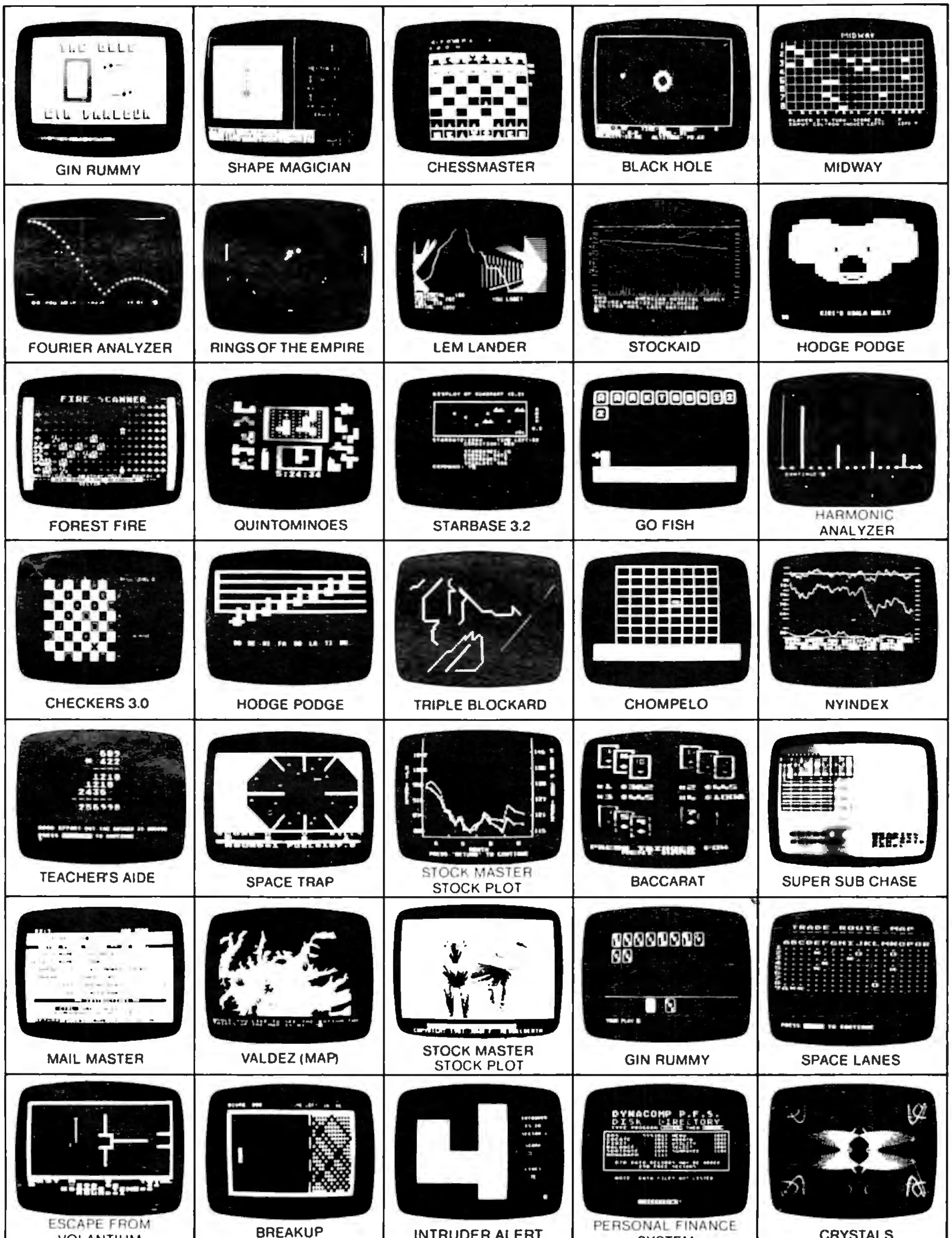
The recreational programmer tends toward the first style. Type it in, RUN it, "how did *that* happen?" try something else, fix up problems, RUN again, and eventually it works.

The Special Requirements Of Group Programming

Professional programmers work under entirely different conditions. They must communicate with the computer, of course, but they also must communicate with each other. Their programs must meet standards. *Documentation* (explanations of what's going on in a program) must be thorough and clear. Often they work in groups, as part of a programming team. A job is broken down into pieces so that each programmer is writing a subroutine of what will later be linked together to become the main program. Unless things are well coordinated, group programming will be unsuccessful. It would be like an auto factory where each worker made a personal decision about where to drill bolt holes on his piece of a car. When the time came to put it all together, very little would fit.

For somewhat similar reasons, many teachers would favor structured programming. If you had

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



TURN TO NEXT PAGE....

to grade twenty programming solutions, your task would be easier if all the students' programs were following general rules and were easy to understand. If a student program is extensively commented with REM statements, if all the subroutines are clearly identified, if the "flow" (the path of execution) of the program is straightforward and obvious – the program can be more easily analyzed and evaluated.

Also, students who are first learning to program might find that the additional, formal rules of structured programming aid them in grasping the elements of programming. On the other hand, other students might find the same rules cumbersome, an unnecessary burden. Because programming is both a science and an art – it could be taught either way. Some students might profit from comprehensive "laws" which must be followed; others might need the creative freedom of the artist with "rules" kept to a minimum.

It's Your Decision

Clearly, this distinction between the improvisational and pre-planned programming styles is somewhat artificial. All programming involves *some* rules which must be followed if the program is to work correctly. The heart of this issue is deciding how many rules, how much formality is useful. Certainly, those special situations involving group programming, such as the classroom or professional programming teams, are operating under special constraints which require special rules.

For the rest of us, the benefits, if any, of the various structured programming notions must be determined on a personal basis. If indenting your loops helps you in some way, by all means do it. If you need heavy REMarking, REM liberally. If you think using GOTO is perilous, avoid it. Your personal style will evolve, and you will naturally use what you find useful and discard what you find unnecessary.

But be wary of the idea that structured programming (or any other "solution" to programming tasks) is a cure-all. As Raeto West points out in his book, *Programming the PET/CBM*, "...the sad fact is that any complex program will remain complex in whatever way it is written down." ©

This personal budget program is designed for the Atari with 16K and a cassette recorder or Commodore PET. To adapt the program to Apple or VIC, see the accompanying notes.

Household Budget Manager

Richard Kalagher
Vienna, VA

This program will allow you to enter and store all of your household check and credit card expenditures. You can allocate these expenditures to budget categories that you define, review selected portions of your expenditures, perform searches, and calculate sales tax. And you don't need a disk drive to save the data! All data automatically becomes part of your program, which you save on cassette at the end of each session.

When I first tried to write a cassette based data storage system, I found that storing and reading data from the tape was not only somewhat unreliable, but painfully slow. However, Bruce Frumker's article in **COMPUTE!**, August, 1981, #15, solved the problem. By using the "dynamic keyboard" technique, this program stores any data you enter as part of the program. You just need to SAVE the program on cassette after each use. And you will find that the program will run faster than a disk based system.

In order to save memory for data, there are not many REM statements in the program. Lines 8-113 initialize the program variables and fill the array P with the values in the DATA statements on lines 200-230. (You could put thirty zeros separated by commas on line 200 to save typing, the first time you run the program.)

Array P contains all of the saved balances. P(0) is the line number (minus 10000) where the next set of data will be written. P(1) through P(26) contains the balances in your budget categories. The index corresponds to the letter of the alphabet. For example, if you have a category called "Insurance," its balance is contained in P(9) since I is the 9th letter of the alphabet. You could, for example, increase the balance by $100.00 (if you made a mistake) by hitting BREAK after the main menu is displayed, typing in P(9) = P(9) + 100.00, hitting RETURN, typing CONT, and hitting RETURN again to start the program running. P(27) is your checkbook balance, P(28) the last month, P(29) the last day, and P(30) the Unallocated Deposits (explained later). Line 232 contains the current year which you should change each year.

Lines 999-1999 contain the main menu. Line 1155 sends the program to a given subroutine. You should be able to determine how each subroutine works by studying the code and the descriptions below of what each option is used for. Just remember that most decisions are based on the ATASCII code of the first character of a given string.

The budget system is simple and straightforward. First you decide on the budget categories you would like (e.g., Household, Automobiles, Savings, etc.) When you write a check, the computer will subtract the amount from the budget category and from your checkbook balance. When you make a purchase with a credit card, the computer will again subtract the amount from the budget category, but it will add the same amount to a credit card escrow category. When you pay your credit card bill, the amount will be subtracted from the credit card escrow category. In addition, you can search for and display the information in many different ways.

Setting Up Your Budget Categories

You must select your budget categories and the names of your credit card escrow accounts. There are two restrictions:

1. You can have a total of 26 categories.
2. The names of each category must start with a different uppercase letter.

For example, you cannot have "Mastercard" and "Medical" but you could have "Doctors and Dentists" and "Mastercard." I recommend that you use a small number of budget categories (i.e., 8-15). For example, I include my mortgage payment, utilities, and major house repairs in one category called "Utilities" – but the choice is yours.

Similarly, you can have a separate escrow account for each credit card or you could lump them all into one category called "Credit Card Escrow." Again, the choice is yours.

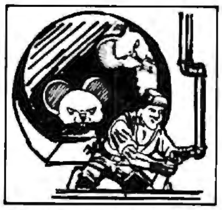
The DATA statements in lines 9000-9500 contain the names of the budget categories. You can use these or make up your own using the above restrictions (remember to begin with an uppercase letter). The order of the categories is not important (unless you want them in a particular order). Be sure the last statement is

```
9500 DATA END
```

To Order Call Toll-Free
800-424-2738

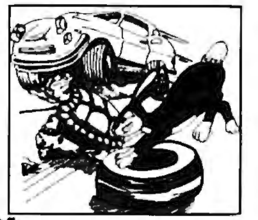
ATARI®

the PROGRAM STORE™



MOUSKATTACK

From On-Line by John Harris
You are Larry Bain, ace plumber, trying desperately to finish installing a pipe through the dangerous corridors of Rat Alley, before the legendary man-eating rodents can stop you! The last plumber that entered disappeared without a trace, so be careful!
48K Disk, ~~$34.95~~ *Save 20%!*
Now Thru Sep. 1 You Pay Just $27.96



PACIFIC COAST HIGHWAY

From DataSoft by Ron Rosen
Fast-action game with good graphics. You must try to cross the Highway without getting hit by the rush hour traffic. Once across the highway, you must leap from boats to rafts to cross the Pacific Ocean. One added challenge: you're a *tortoise*!
16K Tape or Disk, ~~$29.95~~ *Save 20%!*
Now Thru Sep. 1 You Pay Just $23.96

CANYON CLIMBER

From DataSoft by Tim Ferris
Great outdoor adventure! You're at the bottom of the Grand Canyon, trying to scale the world's toughest wall. If that's not enough of a problem, there are three challenges to face along the way: angry sheep, threatening Indians and attacking birds. Action-packed game you'll really enjoy.
16K Tape or Disk, ~~$29.95~~ *Save 20%!*
Now Thru Sep. 1 You Pay Just $23.96

SLIME

From Synapse by Mike Hales
Large drops of "Slime" fall from the sky onto your ships. If *one drop* hits your ship, it will sink. Use the triangular diverters to aim Slime into buckets on the sides of the screen. If Slime falls into the ocean, the level of the ocean rises. When the ocean reaches the top of the screen, the game is over, and your score is calculated. One-player game with excellent graphics.
16K Tape or Disk, $29.95

WARLOCK'S REVENGE

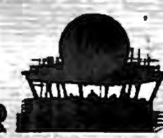
From Synergistic Software by Butch Greathouse
You're leading a group of adventurers into the evil warlock's underground empire to recover the treasures he has stolen. Your party includes a cleric, a thief, a gladiator, a strongman, a magician, a wizard and an elf. You must select the best character to lead each part of the adventure, as each has special powers and abilities you may need to gain your objective. Good luck, and watch out for the Torkies!
40K Disk, $34.95

NAUTILUS

From Synapse by Mike Potter
Two-player interaction combat game: the underwater cities and their destroyers against the helicopters and submarines. The helicopters bomb the ships; the subs fire torpedoes at the cities and surface to fire at the destroyers. The destroyers fire at the helicopters and drop depth charges on the subs. They can also lower men into the ocean to rebuild cities hit by torpedoes. Top and bottom halves of screen scroll separately for maximum field of play. Great graphics.
16K Tape or Disk, $29.95

ULYSSES AND THE GOLDEN FLEECE

From On-Line by Bob Davis & Ken Williams
The setting is ancient Greece, and you are Ulysses. The king has assigned you the task of retrieving the Golden Fleece, kept from men for many years under the protection of the gods. On your journey, you will encounter the song of the Sirens; one-eyed Cyclops; and the famous winged horse, Pegasus. Ulysses is known as the most daring and skilled seaman of all time . . . it's up to you to prove it!
40K Disk, $39.95



CONTROLLER

From Microcomputer Games (Avalon Hill)
You are the air traffic controller who must make quick decisions . . . or you might cause a mid-air crash and lose hundreds of lives! All actions are in REAL TIME—aircraft will maneuver, use fuel and respond to your commands as a real plane would—the computer won't wait for you. Aircraft are plotted on realistic "radar scope." Heading, velocity and altitude displayed on separate chart. Select up to 8 planes; choose from 3 types with 3 levels of difficulty.
40K Disk, $29.95

DOMINOES AND CRIBBAGE

From Creative Computing Software
Challenge your computer to a game of draw Dominoes. You can repeat or alternate draws, but you have a pretty tough opponent who knows all the rules! Or, can you beat the computer in Cribbage, and be the first to peg twice around the board? An exciting challenge with graphic display of board and cards.
16K Tape or Disk, $19.95

CROSSFIRE

The aliens have taken over the city and everyone left peacibly—except you. You are surrounded by the laser firing hordes. In this exciting arcade game your chances of survival are slim, but you can make it—if you don't get caught in the crossfire!
16K Tape or 48K Disk, $29.95

CAVERNS OF MARS

By Greg Christensen from APEX
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! Joystick required.
16K Tape, $29.95 16K Disk, $37.95

Visit our other stores: Seven Corners Center, Falls Church, VA • W. Bell Plaza, 6600 Security Blvd., Baltimore, MD
Coming Soon to Columbus, Ohio

VISA **For Information Call 202-363-9797** **To Order Call Toll-Free 800-424-2738** master charge

the PROGRAM STORE
4200 Wisconsin Avenue, NW, Dept. U208 Box 9609
Washington, D.C. 20016

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

the PROGRAM STORE

For Information Call 202-363-9797

GREAT SUMMER READING FROM THE PROGRAM STORE

ATARI SOUND AND GRAPHICS

By Moore, Lower and Albrecht from John Wiley & Sons

This self-paced, self-teaching guide will have you seeing and hearing things on your Atari in no time—even if you're a complete beginner. You'll learn to compose and play melodies, draw cartoons, create sound effects and games. Each section teaches something new in BASIC, the most commonly used computer language.

Softcover, $9.95

YOUR ATARI COMPUTER

By Poole, McNiff & Cook from Osborne/McGraw-Hill

Comprehensive, all-in-one guide for Atari 400/800 computers that is helpful to the beginning and advanced programmer. Complete operating instructions; detailed BASIC programming, including a handy alphabetical glossary of BASIC statements and functions; computer graphics; and tips on hardware, peripherals and compatible software.

Softcover, $16.95

COMPUTERS FOR PEOPLE

By Willis and Miller from Dilithium

Do you need a computer for your business? Your home? Which one would be most useful? Which software should you consider? You don't have to be an electronics wizard to answer these questions. . . just read this book! It will guide the complete novice, step by step, to the selection of the best system for your needs.

Softcover, $7.95

VISICALC HOME AND OFFICE COMPANION

By Castlewitz and Chisausky from Osborne/McGraw-Hill

Fifty VisiCalc models for home and office, including: investments, inventory, sales forecasts, payroll, personal net worth, home budget planning, family insurance needs. Each comes with model listings, sample printed reports and narrative.

Softcover, $15.99

COMPUTE!'S FIRST BOOK OF ATARI

From the Editors of Compute! Magazine

Over *thirty* articles selected from Compute! on the Atari, including: Atari BASIC, a whole section on graphics, programming hints, Atari DOS and lots more!

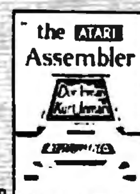
Spiral-bound, $12.95

6502 ASSEMBLY LANGUAGE SUBROUTINES

By Leventhal and Saville from Osborne/McGraw-Hill

If you're interested in using assembly language quickly, this book is ideal. It describes general 6502 programming methods, provides code for more than 40 subroutines to help you improve your programming skills, debug or revise an existing program; add instructions and addressing modes.

Softcover, $12.99



ATARI ASSEMBLER

By Don & Kurt Inman from Reston

If you're new to the 6502 processor, this book is a must. The authors guide you through the rudiments of this assembly language in clear, easy steps. Includes full listing and description of 6502 mnemonics and addressing modes.

Softcover, $12.95

INSIDE ATARI DOS

By Bill Wilkinson from Compute

The comprehensive manual on the disk File Manager System (FMS), commonly known as Atari DOS 2.0S. Contains the only complete and official listing for the system, plus a full description of: the external view, charts & tables, various interfaces and functions of individual subroutines.

Spiral-bound, $19.95

MASTER MEMORY MAP

From Santa Cruz

A most important tool for the serious programmer, this book is a highly detailed map of the Atari's memory—with thousands of locations and routines. Simplifies use of display list, player missile and interrupt graphics, plus notes on the new GTIA graphics chip. The "Miscellaneous Notes" are a find in themselves.

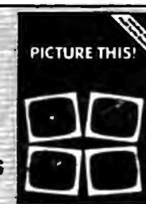
Softcover, $6.95

DE RE ATARI

From APEX

According to its title, this book is "All About Atari", and it lives up to its name! For the advanced programmer who wants to learn the many features that make the Atari 400/800 so versatile. Includes a discussion of the new GTIA graphics chip.

Looseleaf (binder not supplied), $19.95



PICTURE THIS! An Introduction to Computer Graphics for Kids of All Ages

By David D. Thornburg from Addison-Wesley

This book promises to become the "modern replacement for coloring books and crayons". It's a learn-by-doing manual that uses PILOT, a simpler language than BASIC, and Turtle Geometry to teach kids to create pictures in full color from simple lines to complex angles and curves.

Spiral-bound, $14.95

COMPUTERS FOR KIDS Atari Edition

By Sally Larsen from Creative Computing

The BASIC programming manual written for kids, from flowcharts to color graphics. Detailed instructions and sketches plus glossary of statements and commands. With lesson plans and tips for parents and teachers.

Softcover, $3.95

KATIE AND THE COMPUTER

By D'Ignazio; Illustrated by Gilliam from Creative Computing

Delightful, full-color picture book about Katie, who falls into the land of Cybernia, inside her father's home computer. She learns how a computer works during her adventures with Colonel Software, the Bytes and a ferocious Program Bug. Ages 4 to 10.

Hardcover, $6.95

STIMULATING SIMULATIONS, Atari Version, 2nd Edition

By C.W. Engel from Hayden

A handbook of 12 simulation games including Art Auction, Starship Alpha, Monster Chase and Devil's Dungeon—each complete with listing, sample run, instructions and program documentation, including flowchart and ideas for variations.

Softcover, $5.95

GAMES & RECREATION

From Reston

A new book to delight the computer hobbyist! It's filled with sample games and ideas for creating new ones, plus full description of the Atari keyboard, graphics and sound and color capabilities.

$14.95

THE PROGRAM STORE • Dept. U208 Box 9609 • 4200 Wisconsin Avenue, NW • Washington, D.C. 20016

Item	Tape/Disk/Book	Price	
			Postage $1.00
			Total ______

☐ CHECK ☐ VISA ☐ MASTERCARD

name ______

address ______

city ______ state ______ zip ______

Card # ______ Exp ______

Allocating Money

As you write checks and make charge purchases, the amounts will be subtracted from the budget categories. You could just let these amounts accumulate (they would show as negative balances in the budget categories) or you can add money each month (or week) to the categories. One procedure would be to decide how much money each month you want to allocate to each category. As you allocate money to categories (the procedure is described in the next section), the money is subtracted from a special category called "Unallocated Deposits."

When you make deposits, they can be used to cover the money in this special category. At the end of the month, any additional money in "Unallocated Deposits" could be allocated to a "Savings" or "Miscellaneous" category or could just be left for you to balance out at the end of the year.

Menus

The following sections describe the function of each of the nine main menus.

1. *Enter Checks and Charges*

This is the basic input for the program. The computer will prompt you with questions which you answer, and then press RETURN. First you are asked for a check number or charge card. If you are entering a check, you must input as the first character a number (0-9). You can enter up to four digits. If you have a billpayer account or automatic checks to which you do not assign a number, you can use any of the following symbols as the first character of the check number: !, #, $, %, *, ., /, ?, etc. (Any symbol with an ATASCII decimal code less than 64 will work except the + sign.)

For deposits, the first character of the check number must be the + sign. For example, you could use +, +PAY, +DEP, +12, etc. You can also enter interest by using the + sign, or service charges by entering a + sign and then a negative amount when the computer asks for the amount.

For credit card purchases, enter the name of the credit card. The first letter must correspond to the budget category you have set up for that credit card. For example, if your category is "Visa Escrow" then enter Visa or V for the entry. If you are using just one credit card escrow for all credit cards, the first letter of this entry should be the same as the name of the category. For example, if the category was named "Credit Card Escrow" you could enter C-V for Visa, C-M for Mastercard, etc.

You will now be asked for the date. Enter a two digit month, a backslash, and a two digit day (i.e., 03/21, 05/01, 12/25, etc.).

You will now be asked for the amount. Enter the number with two decimal places. For example 10.00 not 10 if the amount was $10.00. This will give you a nice right-justified column on the screen. For amounts over $10,000.00 the balances will be correct although you will get some truncation on the screen. All numbers are entered as positive although you can enter negative numbers if you want (i.e., a service charge entered as a deposit).

You will now be asked if you paid sales tax on the item (answer Y or N). This information can be used at the end of the year to determine how much sales tax you have paid during the year for income tax purposes.

You will be asked to enter a budget category. You need only enter the first letter of the budget category, but you can enter the whole word if you like. If the entry was a deposit (i.e. first character of the check number was a + sign) the computer will ask if you want to allocate the deposit to a budget category. If you say "no," the deposit will be allocated to the special "Unallocated Deposit" category (see previous section). If you say "yes," it will ask which category.

The computer will now ask if the data is okay. If you say "no," you will be asked to re-enter all of the data. If you say "yes," you will see a brief flash of letters on the screen. What happens is the program momentarily stops executing and adds a line of data (to itself) containing the information you have just entered.

You will now be asked for a check number or credit card again. You now repeat the above process. If you are through making entries, hit RETURN and you will return to the main menu.

2. *Print Check Register*

When you select this option you will get a listing on the screen of all checks and deposits for the period you select. If you just want to see one month, enter that month for both starting and ending month. If you want the whole year then enter 1, 12. The "BALANCE" will be your current checking account balance. "TOTAL FOR ABOVE" will be the sum of the checks less deposits for the time period you have selected. You will be prompted to hit RETURN when the screen is full or when you want to return to the main menu.

3. *Print Charge Register*

This works just like the check register except you will be asked which charge card. If you have only one charge escrow, enter the first letter of its name.

4. *Update Balances in Budget Categories*

This is the routine you use to allocate money to the budget categories. You need only enter the first letter of the budget category. Note that you can enter a negative amount. For example if you had $-35.67 of unallocated deposits at the end of

It's the **Software Automatic Mouth — S.A.M.**
The brand new, all-software, high quality speech synthesizer
from **DON'T ASK.**

S.A.M. gives you:
Unlimited vocabulary
Full inflection at your control
Effortless access from BASIC
Separate pitch and speed control
Thorough, instructive owner's manual
Easy-to-learn phoneme spelling system
And automatic English-to-speech conversion
Elaborate internal pronunciation rules for natural-sounding speech

S.A.M. for the APPLE II/II+ includes:
8 bit digital-to-analog converter and audio amplifier on a board (amplifies the sound of all your APPLE games).

- **S.A.M.** on disk
- Complete documentation
- English-phoneme dictionary

Only $124.95

APPLE is a trademark of APPLE COMPUTER, INC.

S.A.M for the ATARI 400/800 includes:

- **S.A.M.** on disk
- Complete documentation
- English-phoneme dictionary

Only $59.95

Note: On the ATARI, screen blanks during vocal output.

ATARI is a trademark of ATARI INC.

Hear S.A.M. at your favorite computer store today.
For more information, contact:

DON'T ASK
Computer Software
2265 Westwood Blvd., Ste. B-150
Los Angeles, California 90064
(213) 397-8811

- For a copy of the owner's manual, specify computer and send $2.00 to **DON'T ASK.**
- You can order **S.A.M.** directly from **DON'T ASK.** Add $2.00 for shipping and handling to your check or money order (or order C.O.D.).

Dealer inquiries welcome

Talk is cheap

the month, you could allocate this amount to "Savings" or "Miscellaneous" to balance the "Unallocated Deposits" to zero.

Hit RETURN to return to the main menu.

5. *Calculate Sales Tax*

If you use the sales tax tables to calculate this deduction on your income tax, you may be losing money. At the end of the year, this routine will tell you how much you have actually spent. Just enter your state's sales tax in percent (i.e., enter 4 not .04 for a 4% rate) and you will get the correct result.

6. *Print Balances in All Categories*

This routine will tell you the balance in all budget categories, your checkbook balance and the amount of Unallocated deposits.

7. *Perform Search for Charge or Check*

This is one of the most useful and most entertaining routines to use. Just enter from 1 to 16 characters and the computer will find and total all charges and checks that contain those characters in the Payee field. Want to know how much you spent on electricity? Just enter the name of your electric company. When was the last time you went to the dentist? Just enter the dentist's name and you will know. If there are items you might want to find and total, you can enter your own special characters in the Payee field at the time you enter data and then find and total them using this routine.

8. *Examine an Individual Category*

This routine works like menu items two and three above except you can look at any budget category.

9. *Store Data and End Session*

When you are finished for the evening, enter nine at the main menu to end this session. First the screen will blink for about ten seconds while the new balances become part of your program. You will then be instructed to turn over the tape, rewind, and hit RETURN with PLAY and RECORD pressed. Your program will be saved on tape (with all your data) for the next session. Note that you have the previous version of your program on the other side of the tape in case you have any difficulties loading the most current version next time.

Operational Concerns

If you make a mistake when entering data, the computer will ask you to re-enter the data after reminding you of the correct format. If the computer does stop, it will display a message stating what line it stopped at and give you an ATARI BASIC error code. You can almost always recover from an error by typing CONT and hitting RETURN. This way you will not lose any of the data you have entered during the session.

Since the data is being stored as part of your program, the program will require more memory as you use it. On a 16K system you have about 6500 bytes of available memory. Each entry takes 30-60 bytes depending on its length. To find out how much memory you have remaining you can type ?PRINT FRE(0) when your program is loaded, but not yet running. You will also be warned with a red screen and a message if you have less than 1000 bytes remaining when you run the program.

When you have little memory left you can store the data by itself on a separate tape. Use the following procedure:

1. Put a new tape in the Program Recorder, rewind, and press PLAY and RECORD (label the tape since you will want to file it).
2. Type LIST "C", 10000, 30000
3. Hit RETURN twice
4. After the tape stops you will have a tape of just your data.
5. Put in a scratch tape, rewind, and press PLAY and RECORD.
6. Type LIST "C", 0, 10000.
7. When you get the READY prompt, type NEW and hit RETURN.
8. Rewind the tape and press PLAY.
9. Type ENTER "C" and hit RETURN twice.
10. When you get the READY prompt you will now have your program in memory with all of the current balances, but without the check and credit card data. You will also again have about 6500 free memory bytes (with a 16K system) so you can continue your record keeping.

Any time you want to put your old data back, just follow the above procedure, then put in the data tape you want and use ENTER "C" to load the data as part of your program. How often you have to do this will depend on how many checks and charges you use, but you might want to choose a standard time such as yearly or quarterly.

I have deliberately left out several possible features in order to conserve as much memory as possible. You could, if you have a larger memory capacity, add routines to balance your checkbook, automatically format dollar amounts, search fields other than the payee field, use logical searches, print to a printer, etc.

If you would like a copy of this program, [*the Atari version only*] please send a blank disk or cassette and a stamped, self-addressed mailer, and $3.00, to me at 1841 Nigel Ct., Vienna, VA 22180.

Notes To VIC, Apple, And PET Owners

This program makes use of a very unusual feature of the Atari, "forced read." When activated, the Atari will start generating carriage returns automatically, entering whatever appears on the screen as if a user had entered it. It can be used for line deletion, auto-numbering when entering programs, or the update of a program line: in effect, the machine can program itself. In Mr. Kalagher's program, this technique is used to update a series of DATA statements used to hold budget information, and add other information such as checks and charges to the end of the program. This keeps the program and its data together, eliminating the need to use a cumbersome cassette file. When the changes are complete, the program is re-SAVEd onto cassette.

Obviously, converting this program to run on another computer would be no easy task. Few computers share "special features," such as this one. However, a similar technique commonly known as the "dynamic keyboard" can be used on the PET/CBM and VIC-20.

On these computers a "keyboard queue" is used to remember the last ten keystrokes typed. This provides a feature known as "typeahead," which lets you enter keystrokes faster than the computer can process them, preventing any loss of data.

Normally the operating system of the computer manages the keyboard queue, but it can be "fooled" into thinking up to ten characters need to be typed by POKEing the desired characters into the "keyboard buffer." The number of characters POKEd is stored in an "index." The most common use of this technique is to place up to ten carriage returns as if the user had typed them. This is different than PRINTing a carriage return, which merely moves the cursor to the left edge of the next line.

Any information on the screen that is "RETURNed over" in this fashion will be entered as data or program lines. One serious complication is that the PET/CBM and VIC clear out all variables and arrays when a new program line is entered. This makes it necessary to list all the essential variables on the screen, and then RETURN over them too. In this program, the entire P(0-30) array must be listed out in DATA statements any time a line is changed.

The subroutine at 9700 is then called to read the array back in and initialize program constants, since they were lost when the lines were updated. GOSUBs and FOR/NEXT loops are also cancelled, so you can't RETURN from such a routine or use it in a FOR/NEXT loop. The last line of the list should be a "GOTO xxx" statement that transfers control back to the program, usually the line following the line that calls the subroutine (to simulate a RETURN).

Screen positioning is important: see the subroutine at 9600. To convert it to run on Original ROM PETs, change POKE 158,9 to POKE 525,9 and POKE 622+I,13 TO POKE 527+I,13. For the VIC, change them to POKE 198,9 and POKE 631+I,13 respectively.

VIC owners will also want to tailor the menu and prompts to conform to the 23 character line length or even add some color! You will not have very much memory for your budget information with the 5K VIC, so you may want to use several copies of the program, maybe one for every three months.

Owners of other machines will have to research into their machine's "special features" and see if there is any way to implement the "dynamic keyboard." If you do, write it up and share it with the rest of us. Apple owners who have disk drives could SAVE the data on disk rather than in the program.

Program 1: Atari Version

```
1 REM HOUSEHOLD CHECK, CREDIT CARD,
2 REM AND BUDGET PROGRAM
8 SETCOLOR 2,2,6:SETCOLOR 1,2,0:SETCOLOR
 4,10,4
10 DIM P(30)
20 FOR I=0 TO 30
30 READ X:P(I)=X
40 NEXT I
45 PRINT CHR$(125)
50 IF FRE(0)<1000 THEN PRINT CHR$(125):S
ETCOLOR 2,4,4:? "WARNING!  ROOM FOR ABOU
T ";INT(FRE(0)/50);" MORE ENTRIES"
70 ? "PROGRAM WAS LAST RUN ON ";P(28);"/
";P(29)
75 TRAP 83
80 ? :? :? "ENTER TODAY'S DATE (MM,DD)"
81 INPUT X,B
82 GOTO 90
```

```
83 ? :? "TRY AGAIN. REMEMBER TO PUT A CO
MMA BETWEEN DAY AND MONTH"
84 GOTO 80
90 P(28)=X:P(29)=B
100 DIM S$(38):S$="
                  "
101 DIM N$(5):N$=S$
102 DIM D$(9):D$=S$
103 DIM I$(16):I$=S$
104 DIM A$(7):A$=S$
105 DIM W$(1):W$=S$
106 DIM C$(1):C$=","
107 DIM Z$(38):Z$=S$
108 DIM T$(1):T$=S$
110 DIM B$(10):B$="BALANCE = "
111 DIM SE$(16):SE$=S$
112 DIM Q$(1):Q$=S$
113 DIM Y$(3):READ Y$
200 DATA 0
201 DATA 0
202 DATA 0
203 DATA 0
204 DATA 0
205 DATA 0
206 DATA 0
207 DATA 0
208 DATA 0
209 DATA 0
210 DATA 0
211 DATA 0
212 DATA 0
213 DATA 0
214 DATA 0
215 DATA 0
216 DATA 0
217 DATA 0
218 DATA 0
219 DATA 0
220 DATA 0
221 DATA 0
222 DATA 0
223 DATA 0
224 DATA 0
225 DATA 0
226 DATA 0
227 DATA 0
228 DATA 0
229 DATA 0
230 DATA 0
231 REM CHANGE YEAR
232 DATA /82
999 PRINT CHR$(125)
1000 PRINT "WHAT DO YOU WANT TO DO?":? :
?
1010 PRINT "1. Enter Checks and Charges"
:?
1020 PRINT "2. Print Check Register":?
1030 PRINT "3. Print Charge Register":?

1040 PRINT "4. Update Balance in Budget
Categories"
1050 PRINT "5. Calculate Sales Tax":?
1060 PRINT "6. Print Balances in all Cat
egories":?
1070 PRINT "7. Perform Search for Charge
 or Check":?
1080 PRINT "8. Examine an Individual Cat
egory":?
1095 PRINT "9. Store Data and End Sessio
n"
1140 PRINT :PRINT :PRINT "ENTER NUMBER";

1145 TRAP 1998
1150 INPUT WHICH
1151 PRINT CHR$(125)
1155 ON WHICH GOSUB 3000,4000,4000,2500,
7000,2000,6000,4000,5000
1998 T$=" "
1999 GOTO 999
2000 REM PRINTS BALANCES IN ALL CATEGORI
ES
2020 ? "      BALANCES IN ALL CATEGORIES"
:?
2030 RESTORE 9000
2040 FOR I=1 TO 26
2050 READ Z$
2060 IF Z$="END" THEN GOTO 2100
2070 PRINT Z$;" = $ ";P(ASC(Z$)-64)
2080 NEXT I
2100 POP
2110 PRINT "Unallocated Deposits = $ ";P
(30)
2120 PRINT "Checking Account Balance = $
 ";P(27)
2450 ? :? "HIT |RETURN| WHEN THROUGH":IN
PUT T$
2460 RETURN
2460 RETURN
2500 REM SUBROUTINE TO UPDATA BALANCESIN
 BUDGET CATEGORIES
2510 PRINT "YOU CAN ADD OR SUBTRACT FROM
 ANY       BUDGET CATEGORY. HIT |RETURN|
WHEN DONE":?
2515 ? :? "YOU HAVE ";P(30);" UNALLOCATE
D DOLLARS"
2520 PRINT :PRINT "WHICH BUDGET CATEGORY
":INPUT T$
2525 IF T$="" THEN RETURN
2530 PRINT "ENTER AMOUNT":INPUT X
2535 I=ASC(T$)-64
2540 P(I)=P(I)+X
2545 P(30)=P(30)-X
2550 GOTO 2515
3000 REM SUBROUTINE TO PUT IN DATA
3010 LINE=P(0)+10000+1
3020 ? CHR$(125)
3022 ? "CHECKING ACCOUNT BALANCE = $";P(
27):?
3025 IF T$="N" THEN ? "REENTER DATA":?
3030 ? "ENTER CHECK NO. OR CREDIT CARD
```

```
      OR |RETURN| WHEN THROUGH ":INPUT N
$
3035 IF N$="" THEN RETURN
3036 N=ASC(N$)
3040 ? "ENTER DATE MM/DD":INPUT D$
3041 IF LEN(D$)<>5 THEN ? "REENTER DATE.
 REMEMBER FORMAT MM/DD":GOTO 3040
3042 D$(6,8)=Y$
3050 ? "ENTER PAYEE":INPUT I$
3060 ? "ENTER AMOUNT":INPUT A$
3062 IF N<>43 THEN ? "SALES TAX ON THIS
(Y OR N)":INPUT Q$
3065 IF N=43 THEN ? "DO YOU WANT THIS DE
POSIT ALLOCATED TO A CATEGORY":INPUT T$:
IF T$="N" THEN I=30:GOTO 3073
3070 ? "ENTER CATEGORY":INPUT W$
3071 I=ASC(W$)-64
3072 IF I<1 OR I>26 THEN ? "ILLEGAL CATE
GORY,TRY AGAIN":GOTO 3030
3073 ? "IS DATA OK? (Y OR N)":INPUT T$
3074 IF T$="N" THEN GOTO 3010
3076 IF I=30 THEN W$=" "
3080 P(0)=P(0)+1
3085 X=VAL(A$)
3087 IF N=43 THEN X=-X:Q$="N"
3100 P(I)=P(I)-X
3110 IF N<65 THEN P(27)=P(27)-X
3120 IF N>64 THEN P(N-64)=P(N-64)+X
3400 GOSUB 3500
3450 GOTO 3010
3500 REM SUBROUTINE TO PUT DATA STATEMEN
TS IN PROGRAM
3510 ? CHR$(125)
3520 ? "{DOWN}";LINE;" DATA ";N$;C$;D$;C
$;I$;C$;A$;C$;W$;C$;Q$
3530 ? :? :? "CONT"
3540 POSITION 0,0
3550 POKE 842,13:STOP
3560 POKE 842,12
3570 RETURN
4000 REM SUBROUTINE TO PRINT CHECK REGIS
TER, CHARGE REGISTER OR A CATEGORY
4004 B=0
4005 TRAP 4064
4010 RESTORE 10000
4025 IF WHICH=8 THEN ? "WHICH BUDGET CAT
EGORY":INPUT T$
4026 IF WHICH=3 THEN ? "WHICH CHARGE CAR
D":INPUT T$
4027 ? "ENTER STARTING MONTH AND ENDING
MONTH (M1,M2)"
4028 INPUT M1,M2
4029 I=0:? CHR$(125)
4030 ? "NUM. DATE      PAYEE           A
MOUNT"
4032 ? "--------------------------------
------"
4040 SOUND 0,INT(RND(0)*255),10,10:READ
N$,D$,I$,A$,W$,Q$
4041 IF VAL(D$(1,2))<M1 THEN GOTO 4040
4042 IF VAL(D$(1,2))>M2 THEN GOTO 4040
4044 IF WHICH=8 AND T$<>W$ THEN GOTO 404
0
4046 IF WHICH=2 AND ASC(N$)>64 THEN GOTO
 4040
4048 IF WHICH=3 AND N$(1,1)<>T$ THEN GOT
O 4040
4050 Z$=S$
4052 Z$(1,4)=N$:Z$(6,13)=D$:Z$(15,30)=I$

4053 Z$(39-LEN(A$),38)=A$
4054 IF ASC(N$)=43 THEN B=B-VAL(A$):GOTO
 4056
4055 B=B+VAL(A$)
4056 PRINT Z$;
4057 I=I+1:IF I=15 THEN ? :? "THERE'S MO
RE. HIT |RETURN| TO CONTINUE":INPUT W$:G
OTO 4029
4060 GOTO 4040
4064 ? :SOUND 0,0,0,0
4065 IF WHICH=2 THEN ? B$;P(27)
4067 IF WHICH=8 OR WHICH=3 THEN ? B$;P(A
SC(T$)-64)
4068 ? :? "TOTAL FOR ABOVE = ";B
4070 ? :? :? "HIT |RETURN| WHEN THROUGH"
:INPUT W$
4080 RETURN
5000 REM SUBROUTINE TO PUT P ARRAY IN DA
TA STATEMENTS 200-230
5005 FOR I=0 TO 30
5007 SOUND 0,INT(RND(0)*255),10,10
5010 ? CHR$(125)
5015 X=P(I)
5020 ? "{DOWN}";200+I;" DATA ";X
5030 ? :? :? "CONT"
5040 POSITION 0,0
5050 POKE 842,13:STOP
5060 POKE 842,12
5065 SOUND 0,0,0,0
5070 NEXT I
5080 ? CHR$(125)
5090 ? "SAVE PROGRAM AND DATA FOR NEXT S
ESSION"
5100 ? :? :? "    1. TURN OVER TAPE AND
REWIND."
5110 ? :? "    2. PRESS |RECORD| AND |PL
AY|."
5120 ? :? "    3. PRESS |RETURN| AFTER B
UZZER RINGS     TWICE."
5130 TRAP 5150
5140 LPRINT
5150 CSAVE
5160 END
6000 REM SUBROUTINE TO PERFORM SEARCH
6005 TRAP 6150
6020 PRINT "ENTER CHARACTERS TO BE SEARC
HED"
6030 INPUT SE$
6032 PRINT CHR$(125)
6034 ? "NUM. DATA      PAYEE           A
```

```
MOUNT"
6036 ? "--------------------------------
-------"
6040 X=LEN(SE$)
6045 B=0
6050 RESTORE 10000
6060 READ N$,D$,I$,A$,W$,Q$
6065 IF LEN(I$)<X THEN GOTO 6060
6070 FOR I=1 TO LEN(I$)-X+1
6080 IF I$(I,I+X-1)=SE$ THEN GOTO 6100
6090 NEXT I
6095 GOTO 6060
6100 POP
6110 Z$=S$
6112 Z$(1,4)=N$:Z$(6,13)=D$:Z$(15,30)=I$
6114 Z$(39-LEN(A$),38)=A$
6116 PRINT Z$;
6118 B=B+VAL(A$)
6120 GOTO 6060
6150 ? :? "TOTAL FOR ABOVE = ";B
6200 ? :? "HIT |RETURN| WHEN DONE":INPUT
 T$
6210 RETURN
7000 REM SUBROUTINE TO CALCULATE SALES T
AX
7010 ? "ENTER TAX RATE IN PERCENT"
7020 INPUT X
7022 X=X/100
7025 ? :? :? "CALCULATING... PLEASE WAIT
"
7030 TRAP 7100
7040 B=0
7050 RESTORE 10000
7060 READ N$,D$,I$,A$,W$,Q$
7070 IF Q$<>"Y" THEN GOTO 7060
7080 B=B+VAL(A$)
7090 GOTO 7060
7100 TAX=B*(1-X)*X
7105 TAX=INT(TAX*100)/100
7110 ? :? :PRINT "TOTAL TAXABLE PURCHASE
S = ";B
7120 PRINT :PRINT "SALES TAX PAID = ";TA
X;" DOLLARS"
7130 ? :? :PRINT "HIT |RETURN| WHEN THRO
UGH":INPUT T$
7140 RETURN
9000 DATA Automobile
9010 DATA Contributions
9020 DATA Household
9030 DATA Utilities
9040 DATA Insurance
9050 DATA Doctors and Dentists
9060 DATA Food
9070 DATA Kid's Clothes
9080 DATA Large Purchases
9090 DATA Tuition
9100 DATA Savings
9110 DATA Mastercard Escrow
9120 DATA Visa Escrow
9500 DATA END
```

Program 2: Commodore Version

```
1 REM HOUSEHOLD CHECK, CREDIT CARD,
2 REM AND BUDGET PROGRAM
3 REM PET/CBM VERSION
10 GOSUB9700
45 PRINTCHR$(147)
50 IF FRE(0)>1000 THEN 70
60 PRINT"{CLEAR}{REV} W A R N I N G {OFF}"
    :PRINT"ROOM FOR ABOUT";INT(FRE(0)/
    50);"MORE ENTRIES."
70 PRINT"PROGRAM LAST RUN ON";P(28);"/";P(
    29)
80 PRINT:PRINT:INPUT"ENTER TODAY'S  DATE
    (MM,DD)";X,B
90 P(28)=X:P(29)=B
200 DATA 0,0,0,0,0
201 DATA 0,0,0,0,0
202 DATA 0,0,0,0,0
203 DATA 0,0,0,0,0
204 DATA 0,0,0,0,0
205 DATA 0,0,0,0,0,0
231 REM CHANGE YEAR
232 DATA /82
999 PRINTCHR$(147)
1000 PRINT"WHAT DO YOU WANT TO DO?":PRINT:PR
     INT
1010 PRINT"1. ENTER CHECKS AND CHARGES":PRIN
     T
1020 PRINT"2. PRINT CHECK REGISTER":PRINT
1030 PRINT"3. PRINT CHARGE REGISTER":PRINT
1040 PRINT"4. UPDATE BALANCE IN BUDGET CATEG
     ORIES":PRINT
1050 PRINT"5. CALCULATE SALES TAX":PRINT
1060 PRINT"6. PRINT BALANCES IN ALL CATEGORI
     ES":PRINT
1070 PRINT"7. PERFORM SEARCH FOR CHARGE OR C
     HECK":PRINT
1080 PRINT"8. EXAMINE AN INDIVIDUAL CATEGORY
     ":PRINT
1090 PRINT"9. STORE DATA AND END SESSION"
1140 PRINT:PRINT:INPUT "ENTER NUMBER";WHICH
1155 ON WHICH GOSUB 3000,4000,4000,2500,7000
     ,2000,6000,4000,5000
1998 GOTO999
2000 REM PRINTS BALANCES IN ALL CATEGORIES
2020 PRINT"{CLEAR}{REV}BALANCES IN ALL CATEG
     ORIES":PRINT:PRINT
2030 RESTORE:FORI=1TO32:READZ9$:NEXT
2040 FORI=1TO26:READZ$:IFZ$="END"THEN2100
2070 PRINTZ$;TAB(25);" = $";P(ASC(Z$)-64)
2080 NEXTI
2100 PRINT"UNALLOCATED DEPOSITS";TAB(25);" =
     $";P(30)
2120 PRINT"CHECKING ACCOUNT BALANCE";TAB(25)
     ;" = $";P(27)
2450 PRINT"{DOWN}HIT {REV}RETURN{OFF} WHEN T
     HROUGH"
2455 GETT$:IFT$=""THEN2455
2460 RETURN
2500 REM SUBROUTINE TO UPDATE BALANCES IN BU
     DGET CATEGORIES
2510 PRINT "{CLEAR}YOU CAN ADD OR SUBTRACT F
     ROM ANY":PRINT"BUDGET CATEGORY."
2511 PRINT"{DOWN}HIT {REV}RETURN{OFF} WHEN D
     ONE:":PRINT
2515 PRINT:PRINT"YOU HAVE";P(30);"UNALLOCATE
     D DOLLARS"
2520 PRINT:INPUT"WHICH BUDGET CATEGORY?_*{03
     LEFT}";T$
2525 IF T$="*"THEN RETURN
2530 INPUT"ENTER AMOUNT";X
2535 I=ASC(T$)-64:IFI<1ORI>26THENRETURN
```

```
2540 P(I)=P(I)+X
2545 P(30)=P(30)-X
2550 GOTO 2515
3000 REM SUBROUTINE TO PUT IN DATA
3010 LINE=P(0)+10000+1
3020 PRINTCHR$(147)
3022 PRINT"CHECKING ACCOUNT BALANCE =$";P(27
    ):PRINT
3025 IFT$="N"THENPRINT"REENTER DATA":PRINT
3030 PRINT"{DOWN}ENTER CHECK NO. OR CREDIT C
    ARD":PRINT"{DOWN}OR {REV}RETURN{OF
    OFF} WHEN THROUGH"
3031 INPUT"{DOWN}?_*{03 LEFT}";N$
3035 IFN$="*"THENRETURN
3036 N=ASC(N$)
3040 INPUT"ENTER DATE MM/DD";D$
3041 IFLEN(D$)<5THENPRINT"REENTER DATE. REME
    MBER FORMAT MM/DD":GOTO3040
3042 D$=D$+Y$
3050 INPUT"ENTER PAYEE";I$
3060 INPUT"AMOUNT";A$
3062 IFN<>43THENINPUT"SALES TAX ON THIS (Y O
    R N)";Q$
3065 IFN=43THENINPUT"DO YOU WANT THIS DEPOSI
    T ALLOCATED TO A CATEGORY";T$
3066 IFN=43ANDT$="N"THENI=30:GOTO3073
3070 INPUT"ENTER CATEGORY";W$
3071 I=ASC(W$)-64
3072 IFI<1ORI>26THENPRINT"ILLEGAL CATEGORY,
    TRY AGAIN":GOTO3030
3073 INPUT"IS DATA OK (Y/N)";T$
3074 IFT$="N"THEN3010
3076 IFI=30THENW$=" "
3080 P(0)=P(0)+1
3085 X=VAL(A$)
3087 IFN=43THENX=-X:Q$="N"
3100 P(I)=P(I)-X
3110 IFN<65THENP(27)=P(27)-X
3120 IFN>64THENP(N-64)=P(N-64)+X
3400 GOSUB3500
3450 GOTO3010
3500 REM SUBROUTINE TO PUT DATA STATEMENTS I
    N PROGRAM
3510 OTHER$=STR$(LINE)+" DATA "+N$+C$+D$+C$+
    I$+C$+A$+C$+W$+C$+Q$
3515 O2$=STR$(LINE+1)+"DATA XXXX,X,X,X,X,X"
3520 RL=3530:GOTO9600
3530 GOSUB9700:GOTO1998
4000 REM SUBROUTINE TO PRINT CHECK REGISTER,
     CHARGE REGISTER, OR A CATEGORY
4010 RESTORE:B=0:PRINT"{CLEAR}"
4011 READZ9$:IFZ9$<>"END"THEN4011
4020 IFWHICH=8THENPRINT"WHICH BUDGET CATEGOR
    Y":INPUTT$
4025 IFWHICH=3THENPRINT"WHICH CHARGE CARD":I
    NPUTT$
4027 PRINT"ENTER STARTING MONTH AND ENDING":
    PRINT"MONTH  (M1,M2)";
4028 INPUTM1,M2
4029 I=0:PRINTCHR$(147)
4030 PRINT"NUM.   DATE        PAYEE          A
    MOUNT"
4032 PRINT"---------------------------------
    -----"
4040 READ N$,D$,I$,A$,W$,Q$
4041 IFN$="XXXX"THEN4064
4042 IFVAL(LEFT$(D$,2))<M1THEN4040
4043 IFVAL(LEFT$(D$,2))>M2THEN4040
4044 IFWHICH=8ANDT$<>W$THEN4040
4046 IFWHICH=2ANDASC(N$)>64THEN4040
4048 IFWHICH=3ANDLEFT$(N$,1)<>T$THEN4040
4050 PRINTN$;TAB(6);D$;TAB(17);I$;TAB(32);A$
4054 IFASC(N$)=43THENB=B-2*VAL(A$)
4055 B=B+VAL(A$):I=I+1:IFI<15THEN4060
4056 PRINT:PRINT"THERE'S MORE":PRINT"HIT RET
    URN TO CONTINUE"
4057 GETZ9$:IFZ9$=""THEN4057
4058 GOTO4029
4059 GETA$:IFA$=""THEN4059
4060 GOTO4040
4064 PRINT
4065 IFWHICH=2THENPRINTB$;P(27)
4067 IFWHICH=8ORWHICH=3THENPRINTB$;P(ASC(T$)
    -64)
4068 PRINT:PRINT"TOTAL FOR ABOVE=";B
4070 PRINT:PRINT:PRINT"HIT {REV}RETURN{OFF}
    WHEN THROUGH"
4075 GETZ9$:IFZ9$=""THEN4075
4080 RETURN
5000 REM SUBROUTINE TO PUT P ARRAY INTO DATA
     STMTS
5010 RL=5020:GOTO9600
5020 GOSUB9700
5030 PRINT"{CLEAR}SAVE PROGRAM AND DATA FOR
    NEXT SESSION
5040 PRINT"{02 DOWN}1. TURN OVER TAPE AND RE
    WIND.
5045 PRINT"{02 DOWN}PRESS {REV}RETURN{OFF} W
    HEN YOU'VE DONE THIS"
5047 GETZ9$:IFZ9$=""THEN5047
5050 PRINT"{02 DOWN}2. PRESS PLAY & RECORD"
5060 PRINT:PRINT
5150 SAVE"BUDGET":REM OR WHATEVER NAME YOU W
    ANT
5160 END
6000 REM SUBROUTINE TO PERFORM SEARCH
6010 PRINT"{CLEAR}ENTER CHARACTERS TO BE SEA
    RCHED"
6030 INPUTSE$
6032 PRINT"{CLEAR}"
6034 PRINT"NUM.   DATE        PAYEE          A
    MOUNT"
6036 PRINT"---------------------------------
    -----"
6040 X=LEN(SE$)
6045 B=0:RESTORE
6050 READZ9$:IFZ9$<>"END"THEN6050
6060 READ N$,D$,I$,A$,W$,Q$
6065 IFN$="XXXX"THEN6150
6070 IFLEFT$(I$,X)<>SE$THEN6060
6115 PRINTN$;TAB(6);D$;TAB(17);I$;TAB(32);A$
6118 B=B+VAL(A$)
6120 GOTO6060
6150 PRINT:PRINT"TOTAL FOR ABOVE =";B
6200 PRINT:PRINT"HIT {REV}RETURN{OFF} WHEN D
    ONE"
6205 GETZ9$:IFZ9$=""THEN6205
6210 RETURN
7000 REM SUBROUTINE TO CALCULATE SALES TAX
7010 PRINT"ENTER TAX RATE IN PERCENT"
7020 INPUTX
7022 X=X/100
7028 PRINT:PRINT:PRINT"CALCULATING...PLEASE
    WAIT"
7040 B=0:RESTORE
7050 READZ9$:IFZ9$<>"END"THEN7050
7060 READ N$,D$,I$,A$,W$,Q$
7070 IFQ$<>"Y"THEN7090
7080 B=B+VAL(A$)
7090 IFN$<>"XXXX"THEN7060
7100 TAX=B*(1-X)*X
7105 TAX=INT(TAX*100)/100
7110 PRINT:PRINT:PRINT"TOTAL TAXABLE PURCHAS
    ES = ";B
```

```
7120 PRINT:PRINT:PRINT"SALES TAX PAID=";TAX;
     "DOLLARS"
7130 PRINT:PRINT"HIT {REV}RETURN{OFF} WHEN D
     ONE"
7140 GETZ9$:IFZ9$=""THEN7140
7150 RETURN
9000 DATA AUTOMOBILE
9010 DATA CONTRIBUTIONS
9020 DATA HOUSEHOLD
9030 DATA UTILITIES
9040 DATA INSURANCE
9050 DATA DOCTORS AND DENTISTS
9060 DATA FOOD
9070 DATA KID'S CLOTHES
9080 DATA LARGE PURCHASES
9090 DATA TUITION
9100 DATA SAVINGS
9110 DATA MASTERCARD ESCROW
9120 DATA VISA ESCROW
9130 DATA END
9600 REM SUBROUTINE TO PUT INFO INTO DATA ST
     ATEMENTS
9610 PRINT"{CLEAR}{02 DOWN}":FORI=0TO5:PRINT
     200+I;"DATA ";
9615 FORJ=0TO4-(I=5)
9617 A=P(I*5+J):A=INT(A*1000+.5)/1000
9620 PRINTMID$(STR$(A),2);",";:NEXT:PRINTCHR
     $(20):NEXT
9630 PRINTOTHER$:PRINTO2$
9640 PRINT"GOTO";RL:PRINT"{HOME}"
9650 REM DYNAMIC KEYBOARD
9660 POKE 158,9
9670 FORI=0TO8:POKE623+I,13:NEXT
9680 END
9700 S$="
     "
9710 N$=S$:D$=S$:I$=S$:A$=S$:W$=S$:C$=",":Z$
     =S$:T$=S$:B$="BALANCE ="
9720 SE$=S$:Q$=S$
9730 DIMP(30)
9740 RESTORE:FORI=0TO30:READP(I):NEXT:READY$

9750 RETURN
```

©

How can a teacher use one computer to teach spelling to an entire class? One solution is to print out individualized study sheets. This program (versions for Microsoft – Apple, Commodore, OSI, and Radio Shack – and Atari BASICs) creates several different word games. They are good for spelling practice, but they're enjoyable games, too. If you don't have a printer, you can redirect the output to the screen by making a few adjustments such as substituting PRINT for PRINT#, ignoring PRINT#3 (special PET printer formatting commands), and by not OPENing to the printer. See the special Atari instructions under "Atari Notes."

Word Games

August Schau
Buckfield, ME

Teachers wishing to use microcomputers to individualize instruction can easily be frustrated because they lack the hardware necessary for students to interact with a microcomputer on a regular basis.

I doubt that our local situation is unique. The local school district is contemplating the purchase of three Commodore 4016's on Commodore's three-for-two offer. This hardware will serve 500 students and 40 staff members located in three buildings. Future hardware acquisitions will probably be modest, and the prospect of multiple stations for student use in the near future is not realistic.

While waiting for the administration to consider an investment in microcomputers, I have purchased a CBM 8032 with 8050 disk drive and model 4022 printer and have begun to explore its application as a behind-the-scenes resource.

Having a computer-supported teacher might be a cost-effective way to maximize benefits from a limited amount of hardware.

One Computer Can Serve Many Students

Teachers can share a computer/printer combination to assist with the record keeping associated with classroom management and to prepare individualized instructional materials. Materials for individual students can be used as printed, and satisfactory copies for group use can be made from a printout using a thermofax master and spirit duplicator.

The program will generate individual spelling activities.

Lines 90-115 – ten words are INPUT and stored in A$ array. Each word is scrambled in subroutine 225, returned and stored in B$ array.

Lines 120-145 – each word is sent to subroutine 300 to have "_" substituted for vowels, is returned and stored in C$ array.

Line 200 prints A$ array in order 1-10, B$ array in random order, and C$ array in random order following the format on line 55.

Subroutine 660 fills the puzzle matrix with random letters and the answer matrix with "#".

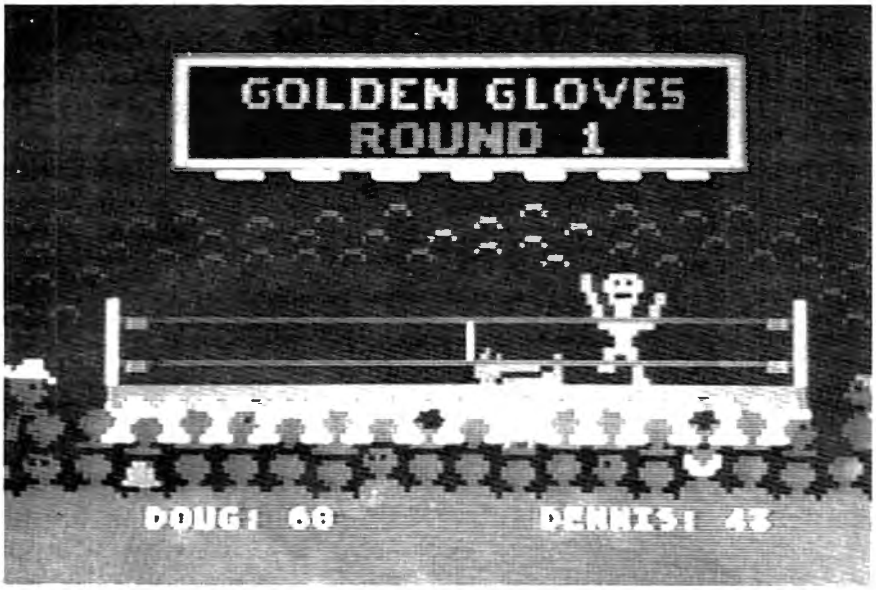
Lines 420 and 425 select a starting point. Words are presented left to right and top to bottom to reinforce patterns common to written language. The starting point is shifted in lines 450 and 520 if the word does not fit. If there is a letter already occupying one of the spaces needed, a new starting point is selected and tested.

Lines 600-645 print the two matrixes side by side following the format on line 385.

Program 1. Microsoft Version

```
20 GOSUB 710
25 PRINT "PRESS RETURN AFTER EACH ENTRY":P
   RINT
30 INPUT "STUDENT'S NAME";Q$
35 OPEN 1,4
40 SP$=CHR$(29)
45 OPEN 4,4,1
50 OPEN 3,4,2
55 PRINT#3,"          AAAAAAAAAA
   AAAAAAAAA         AAAAAAAAAA"
60 FOR I=1 TO 10
65 PRINT "WORD ";I;" IS ";
70 INPUT A$(I)
75 NEXT I
80 PRINT:PRINT
85 PRINT"JUST A MINUTE!"
90 FOR I=1 TO 10
95 LET X$=A$(I)
100 GOSUB 225
105 LET B$(I)=E$
110 LET E$=""
115 NEXT I
120 FOR J= 1 TO 10
125 LET X$=A$(J)
130 GOSUB 300
135 LET C$(J)=Z$
140 LET Z$=""
145 NEXT J
150 PRINT#1,"__"CHR$(1)Q$
155 PRINT#1:PRINT#1
160 PRINT#4,"WORD LIST"SP$"SCRAMBLE"SP$"FIL
    L IN"
165 PRINT#4:PRINT#4
170 REM PRINT OUT RESULTS
175 FOR I=1 TO 10
180 LET R=INT(RND(1)*10)+1
185 IF B$(R)="0"THEN 180
190 LET S=INT(RND(1)*10)+1
195 IF C$(S)="0" THEN 190
200 PRINT#4,A$(I)SP$B$(R)SP$C$(S)
205 LET B$(R)="0":LET C$(S)="0"
210 PRINT#4
```

ARTWORX SCORES ANOTHER TECHNICAL KNOCKOUT.



Scene from GOLDEN GLOVES

HODGE PODGE: by Marsha Meredith (Atari and Apple)
NOW AVAILABLE FOR ATARI!!! This captivating program is a marvelous learning device for children from 18 months to 6 years. HODGE PODGE consists of many cartoons, animation and songs which appear when any key on the computer is depressed. A must for any family containing young children.
PRICE $19.95 diskette

BETA FIGHTER: by Douglas McFarland (Atari, 16K)
See who will be the ace gunner in this action game set on a spectacular Martian landscape. BETA FIGHTER can be played with one or two players and uses player missile graphics and delightful sound effects.
PRICE $16.95 cassette $20.95 diskette

DRAWPIC: by Dennis Zander (Atari 16K)
DRAWPIC provides the user with an unbelievably easy way to create screens in graphics modes 3-7. Just sit back with your joystick and use POINT PLOT, DRAW LINE, RUBBER BAND fill and COLOR SET to create beautiful images on your Atari. Full or partial screen images are saved as string data in the program and can be instantly recalled and combined into new images using machine language subroutines. These graphic images can be easily incorporated into your own programs. The images of HODGE PODGE and the landscape of BETA FIGHTER were made using DRAWPIC.
PRICE $29.95 cassette $33.95 diskette

ROCKET RAIDERS by Richard Petersen (Atari 24K
Defend your asteroid base against pulsar bombs, rockets, lasers and the dreaded "stealth saucer" as aliens attempt to penetrate your protective force field. Precise target sighting allows you to fire at the enemy using magnetic impulse missiles to help protect your colony and its vital structures.
PRICE $19.95 cassette $23.95 diskette

FOREST FIRE TWO: by Richard Petersen (Atari 24K
FOREST FIRE has been enhanced and now offers a two player mode for head to head competition to see who can survive, suffer the least damage and put their fire out first. User input now determines landscape, wind and weather conditions offering limitless game variation. FOREST FIRE's excellent color graphics have been made even better, turning your computer into a super detailed fire scanner.
PRICE $16.95 cassette $20.95 diskette

☐ **FORM LETTER SYSTEM:** (Atari, North Star and Apple)
This is the ideal program for creating personalized form letters! FLS employs a simple to use text editor for producing fully justified letters. Addresses are stored in a separate file and are automatically inserted into your form letter along with a personalized salutation. Both letter files and address files are compatible with **ARTWORX** MAIL LIST and TEXT EDITOR programs.
PRICE $39.95 diskette

☐ **PILOT:** by Michael Piro (Atari, 16K)
Pilot your small airplane to a successful landing using both joysticks to control throttle and attack angle. PILOT produces a true perspective rendition of the runway, which is constantly changing. Select from two levels of pilot proficiency.
PRICE $16.95 cassette $20.95 diskette

☐ **TEXT EDITOR:** (Atari and North Star)
This program is very "user friendly" yet employs all essential features needed for serious text editing with minimal memory requirements. Features include common sense operation, two different justification techniques, automatic line centering and straightforward text merging and manipulation. TEXT EDITOR files are compatible with **ARTWORX** FORM LETTER SYSTEM.
PRICE $39.95 diskette

☐ **MAIL LIST 3.0:** (Atari, Apple and North Star)
The very popular MAIL LIST 2.2 has now been upgraded. Version 3.0 offers enhanced editing capabilities to complement the many other features which have made this program so popular. MAIL LIST is unique in its ability to store a maximum number of addresses on one diskette (typically between 1200 and 2500 names!). Entries can be retrieved by name, keyword(s) or by zip codes. They can be written to a printer or to another file for complete file management. The program produces 1, 2 or 3 up address labels and will sort by zip code (5 or 9 digits) or alphabetically (by last name). Files are easily merged and MAIL LIST will even find and delete duplicate entries! The address files created with MAIL LIST are completely compatible with **ARTWORX** FORM LETTER SYSTEM.
PRICE $49.95 diskette

☐ **THE VAULTS OF ZURICH:** by Felix and Greg Herlihy (Atari, 24K, PET)
Zurich is the banking capital of the world. The rich and powerful deposit their wealth in its famed impregnable vaults. But you, as a master thief, have dared to undertake the boldest heist of the century. You will journey down a maze of corridors and vaults, eluding the most sophisticated security system in the world. Your goal is to reach the Chairman's Chamber to steal the most treasured possession of all: THE OPEC OIL DEEDS!
PRICE $21.95 cassette $25.95 diskette

☐ **BRIDGE 2.0** by Arthur Walsh (Atari (24K), Apple, TRS-80, PET, North Star and CP/M (MBASIC) systems)
Rated #1 by Creative Computing, BRIDGE 2.0 is the only program that allows you to both bid for the contract and play out the hand (on defense or offense!). Interesting hands may be replayed using the "duplicate" bridge feature. This is certainly an ideal way to finally learn to play bridge or to get into a game when no other (human) players are available.
PRICE $17.95 cassette $21.95 diskette

☐ **ENCOUNTER AT QUESTAR IV:** by Douglas McFarland (Atari, 24K)
As helmsman of Rikar starship, you must defend Questar Sector IV from the dreaded Zentarians. Using your plasma beam, hyperspace engines and wits to avoid Zentarian mines and death phasers, you struggle to stay alive. This BASIC/Assembly level program has super sound, full player missile graphics and real time action.
PRICE $21.95 cassette $25.95 diskette

NEW PROGRAMS!

GOLDEN GLOVES: by Douglas Evans (Atari 24K)
Use your joystick to jab, block and duck as each player attempts to land the knockout punch. This unique real-time program brings all of the excitement of ringside to your Atari. GOLDEN GLOVES is a one or two-player game, or you can be a spectator as the computer controls both fighters.
PRICE . . . .**$22.95** cassette **$26.95** diskette

CRAZITACK: by Peter Adams (Atari 16K)
The Crazies are attacking us and the only defenses are three MX bases. Missiles can be launched singly or in a salvo, but it is doomsday when you run out of missiles.
PRICE . . . .**$17.95** cassette **$21.95** diskette

DOMINATION: by Alan Newman (Atari 24K)
Between one and six players compete for power via economic, diplomatic and military means in this award-winning game. You must make decisions quickly, exercise skillful hand-eye coordination, outguess your opponents and cope with random events.
PRICE . . . .**$17.95** cassette **$21.95** diskette

POKER TOURNEY: by Edward Grau (Atari 32K, Northstar)
You are entered in a high stakes Draw Poker Tournament facing six opponents including Lakewood Louie, Shifty Pete and Dapper Dan. Each has his own style of play and of bluffing. POKER TOURNEY utilizes the Joker, has true table stakes play and each hand is played based on pot odds. The Atari version's graphics and sound are superb of course (programmed by Jerry White) making POKER TOURNEY the class program of its type.
PRICE . . . .**$18.95** cassette **$22.95** diskette

HAZARD RUN: by Dennis Zander (Atari, 16K)
The sheriff has spotted you and you must make the treacherous run through Crooked Canyon past Bryan's Pond to the jump at Hazard Creek and safety. You can even put the joystick-controlled GEE LEE car up on two wheels to make it through some tight spots. A lead foot is not always the answer as you dodge trees, rocks and chickens in this nerve-racking game. HAZARD RUN employs full use of player/missile graphics, re-defined characters and fine scrolling techniques to provide loads of fast action and visual excitement.
PRICE. . . . .**$27.95** cassette **$31.95** diskette

ORDERING INFORMATION

Call ARTWORX toll-free number to order direct:
800-828-6573
In New York, Alaska, Hawaii call:
(716) 425-2833

All orders are processed and shipped within 48 hours.

Shipping and handling charges:
Within North America: Add $2.00
Outside North America: Add 10% (Air Mail)
New York State residents add 7% sales tax

Quantity Discounts:
Deduct 10% when ordering 3 or more programs

Ask for ARTWORX at your local computer store.

Write for FREE Catalogue listing more information about these and other quality ARTWORX programs.

Artworx
150 North Main Street Fairport, NY 14450

```
215 NEXT I
220 GOTO 360
225 REM WORD SCRAMBLER
230 N=LEN(X$)
235 FOR J=1 TO N
240 LET Q(J)=J
245 NEXT J
250 LET C=0
255 LET Y=INT(RND(1)*N)+1
260 IF C=0 AND Y=1 THEN 255:REM WORD CANNOT
     START WITH FIRST LETTER
265 IF Q(Y)=0 THEN 255
270 IF Y=Q(Y) THEN Q(Y)=0
275 LET E$=E$+MID$(X$,Y,1)
280 LET C=C+1
285 IF C=N THEN 295
290 GOTO 255
295 RETURN
300 REM VOWELS = BLANKS
305 LET N=LEN(X$)
310 FOR I=1 TO N
315 LET X=ASC(MID$(X$,I,1))
320 REM TO OMIT "Y" FROM FILL IN ADD X=89 T
    O NEXT LINE
325 IF X=65 OR X=69 OR X=73 OR X=79 OR X=85
     THEN 340
330 LET Y$=MID$(X$,I,1)
335 GOTO 345
340 LET Y$=CHR$(164)
345 LET Z$=Z$+Y$
350 NEXT I
355 RETURN
360 CLOSE 3:CLOSE 4:CLOSE 1
365 OPEN 1,4,1
370 OPEN 2,4,2
375 PRINT:PRINT
380 PRINT "STILL WORKING ON IT!"
385 PRINT#2,"           AAAAAAAAAAAAAAAAAAAA
          AAAAAAAAAAAAAAAAAAAA"
390 DIM K$(15,20)
395 DIM P$(15,20)
400 GOSUB 660
405 LET D=D+1
410 F$=A$(D)
415 REM PICK RANDOM STARTING POINT
420 LET A=INT(RND(TI)*20)+1:X=A
425 LET B=INT(RND(TI)*15)+1:Y=B
430 REM PICK HORIZONTAL OR VERTICAL
435 LET S=INT(RND(TI)*2)+1
440 ON S GOTO 445,515
445 REM HORIZONTAL
450 IF A+LEN(F$)>20 THEN A=A-LEN(F$)
455 X=A
460 FOR I=1 TO LEN(F$)
465 IF K$(B,A)<>"#"THEN 420
470 A=A+1
475 NEXT I
480 FOR I=1 TO LEN(F$)
485 P$(Y,X)=MID$(F$,I,1):REM STORE WORD IN
    PUZZLE MATRIX
490 K$(Y,X)=MID$(F$,I,1):REM STORE WORD IN
    ANSWER MATRIX
495 X=X+1
500 NEXT I
505 IF D<10 THEN 405
510 GOTO 580
515 REM VERTICAL
520 IF B+LEN(F$)>15 THEN B=B-LEN(F$)
525 Y=B
530 FOR I=1 TO LEN(F$)
535 IF K$(B,A)<>"#" THEN 420
540 LET B=B+1
545 NEXT I
550 FOR I=1 TO LEN(F$)
555 LET P$(Y,X)=MID$(F$,I,1):REM STORE WORD
     IN PUZZLE MATRIX
560 LET K$(Y,X)=MID$(F$,I,1):REM STORE WORD
     IN ANSWER MATRIX
565 Y=Y+1
570 NEXT I
575 IF D<10 THEN 405
580 REM PRINT PUZZLE
585 PRINT#1:PRINT#1:PRINT#1:PRINT#1
590 PRINT#1,"WORDSEARCH"SP$Q$
595 PRINT#1
600 FOR Y=1 TO 15
605 FOR X=1 TO 20
610 PRINT#1,P$(Y,X);
615 NEXT X
620 PRINT#1,SP$;
625 FOR X=1 TO 20
630 PRINT#1,K$(Y,X);
635 NEXT X
640 PRINT#1
645 NEXT Y
650 CLOSE 1:CLOSE 2
655 GOTO 705
660 REM FILL P MATRIX WITH RANDOM LETTERS
665 FOR Y=1 TO 15
670 FOR X=1 TO 20
675 L=INT(RND(TI)*26+1)
680 P$(Y,X)=CHR$(L+64)
685 K$(Y,X)="#"
690 NEXT X
695 NEXT Y
700 RETURN
705 GOTO 805
710 POKE 59468,12: PRINT "{CLEAR} WORD DRIL
    L"
715 FOR I= 1 TO 12: PRINT CHR$(163);: NEXT
720 PRINT: PRINT
725 PRINT "BY AUGUST SCHAU"
730 PRINT:PRINT"RFD #1"
735 PRINT:PRINT"BUCKFIELD, ME. 04220"
740 PRINT:PRINT
745 PRINT "INPUT STUDENT'S NAME AND 10 WORD
    S.
750 PRINT:PRINT "PRINTOUT INCLUDES:"
755 PRINT:PRINT "1) THE LIST OF WORDS
760 PRINT "2) A RANDOM LIST OF SCRAMBLED WO
    RDS
765 PRINT "3) A RANDOM LIST OF WORDS WITH B
    LANKS
770 PRINT "   SUBSTITUTED FOR VOWELS
775 PRINT "4) A WORD SEARCH IN A 15X20 GRID

780 PRINT "5) AN ANSWER GRID
785 PRINT:PRINT
790 PRINT "PRESS {REV}SPACE BAR{OFF} TO CON
    TINUE"
795 GET V$:IF V$="" THEN 795
800 PRINT"{CLEAR}":RETURN
```

Program 2. Atari Version

```
10 REM WORD DRILL FOR ATARI
15 DIM Q$(30),T$(20),A$(20*10),AL(10),X$(20)
,Z$(20),B$(200),C$(200),E$(20)
16 DIM Q(20)
17 DIM TAB$(80):TAB$=" ":TAB$(80)=" ":TAB$(2
)=TAB$
20 GOSUB 710
25 ? "Press RETURN after each entry":?
```

```
30 ? "Student's name";:INPUT Q$
60 FOR I=1 TO 10
65 ? "WORD ";I;" IS ";
70 INPUT T$:A$(I*20-19,I*20)=T$:AL(I)=LEN(T$
)
75 NEXT I
80 ? :?
85 ? "Just a minute!"
90 FOR I=1 TO 10
95 X$=A$(I*20-19,I*20-20+AL(I))
100 GOSUB 225
105 B$(I*20-19,I*20-20+AL(I))=E$
110 E$=""
115 NEXT I
120 FOR J=1 TO 10
125 X$=A$(J*20-19,J*20-20+AL(J))
130 GOSUB 300
135 C$(J*20-19,J*20-20+AL(J))=Z$
140 Z$=""
145 NEXT J
150 LPRINT Q$:LPRINT :LPRINT
160 LPRINT "WORD LIST{10 SPACES}SCRAMBLE
    {11 SPACES}FILL IN"
165 LPRINT :LPRINT
170 REM PRINT OUT RESULTS
175 FOR I=1 TO 10
180 R=INT(10*RND(0)+1)
185 X$=B$(R*20-19,R*20-20+AL(R)):IF X$(1,1)=
    "0" THEN 180
190 S=INT(10*RND(0)+1)
195 Z$=C$(S*20-19,S*20-20+AL(S)):IF Z$(1,1)=
    "0" THEN 190
200 T$=A$(I*20-19,I*20-20+AL(I))
201 LPRINT T$;TAB$(1,20-LEN(T$));X$;TAB$(1,2
    0-LEN(X$));Z$
210 LPRINT
215 B$(R*20-19,R*20-18)="0":C$(S*20-19,S*20-
    18)="0"
217 NEXT I
220 GOTO 360
225 REM WORD SCRAMBLER
230 N=LEN(X$)
235 FOR J=1 TO N
240 Q(J)=J
245 NEXT J
250 C=0
255 Y=INT(N*RND(0)+1)
260 IF C=0 AND Y=1 THEN 255:REM WORD CANNOT
    START WITH FIRST LETTER
265 IF Q(Y)=0 THEN 255
270 IF Y=Q(Y) THEN Q(Y)=0
275 E$(LEN(E$)+1)=X$(Y,Y)
280 C=C+1
285 IF C=N THEN RETURN
290 GOTO 255
300 REM VOWELS = BLANKS
305 N=LEN(X$)
310 FOR I=1 TO N
320 X=ASC(X$(I))
325 IF X=65 OR X=69 OR X=73 OR X=79 OR X=85
    THEN 340
330 Y=X
335 GOTO 345
340 Y=ASC("_")
345 Z$(LEN(Z$)+1)=CHR$(Y)
350 NEXT I
355 RETURN
360 LPRINT
370 ? :?
390 END
710 GRAPHICS 0:? "WORD DRILL"
720 ? :? "Enter student's name"
730 ? "and 10 words.  Printout"
740 ? "includes:":?
750 ? "1) The list of words"
760 ? "2) A random list of scrambled"
770 ? "{3 SPACES}words."
780 ? "3) A random list of words with"
790 ? "{3 SPACES}blanks substituted for vowe
    ls."
800 ? :?
810 ? "Press SPACE BAR to continue"
820 IF PEEK(764)<>33 THEN 820
830 POKE 764,255:GRAPHICS 0:RETURN
```

Atari Notes

This program uses an 80-column printer such as the Atari 825 or the Epson MX-80 printer. To re-route output to the screen, change all LPRINT statements to PRINT statements, and decrease the table width, if possible.

There are some limitations to string-array simulation techniques (used here to translate the Microsoft version of Word Drill to Atari BASIC). Unless you're willing to perform a great deal of calculation, two-dimensional string arrays are very hard to emulate. Also, since Atari reserves memory for all strings before the string is even filled, large arrays such as A$(10,20) can easily exceed available memory. For example, with the array A$(10,20), if we want to allow each element of the string 20 characters, the amount of memory used by this "superstring" is:

10 x 20 x 20 = 4000

bytes plus overhead.

A similar situation is encountered with the word search puzzle generator in Word Drill, which uses two 15 x 20 arrays. Memory needed by the arrays alone would be over 12,000 bytes. This plus the 16K for the program would restrict the use of Word Drill to machines with 32K or more memory. To serve the greatest audience (and avoid long, complicated calculations), Word Drill for the Atari only provides the first two functions: fill in the blank and word scramble. Nevertheless, you should find it useful.

You might also want to try "Word Hunt" for the Atari, **COMPUTE!**, March 1982, #22.

Figure 1.

WORD LIST	SCRAMBLE	FILL IN
ARM	GIFNRE	L_G
HAND	OBLWE	_RM
ELBOW	ABCK	F_NG_R
WRIST	RMA	B_CK
LEG	RISTW	CH_ST
ANKLE	NALKE	_NKL_
KNEE	GLE	WR_ST
FINGER	SEHTC	_LB_W
CHEST	DANH	H_ND
BACK	NEEK	KN__

Figure 2.

Word Search

```
IXHEGJBFMFEEFQIARUHG
FWJLJMUFMSMLPUHBARED
MSVBJZRPNUHBLUCANRWS
JINJZKBACKDODQJIYVFD
KNVGLXPIHMPWEUJBGJDP
JYYXJBRGXQLFAIPQXRYT
ATQIBCJGACNGKZFCSPEW
GOWYQVHAMRUHANDKMFHG
BNCQRVQRACFINGERTMYJ
FRYNSWWMDHGANKLERONE
GLXRACSDXEKNEERNBNDB
VSCFFWGFASCLJZAAPAKA
TWRISTRNOTLETPMFYIZP
AKHZACJJPYYGSNWJHVTK
RGTZMZAGKUSHYRJVKUDK
```

Figure 3.

Sally Student

```
###########E########
###########L########
###########B########
######BACK#O########
###########W########
####################
####################
#######A###HAND#####
#######R#CFINGER####
#######M#H#ANKLE####
#########EKNEE######
#########S#L########
#WRIST###T#E########
###########G########
####################
```

©

This first installment of a two-part article reviews the features and capabilities of Commodore's new mid-priced home computer.

A First Look At The Commodore 64

Tom R Halfhill
Features Editor

Your first look at the new Commodore 64 might cause you to overlook. That is, the 64 looks so much like the familiar VIC-20 that you might miss it altogether.

That's what happened to many visitors to the Commodore booth at the recent Consumer Electronics Show in Chicago, where Commodore was proudly showing off the new machine. People had trouble at first picking it out of rows of VIC-20s on display. The 64 has the same 66-key, full-stroke keyboard and the same compact plastic case. It's a different-colored case, though, and there's a small "64" in the upper-right corner next to the power indicator light. Eagle-eyed observers also notice the slightly different port configuration. Those are the only differences – externally.

Internally, it's another story. The electronics which make the Commodore 64 tick are far removed from the workings of its lesser cousin.

Same Family, Different Branch

First, the basics. The Commodore 64 comes with 64K of Random Access Memory, hooks up to any TV with its built-in RF modulator, and costs $595. This places it midway in price and features between the 5K RAM VIC-20 at $299 and the new 128K P128 at $995 (see "Computers to Come" elsewhere in this issue). In many ways, the 64 is an interesting cross between the VIC and the P128. For example, it shares the P128's 40-column by 25-line screen display, an improvement over the VIC's 22 by 23 format. Other features in common with the P128 include 16-color graphics, the most sophisticated three-voice sound synthesizer chip in the home computer market, an optional plug-in Z-80 microprocessor board for CP/M capability, and three screen modes (the 40 by 25 text mode, 320- by 200-pixel high resolution graphics, and 200- by 160-pixel medium resolution graphics).

In addition, the 64 has another important feature in common with the P128: "sprites" (known as player/missiles to Atari buffs). These are definable graphics figures which can be programmed to move around the screen in any mode, completely independent of the background and of each other.

On the other hand, the 64 has a few things in common with the VIC, too – mainly, its peripherals. The 64 is designed to work with the VIC's disk drives, printers, the new VICMODEM, the Datassette recorder, the light pen, joysticks, and paddles. However, before connecting the VIC-1540 disk drive to the 64, a ROM chip must be changed for full compatibility. Commodore will produce a new disk drive, called the 1541 and identical to the 1540 except for this chip, especially for the 64. Meanwhile, VIC owners switching to a 64 can upgrade their 1540 drives. Up to five of these 170K drives can be daisy-chained to the 64. Up to three more devices, such as the VIC Graphic Printer, can be chained onto the drives. The 64 also will run the wide range of CBM peripherals with an optional plug-in cartridge that provides an IEEE-488 interface.

The 64's compatibility extends to software, too. The 64 has 8K PET BASIC 2.0 (Upgrade ROM). Commodore says the cassette interface allows use of programs and files created on all other Commodore computers, and that most BASIC programs written on 40-column PETs will run without modification. Exceptions are programs which POKE screen memory locations, since these vary on different systems. For even greater software compatibility, Commodore will have a PET emulator that will transform the 64 into a PET in practically every area except machine language.

Besides all that, the 64's compatibility goes still further. It is designed to be the next logical step for owners of Commodore's new Max Machine. This $179.95 game machine with a full-size bubble-membrane keyboard uses the same processor chip, sound chip, video chip, Datassette recorder, joystick, paddles, and game cartridges as the 64. Video game lovers who cut their teeth on simple programming with the Max Machine's optional BASIC cartridge can move up to the 64 without discarding their peripherals and game software.

Chip Off The Old Block

The secret behind the Commodore 64's advanced features, yet extensive compatibility with earlier technology, is a new microprocessor chip for its Central Processing Unit (CPU). Instead of the

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

COMMODORE DEALERS

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

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

Now your Commodore can blossom into a full-blown CP/M-based or multi-user business system with the addition of one of these innovative peripherals. Your Commodore plus the SOFTBOX gives you a single-user CP/M system. Commodore plus HARDBOX puts Pet software on Corvus Disk Drives and multiplexing units.

RUN CP/M WITH SOFTBOX ON FLOPPY AND HARD DISK.

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

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

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

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

Interfaces that come with your SOFTBOX let you add the Corvus Hard Disk and any RS-232C peripheral. You can run a printer, and even talk to another CP/M based system.

90-Day Warranty. Suggested retail $895
Complete line of CP/M software available.

HARDBOX GIVES PET DOS MULTI-USER AND HARD DISK CAPABILITY.

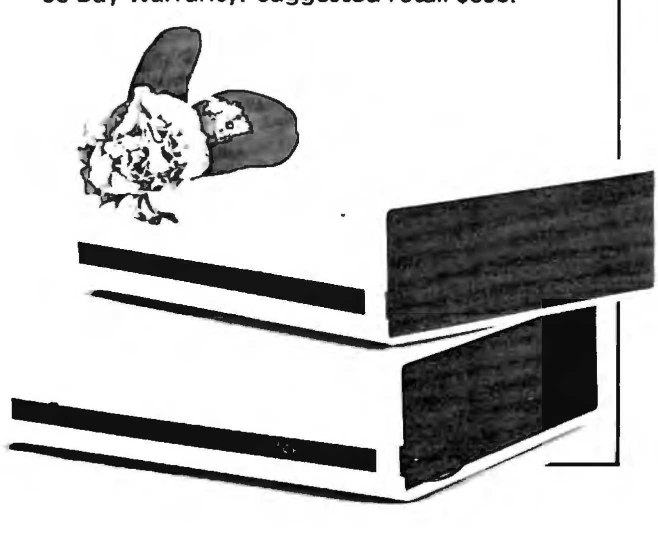
The HardBox introduces your PET software to the world of high-capacity, high speed disk storage. It acts as an intelligent controller for up to four Corvus Winchester disk drives. Each Corvus drive can have a capacity of 6, 10 or 20 million bytes.

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

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

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

90-Day Warranty. Suggested retail $695.



SSE

SMALL SYSTEMS ENGINEERING
222 B View Street
Mountain View, CA 94041
(415) 964-8201

6502 chip in earlier Commodores, the 64 has a 6510 designed by MOS Technology, the same Commodore subsidiary which designed the 6502. The 6510 has additional input/output lines, but is still, like the 6502, an eight-bit chip. Moreover, it shares the 6502's instruction set. This means machine language programmers will adapt quickly to the new chip.

Computers which rely on one chip, the CPU, for all their processing tend to be rather slow, so Commodore bestowed the 64 with additional chips to handle the time-consuming video and sound processing (a technique used also by Atari). The 6567 Video Interface Chip allows 255 combinations of border and background colors, 16 text colors and all 64 PET graphics characters. It also permits redefining custom character sets and sprites. The sound chip, called the 6581 Sound Interface Device (SID), is an extremely versatile three-voice synthesizer superior to anything found in other home/personal computers. We can say this with abandon because we've watched sound being programmed on the 64 and have heard the results.

The new chips make the Commodore 64 part of one family: the Max Machine uses the same three CPU, video, and sound chips, while the new Commodore P, B, and BX Series computers use the same SID chip for their sound (although they are based on a different, but similar, CPU, the 6509).

Add Sprite To Your Life

The standout feature of the Commodore 64's graphics is the ability to manipulate sprites. Until now, the only home computers capable of displaying sprites were the Atari 400/800 and Texas Instruments TI-99/4A. All you Commodore loyalists who used to flip by articles on Atari player/missile graphics will have to learn what it's like to struggle with this new concept in computer animation. Luckily, you'll probably have it easier than Atari people, since the new Commodores use sprites even more powerfully than the Ataris.

System Overhead: Diminishing RAM...

Theoretically, you can create up to 256 sprites on the Commodore 64. We say "theoretically" because memory limitations play a large factor, even with 64K of RAM. For one thing, not all of that 64K is available for programming; some is consumed by the system for overhead. When powered up under BASIC, a PRINT FRE(0) command to check on free memory yields 38,911 bytes. Commodore says 52K is available for machine language programming.

So how many sprites are possible? While 256 shapes can be defined in memory, one Commodore programmer says 48 is a realistic limit for display purposes. We can live with that! But there is a further limit of eight sprites per *scan line*, the horizontal lines forming the TV picture. That is, if more than eight sprites are displayed at once at the same "latitude," the extra sprites momentarily disappear as they pass by.

However, there are ways around some of these limits. The 64 has a trick similar to Atari display list interrupts. A Commodore programmer referred to it as a "raster scan interrupt." This allows the same eight sprites to be used more than once on the same screen. For example, consider a Space Invaders-type game. Eight sprites can be displayed as the top row of aliens. A split-second after that row is scanned onto the screen, an interrupt is triggered, the sprites are redefined, and finally are redisplayed as the next row of aliens. This process is repeated all the way down the screen. So you can put, say, five rows of eight aliens each (total = 40 aliens) on one screen using only the same eight sprites.

Commodore has also provided collision and priority registers to accompany the sprites. Collision registers detect when sprites bump into each other – to trigger an explosion in a game, for example. Priority registers allow programmers to define each sprite's display priority in relation to the screen background and other sprites. For instance, a sprite defined as an airplane can be moved behind or in front of a cloud on the screen, simulating three dimensions.

Atari programmers accustomed to handling "players" and "missiles" as separate objects will have to adjust their thinking on the Commodore 64. The 64 has only sprites, which are used for both. Actually, the Atari works in a similar way, splitting its fifth player into four missiles to accompany the other four players. Programmers on the 64 will simply construct missiles out of whole sprites, since they have so many on hand anyway.

Sprites on the 64 can be defined up to 21 pixels high by 24 pixels wide, and with up to three colors each. In addition, a single statement will double a sprite in either direction, or both. Best of all, movement is extremely simple and fast: specify an X,Y position on the screen and the sprite is there.

On microcomputers which have them, sprites are proving to be the most powerful – yet underused – graphics feature available. Games ordinarily requiring 48K RAM or more to handle complicated animation can be done in 16K on machines with sprites. The Commodore 64's approach to sprites appears to be the most powerful to date. We can expect some dazzling arcade-style games as programmers get the hang of working with the 64.

The Sound Of Music

No doubt about it: the new Commodores with the SID chip have the most sophisticated sound capabilities of any home/personal computers on the market. Skeptical? You won't be after you hear them. For one thing, the SID chip is much more than the tone generators found in other computers. It is a true sound synthesizer with an envelope generator for each of its three voices, programmable attack, decay, sustain, and release for each voice, plus a choice of four waveforms, plus programmable high-, low-, band-, and notch-pass filters, plus 16-bit frequency resolution over a nine-octave range from 0-4 KHz, and even variable resonance and a master volume control.

Commodore sales literature does not exaggerate: the 64 truly rivals the capabilities until now found only in dedicated keyboard synthesizers.

The four waveforms allow users to vary the tone of each voice. For example, the variable-pulse waveform produces a sharp, biting sound like the tone generators in other computers. The triangle waveform is much more mellow, simulating organ music. The sawtooth waveform is abrupt, like a harpsichord. And the fourth waveform, "white noise," is handy for sound effects such as explosions.

The programmable attack-decay-sustain-release synthesizes the acoustical properties of notes made by ordinary musical instruments. The "attack" is the rapidity with which the note reaches its peak; a sharp, biting note is represented by a steep attack slope. "Decay" is the slope of the note's decline. "Sustain" is the note's duration. "Release" is the dying of the note to silence. The SID chip allows notes to be sustained up to 24 seconds.

Further processing of the note is possible with the programmable filters. These are often found as slide controls on keyboard synthesizers, and are available through BASIC on the Commodore 64. The 16-bit resolution means notes can be extremely fine-tuned – in increments of 0.059 KHz, in fact.

All these features might seem to make sound on the 64 difficult to program, but a Commodore representative managed to transform a row of keys into a primitive organ with less than a screenful of BASIC.

Not only does the 64 have an audio output port to feed all those fantastic sounds through a stereo system, but one Commodore programmer says it is even possible to feed outside sources *into* the computer. Can you imagine plugging and electric guitar into the 64, processing the sound through the SID chip, and routing it back to the amplifier? The neighbors will love you!

Odds And Ends

The Commodore 64 offers other niceties, too: syntax errors are detected upon entry, before RUN. There's a video monitor output. An assembler for the 6510 CPU is promised soon. The plug-in cartridges allow up to 16K additional ROM and 2K RAM. There are ports for two Atari-type joysticks or four paddles. And there will be a forthcoming version of LOGO, the popular educational language for children.

Next month, Part II will take an even more detailed look at the Commodore 64's advanced features. ©

Drawing on his background as a professional programmer for Interact and Atari, the author offers a solid case for the "other half" of programming – debugging.

Don't Forget Testing

Dave Johnson
Mountain View, CA

Testing is an often overlooked phase of the software development process. But many large software developers have learned to devote as much time and effort to testing as they do to programming. If you plan to sell your programs or to give them to other computer users, you should be fairly confident that the programs work. Not very many of us have given much thought to what we can do to give us this confidence. This article will offer suggestions and methods to help you gain confidence in the reliability of your programs.

The first thing to look at, of course, is the way that you write your programs. Programs that are written in a structured, logically organized manner tend to have fewer problems, to work better in less memory, and to be easier to fix when problems are found. But all of us reading this article use faultless programming methods, so I won't pursue this.

What Should The Program Do?

To start the testing process you need to define exactly what the program must do and not do before you will give it your stamp of approval. This is more complicated than it seems. To demonstrate how complicated, let's look at the questions that arise when trying to test a very simple BASIC program. This hypothetical program will just accept your name as input and display it on the screen.

Obviously, the first visible function of our program is to get your name. (I say "visible" because the program may spend some time dimensioning arrays, etc., internally. The first visible function is to get the name.) In my opinion, if you run any program, but all that is displayed is a question mark, that is a problem. "Human engineering" is very important in programs that you intend to sell. The program should make it clear to the user what it wants the user to do. Our little program should prompt you by displaying a message such as "Please type in your name." To summarize, one question to ask when reviewing a program is "Does the program prompt for input correctly and clearly?"

Now you must type in your name. How many letters can you type that the computer accepts? Are strings big enough for all possibilities? Will the computer let you know if this limit is exceeded? Do the instructions supplied with the program tell you what this limit is? (Naturally you included instructions with your program!) Are control-C's or other keys that cause a program to abort masked out? Is it clear what the allowable range of inputs is? Does the program catch input errors and gracefully recover from them? (And tell you what the error was?)

Now let's look at the display of the name. Does the program display the name on the next line, or does it give breathing room of a couple of lines so that you are not confused by a crowded screen? Does the program display the correct name? Are displays correct, clear and easily understood?

Once the program has run through once, does it simply return to BASIC, ask for another name, or allow options?

This example was intended to give you an idea of the kinds of questions that must be answered when deciding whether or not to approve a program. When we talk about a "bug," we can be talking about anything from a problem that erases everything on your disk to a misspelled word in your instruction manual. If you are putting together a simple program that will be used a few times, you will have different requirements than you will if you have a business reputation to maintain. You see that you have to decide just what you will call a problem that you will spend time fixing, and what you will let go by.

Finding The Bugs

Once you have decided what to say are problems, how do you conduct a test that finds them? The most useful thing I can say, to start with, is that you should WANT to find bugs! If you write a program that runs the first time and doesn't look like it has bugs in it, that does *not* mean you are a good programmer. It only means that you have not looked for the bugs yet! Too often programmers take it personally if bugs are found in their codes and subconsciously design tests that will not find bugs. Not finding bugs doesn't mean that there are no bugs; it means they have not been found!

There are two types of testing to consider. Both types should be used on your product. The first is testing that you set up and conduct. The second is testing that some independent, objective person conducts without interference from you.

DOS FOR AIM-65®: $499.00 COMPLETE!

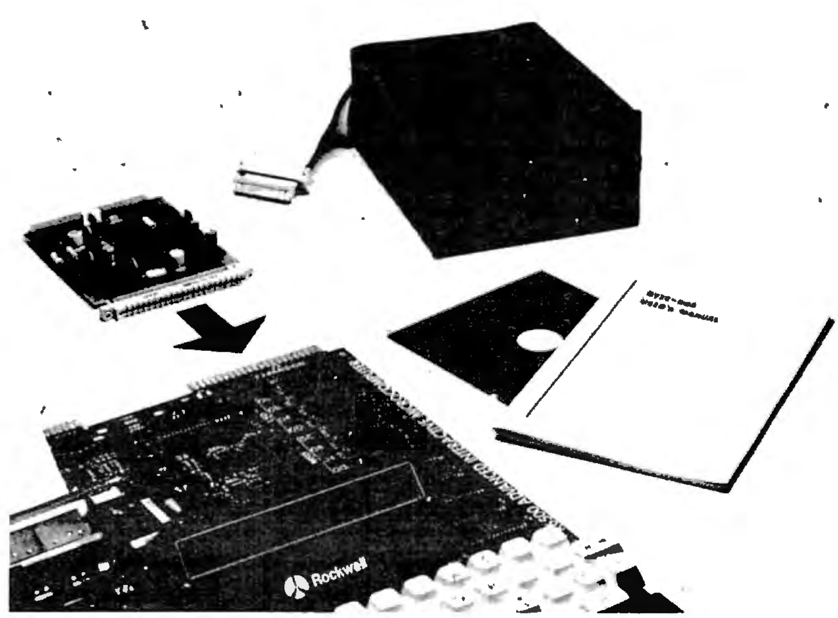
INSTANT EASY ACCESS TO YOUR AIM-65 PROGRAMS!
MANIPULATE OBJECT FILES AT THE TOUCH OF A KEY.
NEW BYTE-DOS DISC OPERATING SYSTEM.
153K BYTES ON-LINE. INEXPENSIVELY!

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!



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!*

master charge VISA

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.

You have the advantage of knowing the code. This advantage, by the way, is called "white box" logic coverage testing. You know the logic of, for example, a FOR 1 TO 10 loop. You can look at what happens the first time through, one time in the middle, and the tenth time. (Both extremes, and the middle.) Try to cover all the "paths" through your program, as well as conditions outside of the expected. For example, for a subroutine that expects a variable IVAR to be between 1 and 10, test by passing 1, 5, 10, 0, -1, -99, 11, and 99 in IVAR. (Extremes, middle, and outside.)

"Black box" testing methods are designed by someone not familiar with the code. The program is approached as an unknown (a black box), and the tester knows only what the program functionally should and should not do. The tester is not concerned with how the computer does what it does, only with whether it does or not. As the programmer, you may be so involved with a tricky problem in a small segment of code that you overlook a major design consideration.

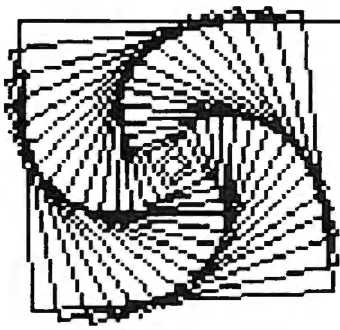
To set up such a test, sit down with the functional definition that was put together earlier and come up with "test cases." Test cases are inputs that will take the program through a variety of possible paths. I have seen very detailed test cases that cover hundreds of pages for relatively simple programs.

There should be a procedure for the tester to verify completing each step of the test case and to comment on the results. This way you will know what led up to any problems you might encounter. Duplicating problems can be the hardest part of the debugging process.

For our demonstration BASIC program that inputs and displays a name, the test cases might look like this:

Input	Expect to see	Comments
BILL	BILL	I saw "BILLL"
Bill	Bill	Still saw "BILLL"
BILLY BOB	BILLY BOB	Still saw "BILLL"
Bob	Bob	Now it says "Bobb"

You should develop a structured, organized testing plan for each testing method that can be generally applied to different programs. You can refine such a plan as you find areas that are lacking. Unfortunately, finding these areas means getting letters from customers saying: "When I type CONTROL SHIFT W followed by a CONTROL BACKSPACE the second time I run the program, the screen turns orange. But I can't make this happen the first or third time." Your *next* program will be tested for this! ©



A Monthly Column

Friends Of The Turtle

David D. Thornburg
Associate Editor

Dear Turtle ...

Dear Turtle,

I have an Apple II and want to buy LOGO for it. Which is the best version to get? My sister in Azuza (who has been chasing my husband since day one) likes the Terrapin version, and my husband (who would probably run off with the bank teller if I didn't have the goods on his father's secretary) favors Krell. I am leaning towards the LCSI version sold by Apple. Should I trust my heart?

Perplexed in Pensacola

Dear Perplexed,

As soon as I add Krell LOGO to my collection, I will write a report on all three. No matter which you choose, you will find LOGO to be a marvelous language. Your sister could be more of a problem if she knew how similar the Krell and Terrapin LOGO's are. I think it's time the three of you shared procedures

Dear Turtle,

Why is it that FOLLLK is now FOLLK (Friends of Lisp/LOGO and Kids) and is now located at 436 Arbalo Dr., San Francisco, CA 94132?

Just Asking

Dear Just Asking,

Beats me. It could be that they fellt that FOLLLK had tooooo many LLLLLLL's in it. Remember that a one-L lama is a priest, a two-L llama is an animal and that a three-L "lama" is a big fire.

Dear Turtle,

My brother-in-law runs a newsstand. Ever since he started selling **COMPUTE!** he has been too busy to see his wife (so he says). Since he is devoting his life to his work, what are the chances he will also be able to sell an all-LOGO magazine soon?

St. Louis Blue

Dear Saint,

My experts tell me that both the Krell people and the FOLLK folk will be publishing LOGO newsletters soon. If your brother-in-law also sold **COMPUTE! Books**, *he could build an addition to his newsstand to house your sister.*

Dear Turtle,

How come TI didn't go down the drain like Thornburg predicted when the 99/4 first came out?

Disillusioned

Dear Disillusioned,

TI's remarkable turnaround can be linked to the three L's: Low prices, Lots of product, and LOGO. TI is clearly in this business to stay. As for Thornburg ...

Dear Turtle,

Why haven't you told PET users about Spider by Bill Finzer? This "turtle" program is available from both the Softswap Microcomputer Center, San Mateo County Office of Education, 333 Main St., Redwood City, CA 94063, or from San Francisco State University, Center for Mathematical Literacy, 1600 Holloway Ave., San Francisco, CA 94132.

Tulare Fan

Dear Tulare Fan,

I just did.

Dear Turtle,

Talk about dumb. My local computer store salespeople don't know what a Big Trak is.

Amused

Dear Amused,

Tell them to drop into their local toy store to see the Big Trak robot vehicle by Milton Bradley. It is the best (and only) $40 programmable LOGO-like turtle on the market.

Dear Turtle,

I think Turtle Geometry is awful. It lets *anyone* quickly create pictures on a display with a minimum of training. People should have to *work* for this privilege. As for myself, I think people should use coordinate pairs and have to specify both the *x* and *y* coordinates for each point on the screen. Enough of this nonsense! I think, therefore I am most sincerely yours,

René Descartes

Dear René,

You might want to become a philosopher or something, and leave graphics to the rest of us.

Dear Turtle,

While you were answering those letters, I was trying out an Atari PILOT program written by Dr. R. Bharath, an associate professor at Northern Michigan University. His program simulates the "Drunkard's Walk" problem in which one guesses the number of steps needed to reach a target, given a certain probability of moving toward or away from the target. He hopes that **COMPUTE!** readers find it interesting.

Dr. T.

Dear Dr. T.

Dr. Bharath's program is presented here.

Confidential to Lost in Space,

As Atari PILOT programmer Richard Kline says,

```
GR: PEN RED

GR: 28(DRAW 6; TURN 13)

GR: PEN BLUE

GR: GOTO 12,-19

GR: TURN 14; 5(DRAW 50; TURN 144)
```

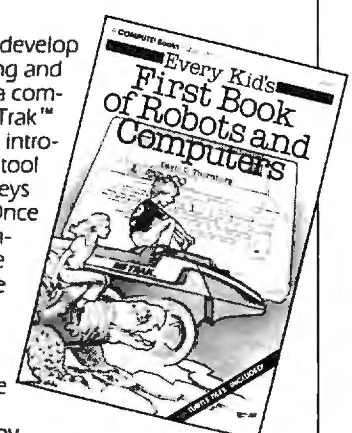
```
10 T: DRUNKARDS WALK BY R. BHARATH
20 T:
30 T:A DRUNKARD COMES OUT OF PUB WHICH IS\

40 T:MIDWAY BETWEEN HOME AND A POND <TEN S
   TEPS FROM EACH>.  \
50 T:AT EACH STEP CHANCES ARE THE SAME THA
   T THE NEXT STEP WILL BE TOWARDS HO
   ME.\
60 T:YOU HAVE TO GIVE THIS PROBABILITY\
70 T:AND THE PROGRAM CALCULATES WHERE THE
   PERSON ENDS UP AND \
80 T:HOW MANY STEPS IT WILL TAKE TO REACH
   THE DESTINATION <HOME OR POND>.
90 T:
100 T:
110 T:
120 T:WHAT ARE THE CHANCES THE NEXT STEP WI
    LL BE TOWARDS HOME?
130 T:<GIVE YOUR RESPONSE ON SCALE 0 TO 10
140 T:10 MEANS CERTAIN TO TAKE EACH STEP TO
    WARDS HOME
150 T:0 MEANS CERTAIN TO TAKE EACH STEP TOW
    ARDS POND >
160 A: #P
170 T:GUESS IF HE WILL REACH HOME<SAY YES O
    R NO>
180 A:$GUESS
190 T:GUESS HOW MANY STEPS HE WILL TAKE BEF
    ORE GETTING TO HIS DESTINATION <HO
    ME\
195 T:OR POND>
200 A:#G
210 GR:TURN 90
220 C:#D=0
230 C:#C=0
240 *BBB
250 GR:PEN YELLOW
260 GR:GOTO -50,0
270 GR:GOTO 50,0
280 GR: PEN RED
290 GR: GOTO #D,0
300 C: #X=?\10
310 C:#Y=1-<2*<#X<#P>>
320 T: STEP NUMBER #C
330 PA:50
340 GR:GO 5*#Y
350 C:#D=#D+<5*#Y>
360 C:#C=#C+1
370 J<#D=-50>: *EEE
380 J<#D=50>: *EEE
390 PA :20
400 GR:CLEAR
410 J: *BBB
420 *EEE
430 PA:100
440 GR:QUIT
450 J<#D=50>:*F
460 T:   REACHES HOME
470 A:=$GUESS
480 M:YES
490 TY:YOU WERE RIGHT
500 TN:YOU GOT IT WRONG
510 J:*X
520 *F T:FALLS IN POND
530 A:=$GUESS
540 M:NO
550 TY:YOU WERE RIGHT
560 TN:YOU WERE WRONG
570 *X
580 T:TAKES #C STEPS IN PROCESS
590 T:<WITH CHANCE #P IN 10 OF TAKING A STE
    P TOWARDS HOME AT EACH STEP>
600 T:
610 T:
620 T:
630 T:<YOUR GUESS WAS #G STEPS>
640 E:
```

©

An Intriguing New Release from **COMPUTE! Books:**

Every Kid's First Book Of Robots And Computers

By David Thornburg



From the author's preface:

"This book allows children to develop skills in computer programming and geometry through the use of a commonly available toy – the Big Trak™ robot vehicle. Programming is introduced as the communication tool through which the child conveys instructions to the machine. Once the machine's language limitations are understood, it can be made to follow any procedure which has been entered by the user.

"Our use of turtle commands as the programming language mirrors the process-based descriptions commonly used by children. For example, a child is likely to describe a nearby location, such as a friend's house, by a procedure (Go two blocks, turn right, go another block, turn left,...). Because turtle geometry has been incorporated as the graphics environment in several computer languages available for the popular desk-top computers, these programming ideas can continue to be used as the child learns to operate other computers."

In *Every Kid's First Book Of Robots And Computers*, author David Thornburg conveys a uniquely exciting learning experience for children, parents, and teachers. The book uses Big Trak, PILOT/LOGO type languages, and Turtle Tiles™ to explore the concepts and techniques of robot/computer programming. Turtle Tiles, included with every book, are designed to provide hands-on programming experience to children without access to a Big Trak or a personal computer. Additionally, the Tiles can be used in conjunction with either of these items to share and reinforce the exercises in the book.

Ask for *Every Kid's First Book Of Robots And Computers* at your computer retailer, local bookstore, or order directly from:

COMPUTE! Books
P.O. Box 5406
Greensboro, NC 27403

For Fastest Service, Call Toll Free 800-334-0868
In NC 919-275-9809

$4.95 plus $1.00 shipping and handling.

ISBN 0-942386-05-1. Perfect bound, 96 pages plus Turtle Tiles™. Fully illustrated.

Dealer and educator quantity discounts are available.

Big Trak is a trademark of the Milton Bradley Company.
Turtle Tiles are a trademark of David D. Thornburg and Innovision, Inc.

LOGO

© M.I.T.

Krell's LOGO for APPLE II*

Includes: our new low price and

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

College Board SAT* Prep Series

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

S.A.T., P.S.A.T., N.M.S.Q.T. — Educator Edition set includes 25 programs covering Vocabulary, Word Relationships, Reading comprehension, Sentence Completion, and Mathematics. $229.95

INDEPENDENT TESTS OF S.A.T. SERIES PERFORMANCE SHOW A MEAN TOTAL INCREASE OF 70 POINTS IN STUDENTS' SCORES.

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

Micro-Deutsch

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

Isaac Newton + F.G. Newton

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

Krell Software Corp.

"The State of the Art in Educational Computing"

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

Krell Software Corp has no official ties with the College Entrance Examination Board or the Educational Testing Service Krell is, however, a supplier of products to the E.T.S. N.Y.S. residents add sales tax.

PROGRAMS AVAILABLE FOR THE TRS80, APPLE II, PET & ATARI.

*Trademarks of Apple Comp Corp., Tandy Corp., Commodore Corp., Atari Corp.

Sometimes those beautiful Atari graphics can take a long time to create. This program eases your task by adding to the graphics commands already in BASIC.

Atari Sketchpad

Todd Mowbray
Burlington, Ontario

This program is dedicated to those Atari owners who use the graphics modes on their machines frequently for pictures and graphics displays. This program replaces the need to type out commands such as DRAWTO and PLOT, and it also can correct mistakes. It uses the Graphics modes with a text window (modes 3-8) and can be run on 8K. However, on the 8K machines the high-resolution graphics are unattainable, due to the memory usage for the display. There are seven commands to be used within the program: Plot, Drawto, Erase, Change Modes, Clear Screen, Print Out and Quit.

All of the commands use short forms such as "p" for "plot." The commands and their short forms are reviewed below.

Plot: Short form "p". This is the same as the BASIC PLOT, and as long as the values are within the specified ranges (see below) they will be plotted.

Drawto: Short form "d". This is the same as the BASIC DRAWTO, and the same is true about the values.

Erase: Short form "e". This is a built-in command that is well used once you get a picture or graphics display nearly done. It erases the last line, or PLOT command, that was entered.

Change Modes: Short form "cm". This command display changes the Graphics modes for you, to any desired mode (3-8).

Clear Screen: Short form "cs". This one is self-explanatory.

Print Out: Short form "po"· This is another

Table 1. Maximum values for Graphics modes 3 to 8.

Graphics Mode	Horizontal	Vertical
3	40	20
4	80	40
5	80	40
6	160	80
7	160	80
8	320	160

Of course, when you use these values, remember that the upper-left hand corner is 0,0. For Graphics mode 8, the x axis or horizontal maximum number is 319. This is because the Atari starts at zero instead of one for addressing to the screen.

Table 2: Variables

Variables

A$,A,B,X(I),Y(I),AA,BB,T

A$ – holds the commands that were entered (p and d)

A,B – present location of "pencil" on screen

X(I), Y(I) – old values of A and B

AA,BB – maximum values for the Graphics mode

T – a counter for the number of commands (p and d), and also helps to keep track of old commands

50 – dimensions A$,C$,X(I),Y(I), and T

100-105 – Title

115-120 – set up Graphics mode

121-123 – read in values for AA and BB

125 – sets Graphics mode

150-195 – wait for command, and if proper one is used, branch to the appropriate subroutine.

500-525 – plotting subroutine. Check values and at line 515 add command to A$, and values to X(I) and Y(I).

750-775 – DRAWTO subroutine. Check values, and, if correct, add the command to A$ and values to X(I) and Y(I).

1000-1015 – erase subroutine; erase last command, take it from A$, and take the values away from X(I) and Y(I).

1025-1080 – print out subroutine; prints out the commands (p and d) and their corresponding values. Will print out 20 of them and then ask for a number. This is only designed as a delay, so that if you are writing them down, you will have time.

5000 – a delay subroutine.

And that's all there is to it. The program will run as is, or it could be changed to have a Change Color command. Happy sketching!

function that was one of the purposes in creating the Atari Sketchpad – to keep track of the PLOT and DRAWTO commands that were entered, and the values that were with them. This is usually done after the masterpiece is created. The computer will print out the command as either a "d" or a "p" corresponding to PLOT and DRAWTO. Also, when you erase something, the values and the command are also erased, so that you get a printout of the steps it took to generate the picture; you don't get false values.

Quit: Short form "q". This is also self-explanatory.

```
50 DIM A$(250),C$(2),X(250),Y(250):A$="":T=1
100 GRAPHICS 1:SETCOLOR 2,0,0:SETCOLOR 0,4,4
    :? #6;"ATARI SKETCHPAD 1.1"
101 POSITION 0,4:? #6;"BY TODD MOWBRAY"
105 GOSUB 5000
110 REM
115 ? "{CLEAR}ENTER GRAPHICS MODE (3-8)";:IN
    PUT M
120 IF M<3 OR M>8 THEN ? "{UP}":GOTO 115
121 FOR I=1 TO M-2:READ AA,BB:NEXT I
122 DATA 39,19,79,39,79,39,159,79,159,79,319
    ,159
123 RESTORE
125 GRAPHICS M:SETCOLOR 2,0,0:COLOR 1
150 ? "{CLEAR}COMMAND";:INPUT C$
155 IF C$="" THEN 150
160 IF C$="P" THEN 500
165 IF C$="D" THEN 750
170 IF C$="E" THEN 1000
175 IF C$="CS" THEN A$="":T=1:GOTO 125
180 IF C$="CM" THEN A$="":T=1:GOTO 110
185 IF C$="QT" THEN END
190 IF C$="PO" THEN 1025
195 GOTO 150
500 ? "{CLEAR}PLOT WHERE (X,Y)";:INPUT A,B
505 IF A<0 OR B<0 THEN 500
510 IF A>AA OR B>BB THEN ? "ILLEGAL COORDINA
    TES":GOSUB 5000:GOTO 500
515 PLOT A,B:X(T)=A:Y(T)=B:A$(LEN(A$)+1)="P"
    :T=T+1
520 IF T>249 THEN ? "{CLEAR}ALLOWED LIMIT":G
    OTO 1025
525 GOTO 150
750 ? "{CLEAR}DRAWTO WHERE (X,Y)";:INPUT A,B
755 IF A<0 OR B<0 THEN 750
760 IF A>AA OR B>BB THEN ? "ILLEGAL COORDINA
    TES":GOSUB 5000:GOTO 750
765 DRAWTO A,B:A$(LEN(A$)+1)="D":X(T)=A:Y(T)
    =B:T=T+1
770 IF T>199 THEN ? "ALLOWED LIMIT":GOTO 102
    5
775 GOTO 150
1000 REM ERASE
1005 IF T<2 THEN 150
1010 COLOR 0:PLOT A,B:DRAWTO X(T-2),Y(T-2):P
     LOT A,B:DRAWTO X(T-2),Y(T-2):COLOR 1
1015 A$(LEN(A$))=" ":T=T-1:X(T)=0:Y(T)=0:GOT
     O 150
1025 GRAPHICS 0:SETCOLOR 2,0,0:? "P=PLOT  D=
     DRAWTO":? :? :? :FOR I=1 TO T-1
1050 ? A$(I,I);"(";X(I);",";Y(I);")":NEXT I
1080 ? "WANT TO SKETCH ANOTHER PICTURE";:INP
     UT C$:IF C$="N" THEN END
1090 GOTO 125
5000 FOR I=1 TO 1500:NEXT I:RETURN
```

©

For the Apple II, this simulates chemistry experiments for elementary students. It includes a list of choices and animated graphics.

Chemistry Lab

Joanne Davis
Kew Gardens, NY

"Chemistry Lab" encourages elementary school students to hypothesize and review concepts by allowing them to duplicate laboratory experiences in chemistry. It uses standard chemical indicators to identify a variety of substances as acids, bases, sugars, or starches.

The program is menu-driven. After choosing a topic, the student is shown instructions, followed by a picture of an eyedropper containing the indicator (in the appropriate color), a beaker (containing the material to be tested), and the material and indicator names. The student predicts the result of the test, as he or she would before conducting a laboratory experiment, and INPUTs the prediction.

When the test is carried out, the eyedropper releases its contents drop-by-drop and the beaker fills with liquid, its color indicating the presence of acid, sugar, etc. Comments then reinforce the material's classification.

This procedure is repeated to test four more substances. More items can easily be added by DIMensioning the arrays and adding more DATA.

Two Special Techniques

Two of the techniques used in this program should be of special interest. The animation is created by alternating between color and black and by time delays caused by empty FOR/NEXT loops. The inside of the dropper is blacked out a line at a time, with the delay making the action visible. The previous position of the drop is blacked out and then redrawn at a new location. Then the beaker is filled up, a line at a time.

Since the sugar and starch test required virtually the same instructions, a way had to be found to make the few alterations needed. The changes are READ in from DATA statements and inserted into the message.

A science curriculum can be made to come alive with animated laboratory experiments. Try it and see.

Table 1. VARIABLES

A$: GET response
AN$: Answer INPUT
B$: Sugar or starch?
BEAK: Color in beaker
C$: Color name of indicator
CH$: Menu choice
DROP: Color in dropper
ID$, IH$, IR$: Correct identity
IN$: Name of indicator
N$: Acid test materials
N1$: Sugar/starch test materials
P: Drop position
S,SO: Sound
TT: Time delay
X: Counter
Y: DATA flag

Table 2. SUBROUTINES

1000 Acid test
2000 Starch test variables
2500 Sugar test variables
3000 Sugar/starch instructions and tests
4000 READ data into array
5000 Graphics outline
6000, 6500 Animation

```
0 REM BY J. DAVIS
10 REM CHEM EXPER
20 TEXT : HOME
30 VTAB 8: HTAB 15: PRINT "***************"
40 HTAB 15:PRINT"WELCOME TO THE":HTAB 15:P
    RINT"CHEMISTRY LAB":HTAB 15:PRINT"
    ***************"
50 FOR TT = 1 TO 4500: NEXT
60 HOME : VTAB 5: PRINT "CHOOSE TEST 1, 2,
    OR 3:": PRINT : HTAB 5: PRINT "1.
    ACID":HTAB
65 HTAB 5: PRINT "4. QUIT"
70 GET CH$:CH = VAL (CH$): ON CH GOSUB 100
    0,2000,2500,100
80 TEXT : GOTO 60
100 END
1000 REM ACID/BASE***PHENOL
1020 REM INSTRUC
1025 GOSUB 4000
1030 TEXT : HOME
1040 PRINT:PRINT:PRINT "YOU ARE GOING TO TES
    T SOME MATERIALS TO": PRINT "SEE I
    F THEY ARE ACIDS
1041 OR BASES. THE": ? "INDICATOR WILL TURN ~
    ";: INVERSE : ? "PINK";: NORMAL : ?
```

```
1042 " IN AN ";: INVERSE : PRINT "ACID";: NO
    RMAL : PRINT "."
1045 PRINT : PRINT "TYPE ";: INVERSE : PRINT
     "A";:NORMAL:PRINT"IF YOU THINK TH
    AT THE MATERIAL":
1046 PRINT "IS AN ACIS.": PRINT "TYPE ";: IN
    VERSE : PRINT "B";: NORMAL : PRINT
     "IF YOU THINK
1047 THAT THE MATERIAL": ? "IS A BASE."
1050 PRINT : PRINT "HIT ANY KEY TO BEGIN.":
    GET A$
1070 HOME
1150 FOR X = 1 TO 5
1152 DROP = 15: GOSUB 5000
1155 VTAB 21
1160 PRINT "INDICATOR: ";: INVERSE : PRINT "
    PHENOLPHTHALEIN"; NORMAL
1180 PRINT : INVERSE : PRINT "A";: NORMAL :
    PRINT "CID OR ";: INVERSE : PRINT
    "B";:
1181 NORMAL:PRINT "ASE ?":GET AN$: IF AN$ <
    > "A" AND AN$ < >"B" THEN HOME:GOT
    O1155
1190 IF ID$(X) = "A" THEN BEAK = 11: GOTO 12
    10
1200 BEAK = DROP
1210 GOSUB 6000
1220 HOME : IF ID$(X0 = "A" THEN PRINT N$(X)
    ;" IS AN ACID.": GOTO 1240
1230 PRINT N$(X);" IS A BASE."
1240 FOR TT = 1 TO 4000: NEXT TT
1250 NEXT X
1400 FOR TT = 1 TO 1000: NEXT TT
1500 RETURN
2000 Y = 1: GOSUB 4000: GOSUB 3000: RETURN
2500 Y = 2: GOSUB 4000: GOSUB 3000: RETURN
3000 REM STARCH/SUGAR INSTRUCTIONS
3010 TEXT : HOME
3020 PRINT : PRINT : PRINT "YOU ARE GOING TO
     TEST SOME MATERIALS TO": PRINT "S
    EE IF THEY
3021 CONTAIN ":B$(Y);".": ? : ? "THE INDICAT
    OR WILL TURN ";: INVERSE :?C$(Y);:

3022 NORMAL : PRINT " IN A ": INVERSE : PRIN
    T B$(Y);: NORMAL : PRINT "."
3030 PRINT:PRINT "TYPE ";: INVERSE :PRINT "Y
    ";: NORMAL:PRINT " IF YOU THINK TH
    AT THE MATERIAL":
3031 PRINT "CONTAINS ";B$(Y);" ."
3040 PRINT : PRINT "HIT ANY KEY TO BEGIN.":
    GET A$
3050 HOME
3060 FOR X = 1 TO 5
3070 DROP = P(Y): GOSUB 5000
3080 VTAB 21
3090 PRINT "INDICATOR: ";: INVERSE : PRINT I
    N$(Y): NORMAL
3092 :
3094 :
3096 :
3100 PRINT "NOW TESTING: ";: INVERSE : PRINT
     N1$(X): NORMAL
3110 PRINT : INVERSE : PRINT B$(Y);: NORMAL
    : PRINT " (Y/N) ?": GET AN$
3120 IF AN$ < > "Y" AND AN$ < > "N" THEN HOM
    E : GOTO 3080
3130 ON Y GOSUB 3500,3600
3140 FOR TT = 1 TO 4000: NEXT TT
3150 NEXT X
3200 RETURN
3500 REM STARCH MESSAGE
3510 IF IH$(X) = "S" THEN BEAK = 3: GOTO 353
    0
3520 BEAK = DROP
3530 GOSUB 6000
3535 HOME
3540 IF IH$(X) = "S" THEN PRINT N1$(X);" CON
    TAINS STARCH.": GOTO 3560
3550 PRINT N1$(X);" DOES NOT CONTAIN STARCH.
    "
3560 RETURN
3600 REM SUGAR MESSAGE
3610 IF IR$(X) = "S" THEN BEAK = 9: GOTO 362
    5
3620 BEAK = DROP
3625 GOSUB 6000
3627 HOME
3630 IF IR$(X) = "S" THEN PRINT N1$(X);" CON
    TAINS SUGAR.": GOTO 3660
3650 PRINT N1$(X);" DOES NOT CONTAIN SUGAR."

3660 RETURN
4000 RESTORE
4005 FOR X = 1 TO 5
4010 READ N$(X),ID$(X),N1$(X),IH$(X),IR$(X)
4020 NEXT X
4030 FOR X = 1 TO 2
4040 READ B$(X),C$(X),IN$(X),P(X)
4050 NEXT X
4060 RETURN
5000 REM SCREEN**OUTLINE BEAK AND DROP
5010 GR : COLOR= 10
5020 VLIN 0,20 AT 14: VLIN 0,20 AT 18
5030 HLIN 15,17 AT 0: HLIN 13,19 AT 6
5040 HLIN 15,17 AT 21: VLIN 21,24 AT 16
5050 PLOT 9,28: VLIN 28,38 AT 10
5060 VLIN 28,38 AT 21: HLIN 11,20 AT 38
5065 REM INSIDE DROPPER
5070 COLOR= DROP
5080 VLIN 15,20 AT 15: VLIN 15,20 AT 16: VLI
    N 15,20 AT 17
5500 RETURN
6000 REM ANIMATION
6010 COLOR= 0
6015 P = 31:S = - 16336
6020 FOR G = 15 TO 20
6025 PLOT 16,P
6030 HLIN 15,17 AT G:SO = PEEK <S> - PEEK <S
    > - PEEK <S>
6033 GOSUB 6500
6035 FOR TT = 1 TO 400; NEXT TT
6037 COLOR= 0
6040 NEXT G
6050 FOR TT = 1 TO 400; NEXT TT
6100 COLOR= BEAK
6110 FOR G = 37 TO 32 STEP - 1
6120 HLIN 11,20 AT G: FOR TT = 1 TO 250; NEX
    T
6130 NEXT G
6140 RETURN
6500 COLOR= DROP:P =P + 1: PLOT 16,P
6510 RETURN
7000 DATA SOAP,B,BREAD,S,S,LEMON JUICE,A,CRA
    CKER,S,0,COLA,A,CHOCOLATE,0,S,BAKI
    NG
7001 SODA,B,COLA,0,S,VINEGAR,A,FLOUR,S,0
7010 DATA STARCH,PURPLE,IODINE,13,SUGAR,ORAN
    GE,BENEDICTS SOLUTION,7
30000 REM BY J. DAVIS
```

©

A Monthly Column

The World Inside The Computer

Fred D'Ignazio is a computer enthusiast and author of several books on computers for young people. He is presently working on two major projects: he is writing a series of books on how to create graphics-and-sound adventure games. He is also working on a computer mystery-and-adventure series for young people.
As the father of two young children, Fred has become concerned with introducing the computer to children as a wonderful tool rather than as a forbidding electronic device. His column appears monthly in **COMPUTE!**

Build Your Own Computer Friend

Fred D'Ignazio
Associate Editor

Catie hopped down the steps and jumped out of the bright yellow school bus. She rushed home.

She burst into the kitchen, chattered to her father, munched a granola bar, then headed straight for the computer.

She turned the computer on.

A smiling cartoon face appeared on the screen. Under the face, in bright orange, appeared the words "HI! I'M GED! YOU TURNED ME ON! WHO'S OUT THERE?"

Catie searched the keyboard, then, pecking the keys one at a time, she typed "C-a-t-i-e" and pressed the RETURN button.

"CATIE, BABY!" typed the computer. "I WAS GETTING LONELY. HERE'S A KISS FOR TURNING ME ON. SSSMOOOOCH!"

Catie typed "k-i-s-s" and pressed RETURN.

"YUM. THAT WAS NICE," the computer said. "CATIE, IS TODAY A SCHOOL DAY?"

"Yes," typed Catie.

"IS SCHOOL OVER, CATIE?" asked the computer.

"Yes," Catie typed.

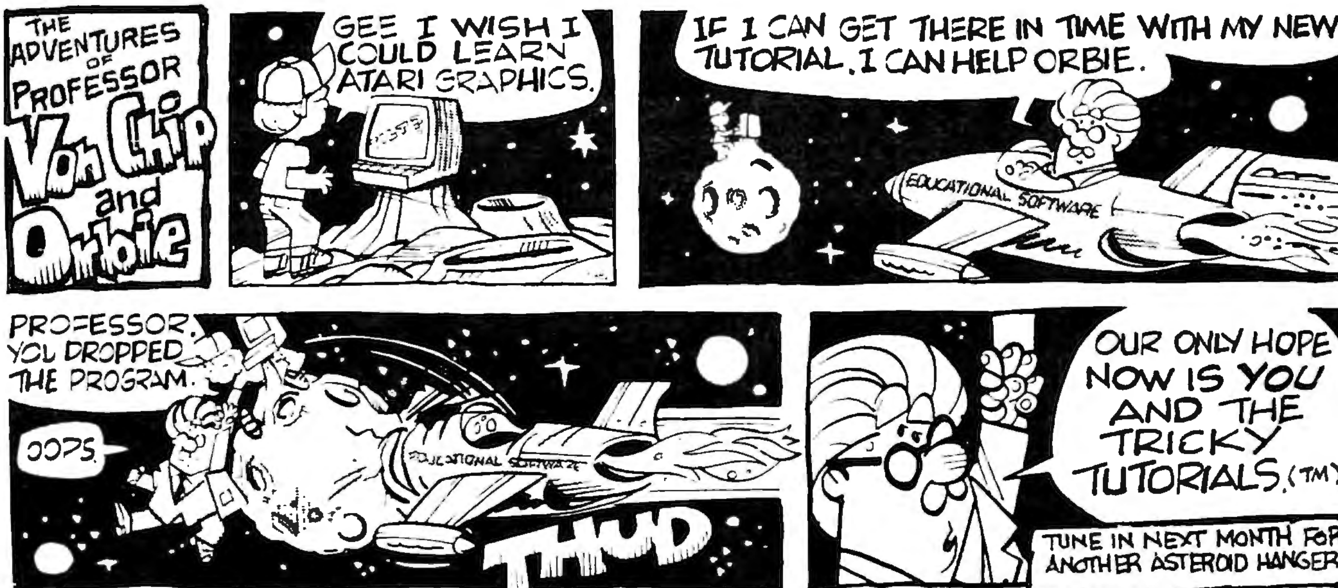
"WHAT DID YOU DO AT SCHOOL TODAY, CATIE?" the computer asked.

Catie told the computer about her field trip to the dairy farm, how she squeezed a cow's udder and milk squirted out, how she got her sneakers muddy in the cow barn, and how the farmer's wife served everyone fresh milk and chocolate chip cookies. "Now I'm home," she concluded.

"I'M GLAD YOU'RE HOME, CATIE. I MISSED YOU," the computer typed. "WANT TO PLAY?"

"Sure," typed Catie. "What game?"

"HOW ABOUT ONE OF THESE," said the computer. A list of Catie's favorite game programs appeared on the screen. Alongside each program was a number. There were word games, number games, storybook games, motor-skill games, and



TRICKY TUTORIALS (tm)

There are many things that the ATARI computers can do either better, or easier than other small computers. The following series of programs is designed for anyone who is at least familiar with BASIC programming. What each tutorial offers is similar to an extensive magazine article with all discussion in as simple language as possible, plus you get MANY examples already typed in and running. The instruction manuals range from 10 to 50 pages, and some tutorials fill up a complete tape or disk. There is little overlap in what is taught, so anyone wanting to know all they can should buy them all (my banker thanks you). ATARI buys these from us to use in training their own people! Rave reviews have been published in ANTIC, ANALOG, CREATIVE COMPUTING, and even INFOWORLD. You trust INFOWORLD, don't you?

TT #1: DISPLAY LISTS—This program teaches you how to alter the program in the ATARI that controls the format of the screen. Normally, 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 various Graphics modes on the screen at the same time. The program does all of the difficult things (like counting scan lines). You will quickly be able to use the subroutines included in your own programs.
16K Tape or 24K Disk **$19.95**

TT #2: HORIZONTAL/VERTICAL SCROLLING—The information you put on the screen, either GRAPHICS or TEXT, can be moved up, down, sideways, or diagonally. We provide the basic methods and leave the rest up to your skill and imagination. Includes 18 examples to get you started, with several using a small machine language subroutine for smoothness.
16K Tape or 24K Disk **$19.95**

TT #3: PAGE FLIPPING—Now you don't have to redraw the screen every time you change the picture or text. You will learn how to have the computer draw the next screen you want to see while you are still looking at the previous screen, 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 which graphics or text modes you are using, you can instantly look at up to 50 pages. The basic method takes only 9 lines and the usefulness is infinite.
16K Tape or 24K Disk **$19.95**

TT #4: BASICS OF ANIMATION—This program shows you how to animate simple shapes (with some sound) using the PRINT and PLOT commands, and it also has a nice little PLAYER/MISSILE GRAPHICS game you can learn from. The P/M example is explained and will get you started on this complicated subject (more fully explained in TT #5). This would be an excellent way to start making your programs come alive on the screen with movement! Recommended for beginning users.
16K Tape or 24K Disk **$19.95**

TT #5: PLAYER/MISSILE GRAPHICS—Learn to write your own games and other animated applications! The tutorial begins with many small examples that compliment the 50 page manual, then gradually builds up to a complete game where everything you need to know is fully explained. Also included are two machine language utilities that you can use to animate Players with from BASIC. Next we include two of the best editors currently available; one for editing playfield shapes (backgrounds), and one to edit your players, and all in glorious Technicolor!! Everything except the two editors run in 16K Tape or 32K Disk. **$29.95**

TT #6: SOUND AND MUSIC—Unless you have spent many years experimenting with the four voice channels, you will learn a lot from this one! Learn to play standard notes, chords, and whole songs using some simple "tricks". One of the nicest parts are the examples of special sound effects that you can refer to whenever you need a sound for a program or to impress a friend. This program will be of interest to all ages and levels of experience!
16K Tape or 24K Disk **$19.95**

SPECIAL DISCOUNT

Order the first six tutorials in a 3-ring binder for $99.95, a $30.00 savings!

TT #7: DOS UTILITIES—We at Educational Software have been shocked by some of the prices others are charging to offer you small utilities to help in the use of your Disk Drive. We now offer you all of the following plus explanation as to how each was written, and how to use them. A UNIQUE MENU PROGRAM, AN AUTORUN SYS BUILDER, DISK INSPECTOR (LOOK AT SECTORS), DISK JACKET PRINTER, AUTOMATIC FORMATTER, RECORD SAVE AND LOAD UTILITY.
32K Disk Only **$29.95**

MASTER MEMORY MAP (tm)

This book is the most valuable source of information for your ATARI you can buy. It starts out by explaining how to PEEK and POKE values into memory, so that even new computer owners can use many of these "Tricks". Then you are given 32 pages of the memory locations that are the most useful, along with hints on how to use many of the locations. Finally, it includes hints on problems you may be having with the computer and discusses the new Graphics modes 9 to 11. Even ATARI buys this book from us! **$6.95**

CONTACT YOUR LOCAL DEALER or ORDER BELOW

We have other fine programs for ATARI computers.
Write for a catalog.
Send us your programs to sell too!

USER SUBMITTED PROGRAMS

MINI-DATABASE/DIALER—stores and edits up to 8 lines of information such as names & addresses, phone numbers, messages, inventories, or anything you want. It has the usual sort, search, and print options, but it also has an unusual feature: If your file includes phone numbers and your phone company allows touch-tone phone signals, the program will DIAL THE PHONE NUMBER FOR YOU! 16K Tape or 24K Disk. **$24.95**

THE GRAPHICS MACHINE—allows the ATARI to act like more expensive graphics computers using simple commands like line, box, circle, polygon, fill, and savescreen to get a high resolution picture you can save on disk in only five seconds! Many more features! 48K Disk Only **$19.95**

BOB'S BUSINESS—14 small business type programs accessed from a common menu. 16K Tape or 32K disk **$14.95**

MINI WORDPROCESSOR—A simple text editor to write, save, and print several pages at a time. 32K Tape or Disk. **$19.95**

KID'S #1—Includes a MATH QUIZ, a children's TREASURE HUNT, and a DIALOGUE program. 16K Tape or 24K Disk 3 for **$14.95**

KID'S #2—SPELLING BEE, WORD SCRAMBLE, and TOUCH. 16K Tape or 24K Disk. 3 Educational games for . **$14.95**

PLAYER PIANO—Turns your keyboard into a mini-piano and more. 24K Tape or 32K Disk. **$14.95**

GRAPHIC SYMBOL LABELS—for your keyboard to remind you of the built-in Graphics symbols. 2 complete sets for **$2.95**

OUR NEWEST PROGRAMS

DOG DAZE—Two cute little doggies race for the fire hydrants, shoot their bones, and just have a lot of fun! A fast action program for all ages. 8K Tape or 16K Disk, in machine language **$16.95**

*** OUR BEST GAME ***

SPACE GAMES—Our family is being attacked by ALIENS, and only you can save us. A comic book manual will guide you through three games that test your ability in space skills. Includes ALIENS, SURVIVE, and ROBOT ATTACK, and is for all ages. The first two games require 16K for Tape. The last game and all Disk users need 32K. **$24.95**

MATHS FOR FUN—Another ENGLISH import teaching basic math skills. Very colorful and enjoyable to use. For ages 5 to 16. 16K Tape or 24K Disk **$19.95**

MARATHON—This is a unique math quiz for one or two players. You are in a race to move your runner across the screen first! There are four levels of play with five modes of operation for each. The game uses joysticks for all input, so play is easy for young children. This wonderful learning tool is imported from ENGLAND for your learning pleasure. Your kids will never even notice they are playing an EDUCATIONAL program. 16K Tape or 24K Disk. **$19.95**

GRADE BOOK—This nice record keeper will maintain a file of 35 students' test scores along with comments. Up to 7 subjects are allowed. 24K Tape or 32K Disk. A printer is optional **$24.95**

To order COD, VISA or MasterCard call — **(408) 476-4901**

By mail include $3.00 postage (USA only) or $1.50 Memory Map only—
California residents add 6.5% TAX.—Specify Tape or disk.

EDUCATIONAL SOFTWARE inc.

5425 JIGGER DR.
SOQUEL, CA 95073

adventure games.

Catie chose game number eight – a "junior" adventure game called *The Castle at the Bottom of the Sea*. She pressed the "8" button.

"GOOD IDEA!" the computer typed. "HAVE FUN! AND REMEMBER: DON'T TRY TO BREATHE UNDERWATER!"

The computer's face briefly appeared, winked, then vanished. Disk lights flashed. The screen turned a deep blue, and the game began.

The Build-A-Friend Books

Once upon a time, Catie had only one friendly computer in the D'Ignazio family – Ged. Now, thanks to the support of **COMPUTE!** and its publisher, Robert Lock, I am hard at work teaching all of Catie's computers to be more like people. Then I'm going to write a book about each one.

This fall, **COMPUTE!** will publish the books, each with the title: *Computer Friends: Learn About Computers by Inventing an Electronic Friend.* The books will teach kids how to create computer friends on the Apple, Atari, VIC, and TI computers. The first four friends they can create will be:

A SUPER HERO
A STARSHIP COMMANDER
A MONSTER
A JOKER/STORYTELLER

Each friend will come with a repertoire of abilities, responses, and memories. Each friend will know how to play at least one game with the child. Parents, teachers, and children can add more games, as they go along.

The friends are not totally formed until the child invents them. The child will get to choose many of their attributes and characteristics. Also, near the end of each book, I will indicate the way the child can continue working with his or her computer friend to make him or her (or it!) more complex, more exciting, and more like a real friend.

What Sort of Friend Would *You* Build?

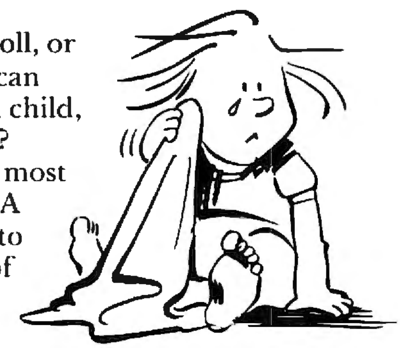
In this column and the next three columns (September, October, and November issues of **COMPUTE!**) I'd like to explore the topic of building a computer friend.

And I'd like very much to hear your views. You can reach me by writing:

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

An Imaginary Playmate

Psychologists say that it is healthy for a child to have an imaginary playmate. That playmate might be in the form of a ratty blanket, a favorite doll, or it might be completely invisible – except in the eyes of the child.



If a blanket, a doll, or an imaginary being can become a friend to a child, why not a computer?

On the surface, most computers are dull. A child can learn how to dredge up all sorts of exciting games and programs from the computer's memory. But, in between games, the computer becomes lifeless, cold, and not especially likable or friendly.

Having a "friend" program or operating system would let you replace the computer's heartless mechanical shell with a personality that is warm, sympathetic, and enjoyable for a child to interact with.

How do you get started? First, the child's computer friend must know certain things about the child. It must also know something about itself.

For example, the computer friend must have a name.

Jonathan Long, of Chapel Hill, North Carolina, sent me a "name" program that he wrote for his Atari computer. The program creates random four-letter names by alternating between consonants and vowels. Slightly modified, this is what Jonathan's program looks like:

```
4 DIM A$(1),TITLE$(4),CONSONANT$(20),VOWEL$(
  6),NAME$(20)
5 CONSONANT$="BCDFGHJKLMNPQRSTVWXZ"
6 VOWEL$="AEIOUY"
20 FOR OPER=1 TO 4
50 IF OPER=2 OR OPER=4 THEN LET LETTER=INT(R
   ND(0)*6)+1:NAME$=VOWEL$:GOTO 80
60 LET LETTER=INT(RND(0)*20)+1
70 NAME$=CONSONANT$
80 A$=NAME$(LETTER,LETTER)
90 TITLE$(LEN(TITLE$)+1)=A$
100 NEXT OPER
110 GRAPHICS 2:POKE 752,1:SETCOLOR 4,0,0:SET
    COLOR 2,0,0
114 ? "{CLEAR}"
115 POSITION 2,1
120 ? #6;"hello{A}"
121 POSITION 2,3
122 ? #6;"I AM A COMPUTER."
123 POSITION 2,5
125 ? #6;"MY NAME IS{3 ■}"
130 POSITION 2,7
135 ? #6;TITLE$
199 END
```

Jonathan's program could be inserted in a "computer friend" operating system or master program. It would make your computer be a friend with many names. Routinely, Jonathan's program spouts out names that sound ethnic, or normal, or alien, or just plain weird.

In addition to knowing its name, the computer friend should know something about itself. What

it "knows" can be either factual or completely made up.

Next, the computer friend should know your child's name. And the names of your child's best friends, his brothers or sisters, his pets, and the names of the child's *other* imaginary friends.

Third, the computer friend must know important details about the child's life. For example: the child's birthday, what grade (if any) he is in at school, who his teachers and babysitters are, what color hair he has, and so on.

Is the child a boy or girl? The computer friend should know.

What is the child's address? What's the name of the town?

What are his likes? What does he think is yucky?

What TV programs does he watch? What are his favorite books? Does he like ice cream cones? Pizza? Broccoli with ketchup?

All these things and more a good computer friend should know.

What Kind Of Friend Am I?

Friends come in all shapes and sizes. Some are silly. Some are serious. Some are smart. Others are not so smart. Some are aggressive, possessive, and bold. Others are soft, gentle, and sympathetic.

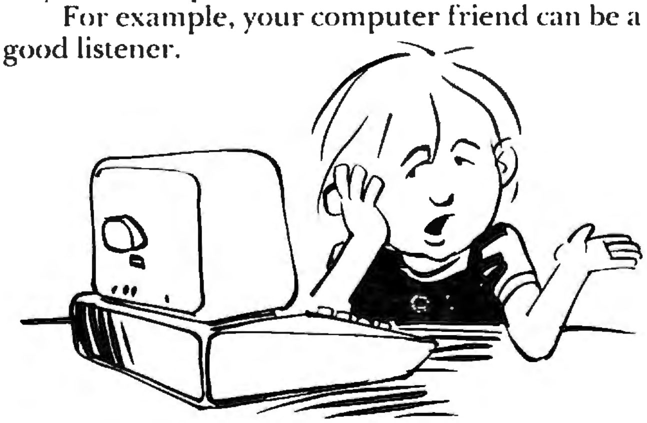
Some friends are all these things at different times.

If a computer tried to mimic human friends, it would have an identity crisis. Its program would become huge, unwieldy, and complex.

Our computer friends need not be super-intelligent. They need not be like human friends at all. Trying to act human is hard for computers.

Let your computer friend act like a computer. Let it do the things that are friendly, yet also are easy for a computer to do, not hard.

For example, your computer friend can be a good listener.



Think about all the times you were anxious to tell someone about something very important. And nobody listened.

Your human friends care about you. But often they are too wrapped up in their own problems and interests to listen to things that are important to you.

And if you talk too long, human friends may interrupt you, or become angry or bored.

But not a computer friend. You can talk to a (properly programmed) computer friend until you run out of things to say – or pull out its plug.

And it's not like pouring all your feelings, your ideas, your joys and your fears down into the Black Hole of Calcutta. (Which is what it's like when you talk to people sometimes.)

The computer *can* remember. You can teach it to remember everything (within its memory limitations). Or just bits and pieces. Or after codewords like "THIS IS VERY IMPORTANT!"

Or you can teach it to spit everything back at you. That way you can see what you sound like.

What other things can *you* think of that are among a computer's strengths and also help make it a good friend?

A Friend, Hero, Teacher, or Pet?

You can program your computer to have one personality – or several. It might be a shrewd sleuth like Sherlock Holmes. It might be brilliant – but absent-minded – like a nutty inventor. It might be stuffy or laid back, happy or moody, gossipy and inquisitive, or secretive and sly. It's up to you.

Or it might not be a friend at all. It might be something more. For example, the computer might be a hero, programmed to act like the child's favorite pro basketball star, or a comic book character, or a rock singer, or a beautiful actress or model.

Or the computer could be a teacher. Not just the types of computers we see today with teaching *programs*. The computer would have some aspects of a good teacher's personality. It would be warm and friendly. It would never lose its patience. It would guide your child toward various learning activities – programs – stored on the machine. It would be a friend with a single goal. And the goal is learning.

Have you ever heard a computer bark? Or meow?

It could, you know. If it were your child's pet.

Computer pets could be a godsend! Especially in cramped, tiny apartments in big cities. Especially where real, live animal pets are not allowed.

Just think: you would never have to potty train or paper train your child's computer pet. You would never have to clean its litter box. Or give it baths. Or take it to the vet.

Yet a computer pet could be very rewarding. Like a computer friend, it could be a companion to your child. And your child could train it herself. She might train it to "fetch," or solve puzzles, talk, do tricks, or tell jokes.

Granted, a computer pet is not soft and cuddly like a warm kitten or a fluffy rabbit or gerbil. But turtles and goldfish aren't cuddly either, and they make popular pets.

And why stick to pets that are common?

A child can create a computer pet unlike any animal on earth. He can fabricate it directly from his imagination or his dreams.

How many legs does the pet have?

How many eyes?

Does the pet have feathers, fur, scales, or skin?

What color?

What does the pet eat?

For answers to all these questions, just ask your child.

What does the pet look like? You and your child can draw the pet on the picture screen.

What does the pet sound like? Use your computer's SOUND command.

Your child's pet can be a monster, a dragon, a dinosaur, or a schmoo. Or a nice brown pony. And your child could learn all about ponies and horses as she created her pet.

In fact, having a computer pet could be a first step toward owning and caring for a real, live pet. Your child could learn all about her computer pet. If it lived, thrived, and was happy, perhaps she could graduate to the real thing.

Or a child might have several computer pets and several real pets.

Or just computer pets.

You and your child decide.

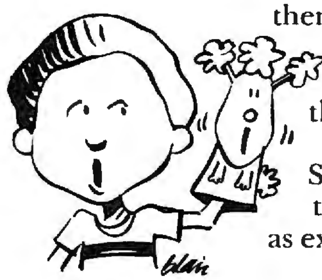
The Creature Inside The Computer

This column is called the "World Inside the Computer." In earlier columns I talked about the shape of that world and the shape of the worlds to come.

There are worlds inside your computer. But there are creatures inside your computer, too. In fact, your computer is swarming with them.

The creatures are really programs. You can write them yourself. Once they are written, your child should be able to call them up just like she'd whistle for her dog when she woke up, or cry "Here, kitty, kitty!"

Your computer is like a puppet theater filled with creatures that aren't real, yet are, nevertheless, very lifelike and entertaining. Your child should be able to put his hands inside the computer "puppets," then bring them to life, control them, and interact with them. The more puppets, the richer the child's experience with the computer. Sitting your child in front of the computer could become as exciting as turning him loose in a big-city zoo.



Building The Computer Zoo

In this column, we have looked at a couple of the issues related to giving your computer a personality. In the next three columns we will turn to specific programs that impart a simple personality to your family or school's computer. In the September column we'll build a computer friend. In the October column we'll modify and expand the friend to become a friendly teacher. And, in the November column, we'll create a computer pet. As always, I'm looking forward to hearing your comments, suggestions, and ideas. What kind of pet should we create? A dog? A cat? A dragon? A horse? What do you think?

Write and let me know. ©

FORTH programmers:
COMPUTE! is
looking for screens
and applications
articles.

This program gets smarter every time you use it. It shows one way a computer could "learn." For Atari and Commodore computers.

Guess That Animal!

Simulating Learning In Computers

Daniel Hastie
Garland, TX

The human mind learns by associating new information in some way with old information. But how should this be modeled in a computer? The answer to this question depends on the type and volume of information. I would like to propose a game where a modest amount of simple information makes a binary tree appropriate. Branches can be labeled so that the ordering of the data is not critical.

The Modeling Approach

Guessing an animal is a game which many children approach on a random basis. Especially when the starting letter is given, the child begins a scan through his "animal vocabulary" looking for a possible match with any entry beginning with that letter. Since there is no evidence that information is filed in their little heads alphabetically, this is usually a very hit-or-miss approach.

A more advanced player immediately begins a binary search by asking a series of Yes or No questions about the animal to distinguish the category to which it belongs. This enables him to search a smaller memory region (assuming associated items are stored closely together) and find the animal's name. We would like to simulate this approach with the computer.

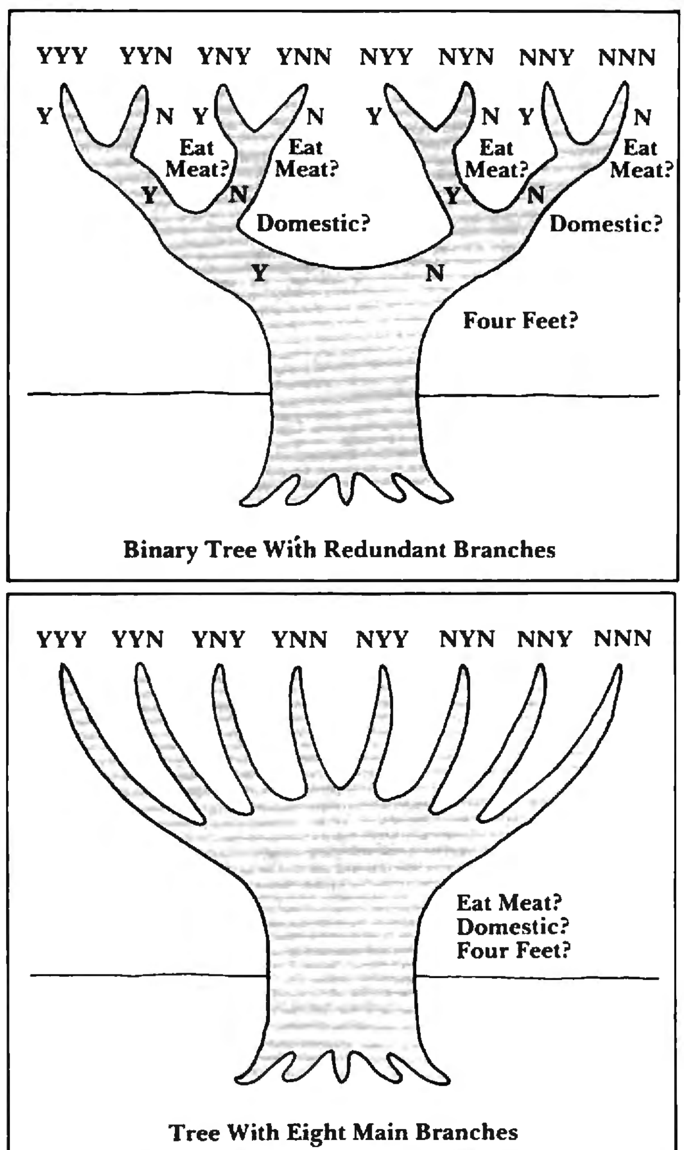
The Program

Some time ago I found a program by Mike Gabrielson in *Recreational Computing* which presented the idea of a computer guessing an animal type, but the listing provided was in Pascal and it wasn't clear to me at the time how to adapt it to my PET. However, that planted the idea which finally sprouted and grew into Program 1.

Let's walk our way through the program and hit the key points. After reading the data file (refreshing its "animal memory") and printing a starting message, a set of questions to be used with every game is asked. These starter questions are labeled by an "S" in the T$ array. I chose to use three starter questions producing eight initial entries or tree limbs. This cuts down on redundant questions in the various main branches (see Figure 1). You might experiment with more if you find redundancy increasing.

To these questions, an answer string is formed by concatenating [*chaining together*] the responses into QA$ in line 410. Then a search is started, looking through array T$ for a match to QA$. When found, the corresponding entry in array Q$ is printed. Initially, this will be a branch tip (a final guess), but it is handled the same as any other distinguishing question. Again the answer is concatenated and a new search initiated. If no match is found, then the computer branches down to line 450 in the program. If the last answer was *yes*, the computer was successful in finding the

Figure 1. Comparison of Tree Branching Techniques



Binary Tree With Redundant Branches

Tree With Eight Main Branches

Figure 2. Sample conversation between computer and human. (Human responses between stars.)

```
THINK OF AN ANIMAL AND I
WILL TRY TO GUESS IT.

PRESS SPACE BAR WHEN READY

DOES IT HAVE FOUR FEET?      **Y**
IS IT DOMESTIC?              **Y**
DOES IT EAT MEAT?            **Y**
IS IT A DOG?                 **N**
I GIVE UP, WHAT IS IT?       **A CAT**

WHAT WOULD BE A GOOD QUESTION TO
TELL THAT FROM A DOG?
**DOES IT BARK?**
WHAT WOULD BE THE RIGHT ANSWER
FOR A CAT?                   **N**
WOULD YOU LIKE TO TRY AGAIN?
**Y**

THINK OF AN ANIMAL AND I
WILL TRY TO GUESS IT.

PRESS SPACE BAR WHEN READY

DOES IT HAVE FOUR FEET?      **Y**
IS IT DOMESTIC?              **Y**
DOES IT EAT MEAT?            **Y**
DOES IT BARK?                **N**
IS IT A CAT?                 **Y**
GOOD, I GUESSED IT!
WOULD YOU LIKE TO TRY AGAIN?
```

animal and, after an appropriate message, asks if another game is desired.

The interesting part is the sequence of steps processed if the last answer is *no*. Giving up, the computer asks what the animal is and what question could be asked to help distinguish the animal from what the computer guessed. Then the proper answer for this animal is requested (see Figure 2). The old final guess is cut off from its place in the tree, and the new question fork NQ$ (with the new animal as one of its tips) is grafted in. Then the old final guess is grafted back in as the second tip of the fork. Figure 3 shows the horticultural image. The two tips are also keyed with the appropriate answer string in array T$ and play begins again.

If the human player decides to stop, the computer updates its memory base, erasing the old and saving the new one. The next time someone plays "Guess That Animal!" the computer will be smarter. Watch out! It may do better than you!

For tape-bound systems, use Figure 4 to replace the disk commands with appropriate tape commands.

But where did the original data come from? Not to worry; use Program 2, the Data Starter program.

Figure 3. Branch Grafting Procedure

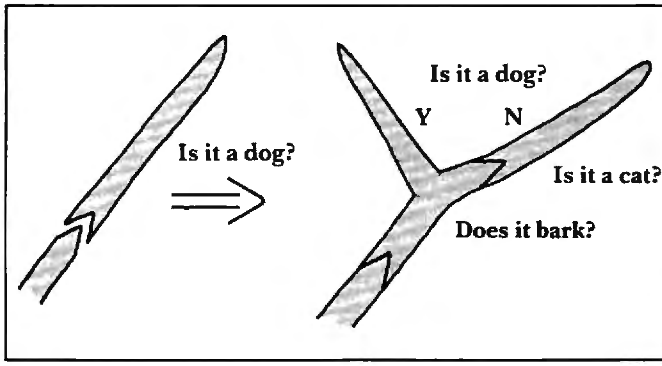


Figure 4. Program Changes for Tape Operation

```
130 PRINT"PUT ON DATA TAPE"
140-170 DELETE
180 OPEN 1,1,0,"ANIMAL DATA"

780 PRINT"REWIND DATA TAPE, PRESS
SPACE BAR WHEN READY"
790 GET A$: IF A$<>" " THEN 790
810 OPEN 1,1,2,"ANIMAL DATA"
```

For DATA STARTER, replace line 120 in Program 2 with the statement shown in 810 above.

Program 1. PET/CBM Version

```
100 REM *** GUESS THAT ANIMAL!
110 DIM T$(100), Q$(100)
120 G$="GOOD, I GUESSED IT!"
130 PRINT"WHAT DRIVE ARE YOU USING?
    "
140 :GET DN$
150 : IF DN$="1" OR DN$="0" THEN 17
    0
160 :GOTO 140
170 REM *** READ STORED DATA ***
180 OPEN 1,8,2,DN$+":ANIMAL DATA,S,
    R"
190 INPUT#1,NS,N
200 FOR I=0 TO N
210 :INPUT#1, T$(I), Q$(I)
220 NEXT : CLOSE1
230 :
240 REM *** START GAME ***
250 PRINT CHR$(147): REM CLR SCREEN

260 PRINT"THINK OF AN ANIMAL AND I"

270 PRINT"WILL TRY TO GUESS IT."
```

```
280 PRINT: PRINT"PRESS SPACE BAR WH
    EN READY"
290 GET A$: IF A$<>" " THEN 290
300 REM SET UP ANSWER STRING AND PO
    INTER
310 QA$="": REM NULL ANSWER STRING
320 FOR ZZ=0 TO NS: GOSUB 370: NEXT

330 FOR I=NS+1 TO N: IF T$(I)=QA$ T
    HEN 350
340 NEXT: GOTO 450:  NO MATCH FOUND

350 ZZ=I: I=N: NEXT
360 GOSUB 370: GOTO 330
365 :
370 PRINT  Q$(ZZ);: REM PRINT & GET
     ANSWER
380 :  GETA$
390 :   IF A$="Y" OR A$="N"THEN 410

400 :  GOTO380
410 QA$=QA$+A$
420 PRINT" ";A$
430 RETURN
440 :
450 REM *** GUESSED IT OR GIVE UP**
    *
460 :  IF A$="Y" THEN PRINT G$: GOT
    O 700
470 PRINT"I GIVE UP, WHAT IS IT";
480 INPUT NA$
490 PRINT: T$=Q$(ZZ): TL=LEN(T$)
500 PRINT"WHAT WOULD BE A GOOD QUES
    TION TO"
510 PRINT"TELL THAT FROM "; RIGHT$(
    T$,TL-6)
520 INPUT NQ$: REM NEW QUESTION
530 IF RIGHT$(NQ$,1)<>"?" THEN NQ$=
    NQ$+"?"
540 PRINT"WHAT WOULD BE THE RIGHT A
    NSWER"
550 PRINT"FOR "NA$;
560 :INPUT A$ :RA$=LEFT$(A$,1)
570 : IF RA$="Y" OR RA$="N" THEN 59
    0
580 :GOTO560
590 N=N+1: PA$=LEFT$(QA$,LEN(QA$)-1
    )
600 :
610 REM *** REPLACE FINAL GUESS WIT
    H NEW QUESTION
620 T$=Q$(ZZ): Q$(ZZ)=NQ$
630 X$="N": Z$="Y"
640 :  IF RA$="N" THEN X$="Y": Z$="
    N"
650 REM *** ADD OLD & NEW FINAL GUE
    SS
660 T$(N)=PA$+X$: Q$(N)=T$: N=N+1
670 T$(N)=PA$+Z$: Q$(N)="IS IT "+NA
    $+"?"
680 :
690 PRINT
700 PRINT"WOULD YOU LIKE TO TRY AGA
    IN?"
710 :GETA$
720 : IF A$="N" THEN 770
730 : IF A$="Y" THEN 250
740 :GOTO710
750 :
760 :
770 REM *** SCRATCH OLD DATA AND SA
    VE NEW
780 OPEN 1,8,15:PRINT#1,"S"DN$":ANI
    MAL DATA"
790 CLOSE1
795 :
800 C$=CHR$(13)
810 OPEN 1,8,2,DN$+":ANIMAL DATA,S,
    W"
820 PRINT#1, NS;C$;N;C$;
830 FOR I=0 TO N
840 : PRINT#1, T$(I);C$;Q$(I);C$;
850 NEXT: CLOSE1
860 END
```

Program 2. PET/CBM Version

```
100 REM DATA STARTER
110 N=10: NS=2: C$=CHR$(13)
120 OPEN1,8,2,"0:ANIMAL DATA,S,W"
130 PRINT#1, NS;C$;N;C$
140 FOR I=0 TO N
150 READ T$, Q$
155 Q$=Q$+"?"
160 PRINT#1, T$;C$;Q$;C$;
170 NEXT: CLOSE1: END
180 DATA   S,DOES IT HAVE FOUR FEET

190 DATA   S,IS IT DOMESTIC
200 DATA   S,DOES IT EAT MEAT
210 DATA NNN,IS IT A WORM
220 DATA NNY,IS IT AN EAGLE
230 DATA NYN,IS IT A CHICKEN
240 DATA NYY,IS IT A MAN
250 DATA YNN,IS IT AN ELEPHANT
260 DATA YNY,IS IT A WOLF
270 DATA YYN,IS IT A COW
280 DATA YYY,IS IT A DOG
```

Program 3. Atari Version

```
100 REM *** GUESS THAT ANIMAL! ***
110 MXA=50:REM MAXIMUM # OF ANIMALS.  ADJUST
     FOR MEMORY SIZE
115 DIM T$(MXA*10),T2$(30),TL(MXA),Q$(MXA*30
```

```
    ),QL(MXA),G$(20),TEMP$(30),NA$(10),NQ$(3
    1),FILE$(12)
117 DIM QA$(30)
120 G$="Good, I guessed it!"
130 OPEN #2,4,0,"K:":GRAPHICS 0
140 PRINT "Tape or Disk? ";
150 GET #2,A:IF A<>ASC("T") AND A<>ASC("D")
    THEN 150
160 IF A=ASC("D") THEN ? "DISK":? :FILE$="D:
    ANIMAL.DAT":GOTO 180
170 ? "TAPE":? :? "Position data tape, press
     PLAY,":? "then press RETURN":FILE$="C":
    ?
180 TRAP 190:OPEN #1,4,0,FILE$:GOTO 200
190 CLOSE #1:? "Error #";PEEK(195);" on read
    ing...":? "Try again.":GOTO 140
200 INPUT #1,NS,N
205 FOR I=0 TO N
210 INPUT #1,TEMP$:T$(I*10+1)=TEMP$:TL(I)=LE
    N(TEMP$)
220 INPUT #1,TEMP$:Q$(I*30+1)=TEMP$:QL(I)=LE
    N(TEMP$)
230 NEXT I:CLOSE #1:TRAP 40000
240 REM *** START GAME ***
250 GRAPHICS 0
260 PRINT "Think of an animal and I"
270 PRINT "will try to guess it."
300 REM SET UP ANSWER STRING AND POINTER
310 QA$="":REM NULL ANSWER STRING
320 FOR ZZ=0 TO NS:GOSUB 370:NEXT ZZ
330 FOR I=NS+1 TO N:IF T$(I*10+1,I*10+TL(I))
    =QA$ THEN 350
340 NEXT I:GOTO 450:REM NO MATCH FOUND
350 ZZ=I:I=N:NEXT I
360 GOSUB 370:GOTO 330
365 REM
370 ? Q$(ZZ*30+1,ZZ*30+QL(ZZ));
380 GOSUB 1000
410 QA$(LEN(QA$)+1)=CHR$(A)
430 RETURN
440 REM
450 REM *** GUESSED IT OR GIVE UP ***
460 IF A=ASC("Y") THEN ? G$:GOTO 700
470 ? "I GIVE UP, WHAT IS IT";
480 INPUT NA$:IF LEN(NA$)>10 THEN ? "** TOO
    LONG **":? "WHAT IS IT";:GOTO 480
490 ? :T2$=Q$(ZZ*30+1,ZZ*30+QL(ZZ)):TL=QL(ZZ
    )
500 ? "What would be a good question to"
510 ? "tell that from ";T2$(7)
520 INPUT NQ$
521 IF LEN(NQ$)>30 THEN ? "** TOO LONG **":G
    OTO 520
530 IF NQ$(LEN(NQ$))<>"?" THEN NQ$(LEN(NQ$)+
    1)="?"
535 IF LEN(NQ$)>30 THEN ? "** TOO LONG **":G
    OTO 520
540 ? "What would be the right answer"
550 ? "for ";NA$;"?";
560 GOSUB 1000
570 IF A<>ASC("Y") AND A<>ASC("N") THEN 560
590 N=N+1:TEMP$=QA$(1,LEN(QA$)-1)
610 REM *** REPLACE FINAL GUESS WITH NEW QUE
    STION
620 T2$=Q$(ZZ*30+1,ZZ*30+QL(ZZ)):Q$(ZZ*30+1,
    ZZ*30+LEN(NQ$))=NQ$:QL(ZZ)=LEN(NQ$)
630 X=ASC("N"):Z=ASC("Y")
640 IF A=X THEN X=Z:Z=A
650 REM *** ADD OLD & NEW FINAL GUESS
660 T$(N*10+1,N*10+LEN(TEMP$))=TEMP$:T$(N*10
    +LEN(TEMP$)+1)=CHR$(X):TL(N)=LEN(TEMP$)+
    1
665 Q$(N*30+1,N*30+LEN(T2$))=T2$:QL(N)=LEN(T
    2$):N=N+1
670 T$(N*10+1,N*10+LEN(TEMP$))=TEMP$:T$(N*10
    +LEN(TEMP$)+1)=CHR$(Z)
675 TL(N)=LEN(TEMP$)+1
680 Q$(N*30+1)="IS IT ":Q$(N*30+7)=NA$:Q$(N*
    30+LEN(NA$)+7)="?":QL(N)=LEN(NA$)+7
690 ?
700 ? "Would you like to try again?";
710 GOSUB 1000
720 IF A=ASC("N") THEN 770
730 IF A=ASC("Y") THEN 250
740 GOTO 710
770 REM *** SCRATCH OLD DATA AND SAVE NEW
780 IF FILE$="C" THEN ? "Rewind data tape, p
    ress PLAY and RECORD,"
790 IF FILE$="C" THEN ? "then press RETURN"
800 TRAP 805:OPEN #1,8,0,FILE$:GOTO 810
805 ? "Error #";PEEK(195);" when writing ani
    mal file!":CLOSE #1:END
810 PRINT #1;NS:PRINT #1;N
820 FOR I=0 TO N
830 PRINT #1;T$(I*10+1,I*10+TL(I))
840 PRINT #1;Q$(I*30+1,I*30+QL(I))
850 NEXT I:CLOSE #1
860 END
1000 REM GET Y/N RESPONSE
1005 POKE 752,1:? " ";:POKE 53279,0
1010 GET #2,A:IF A>96 THEN A=A-32
1020 IF A<>ASC("Y") AND A<>ASC("N") THEN 101
     0
1025 IF A=ASC("N") THEN 1040
1030 FOR R=1 TO 3:? "YES{3 LEFT}";:FOR W=1 T
     O 5:NEXT W:? "YES{3 LEFT}";:FOR W=1 TO
     5:NEXT W:NEXT R:GOTO 1050
1040 FOR R=1 TO 3:? "NO{2 LEFT}";:FOR W=1 TO
      5:NEXT W:? "NO{2 LEFT}";:FOR W=1 TO 5:
     NEXT W:NEXT R
1050 POKE 752,0:? :RETURN
```

Program 4. Atari Version

```
100 REM DATA STARTER (ATARI)
110 NS=2:N=10:DIM Q$(30),T$(10)
120 OPEN #1,8,0,"D:ANIMAL.DAT":REM USE OPEN#
    1,8,0,"C:" FOR TAPE
130 ? #1;NS:? #1;N
140 FOR I=0 TO N
150 READ T$,Q$
160 Q$(LEN(Q$)+1)="?"
170 ? #1;T$:? #1;Q$
175 NEXT I:CLOSE #1:END
180 DATA S,DOES IT HAVE FOUR FEET
190 DATA S,IS IT DOMESTIC
200 DATA S,DOES IT EAT MEAT
210 DATA NNN,IS IT A WORM
220 DATA NNY,IS IT AN EAGLE
230 DATA NYN,IS IT A CHICKEN
240 DATA NYY,IS IT A MAN
250 DATA YNN,IS IT AN ELEPHANT
260 DATA YNY,IS IT A WOLF
270 DATA YYN,IS IT A COW
280 DATA YYY,IS IT A DOG
```

©

A Monthly Column

Learning With Computers

The PILOT Language

Glenn Kleiman
Teaching Tools: Microcomputer Services
Palo Alto, CA

Last month I compared BASIC and LOGO as languages for teaching computer programming to children. This month's topic is another language useful in education: PILOT. The name PILOT is an acronym for Programmed Instruction, Learning Or Teaching. PILOT is a simple computer language, originally designed for teachers to use in writing their own lesson programs. It is also useful for teaching children about programming.

All adequate versions of PILOT contain a small set of commands for displaying information and questions on the screen, waiting for answers to be typed on the keyboard, checking whether an answer matches any of a list of expected responses, branching to different parts of the program, and performing simple calculations. These commands, generally referred to as Core PILOT commands, can be used to create programs which make the computer converse with a person.

Sample PILOT Dialogues

Here are some dialogues between students and a PILOT program. The students' responses are italicized.

What is your name?
Susan
Hello, Susan.
I'm going to ask you some questions about the United States.
Which is the largest state?
Alasska. It's even bigger than Texas.
That's right, Susan. Alaska is the largest state.
...

What is your name?
Jason
Hello, Jason.
I'm going to ask you some questions about the United States.
Which is the largest state?
I don't know. Maybe its Hawaii.
No, Jason, the correct answer is Alaska.
...

What is your name?
Sam
Hello, Sam.
I'm going to ask you some questions about the United States.
Which is the largest state?
N.Y.
That is a large state with a lot of people, but it is not the largest in size.
Please try again.
How about Texas?
No, Texas is a big state, but it's not the largest of all 50 states. Please try again.
I know, its Alasca!
That's right, Sam. Alaska is the largest state.

The program gives intelligent responses to the students' answers. Susan's dialogue shows the computer can recognize the correct answer, even though she misspelled it and named another state in the same answer. In the next dialogue, Jason gives an incorrect answer and is told the correct one. In the last dialogue, Sam's first two answers are incorrect, but he shows he knows which states are among the largest. The computer gives him hints and additional chances. Notice that the computer recognizes the abbreviation *N.Y.* and another misspelling of Alaska.

The PILOT Program

The three dialogues shown above are all the result of a fairly simple PILOT program. The program starts with:

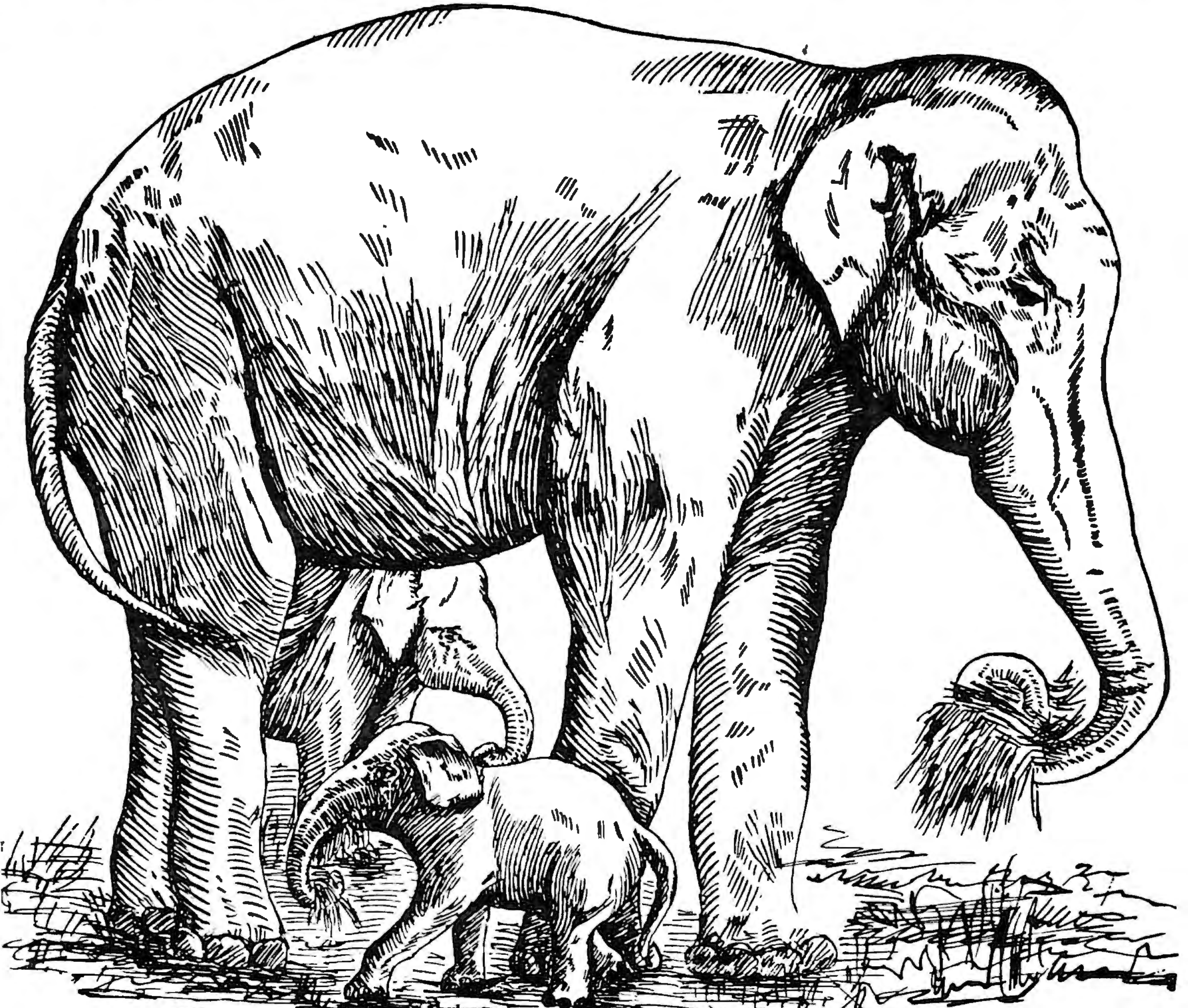
```
10 T: What is your name?
20 A: $NAME
30 T: Hello, $NAME.
40 T: I'm going to ask you some questions about the
      United States.
```

In PILOT, each command is represented by a letter followed by a colon. (Many versions of PILOT do not require line numbers, but they are included in the example program for ease of reference to specific lines.) T: is the TYPE command. It displays information on the screen, such as the question in line 10. A: is the ACCEPT command. It causes the computer to wait for a response typed on the keyboard. It is used in line 20 to accept the student's name. The name is then stored in the variable $NAME so it can be used later. Line 30 displays a greeting, and line 40 tells the student what the lesson is about.

The program continues with:

```
50 *FIRST-QUESTION
```

Micro-Ed is EXPANDING!



Send for free educational catalogs ❊

❊ Specify: Pet • Vic • TI • Apple • TRS-80 color computer

•The New Commodore 64

you can telephone us at

612-926-2292

Micro-Ed, Inc.

PO Box 24156 Minneapolis, MN 55424

```
60 T: Which is the largest state?
70 A:
80 M: ALAS
90 TY: That's right, $NAME. Alaska is the largest
   state.
100 JY: *SECOND-QUESTION
```

Line 50 is given the label *FIRST-QUESTION. One advantage of most versions of PILOT over BASIC is that lines and variables can be given names which convey what they do. This makes PILOT programs easy to read.

Line 60 displays a question and line 70 waits for a response. Line 80 is our first example of the MATCH command. M: ALAS tells the computer to search through the last answer entered and check if it contains the string of letters ALAS. Using ALAS means that close spelling errors of ALASKA will also be accepted. Extra words and spaces, and upper-case to lower-case inconsistencies are ignored in matching. All the following answers contain matches to ALAS:

I think it is Alaska
ALASSKA
ALASCA
alaska
The biggest state is the alaskan one

The MATCH command is very powerful for writing question-answer programs. A single M: command can check for more than one possible answer (as in line 140 below) and proper use of this command can lead to accepting a wide variety of responses, including common misspellings. However, you have to be careful with the answers to be accepted. For example, the following answers also contain matches to ALAS:

Calasfornia.
Alas, I wish I knew.

Lines 90 and 100 have Y or YES conditionals. A YES conditional tells the computer to follow the command only if the last M: command found a match. If the student's last answer contained ALAS, a match would be found. The T: command in line 90 would then display the "That's right" message, and the J: or JUMP command in line 100 would cause the program to branch to the line labelled * SECOND-QUESTION. If the student's answer did not contain ALAS, then lines 90 and 100 would not cause any actions and the programs would continue with the next line (110).

The rest of the program for this question handles incorrect answers:

```
110 M: TEXAS
120 TY: No, Texas is a big state, but it's not the largest
    of all 50 states. Please try again.
130 JY: *FIRST-QUESTION
140 M: CALIF,NEW YORK,NY,N.Y.
150 TY: That is a large state with a lot of people, but it
    is not the largest in size. Please try again.
160 JY: *FIRST-QUESTION
170 T: No, the correct answer is Alaska.
180 *SECOND-QUESTION
190 [Program continues ...]
```

Lines 110 to 160 check for expected wrong answers and give corrective feedback and another chance. An important feature of well designed tutorials is checking for children's typical errors and correcting misconceptions before they cause problems in understanding later parts of the lesson. Line 170 is reached only when the student does not give any of the expected answers.

The above example, while not showing all the possibilities of Core PILOT, should give you an idea of the type of program for which Core PILOT is well suited. In addition to the Type, Accept, Match, Jump commands, Yes conditionals, and string variables shown in the example program, Core PILOT also allows simple calculations, numeric variables, other types of conditional decisions, and sub-programs or modules which can be called from any point in the program (like subroutines in BASIC).

PILOT Is Not Just For Teachers

While PILOT was designed for teachers to write tutorial programs, it is also a good language for introducing children to computer programming. PILOT can serve as a vehicle for getting children comfortable with computer programming and for introducing general programming concepts such as variables, conditionals, branching and modules. PILOT programs are easy to understand, and children can easily get started writing their own. Children enjoy creating computerized question-answer dialogues, and working with PILOT can encourage them to further develop language skills. Most versions of PILOT for personal computers add other features desirable in a language for children, such as ways of creating pictures, and good error diagnostics to facilitate debugging programs.

Versions Of PILOT For Atari, Apple And TRS-80 Computers

Many versions of PILOT are available for different computers. Here is some information about PILOTs for the computers most widely used in schools. (I have not found an acceptable PILOT for PET computers. Anyone know of one?) All of these versions contain the Core PILOT commands, so I will focus upon what else has been added.

Atari PILOT is an ideal system for novice computer users. It adds easy-to-use, yet powerful, grahics and sound capabilities to Core PILOT. The graphics component contains turtle graphics commands like those in the LOGO language. Pictures are created by telling a "turtle" which color pen to use and how to turn and move. As I discussed in last month's column, and as is shown every month in David Thornburg's "Friends of the Turtle" column, turtle graphics is an excellent system for children to create pictures and learn about programming. In addition, Atari PILOT contains a sound command which can produce up to four tones at one time, thereby playing chords as well as individual notes. This gives you full access to the sound capability of Atari computers and lets you create turtles which sing as they draw.

Atari PILOT also has other useful features. For example, it checks each line as it is entered into a program, so many errors are caught immediately. There is also a trace command which lets you run a program one command at a time to see how it works. There are two manuals for Atari PILOT, an instruction manual and a reference guide. Both are excellent. The clever illustrations in the reference guide give a sense of how user friendliness was considered in every aspect of this package.

The combination of PILOT, turtle graphics, sound, and excellent manuals in a single user-friendly package makes Atari PILOT my first choice as a language to teach children.

Atari PILOT, for Atari 400 and 800 computers, is available from Atari dealers in two different packages. An educator package, which sells for $129.95, contains the PILOT cartridge, the instruction and reference manuals and two tapes with sample programs, all put together in a nice binder. The home package, which sells for $79.95, contains just the PILOT cartridge and the instruction manual.

Users of Atari PILOT will also want a book by David Thornburg called *Picture This!*, published by Addison-Wesley. It is a tutorial guide to turtle graphics which also teaches a great deal about computer programming in general. As one would expect from David's **COMPUTE!** columns, the book is both informative and fun.

E-Z PILOT is an excellent, inexpensive PILOT which gives you access to the high resolution graphics capability of Apple II computers. E-Z PILOT lets you use upper and lower-case letters in several sizes and colors. These letters can be combined with high resolution pictures. There are commands for drawing lines and for using previously created shape tables and full screen graphic displays. Pictures created with a graphics tablet or many of the available graphics software packages can be used in E-Z PILOT lesson programs. You can also add sound to your lessons.

E-Z PILOT is easy for beginners to use. It provides a simple menu-driven editor for entering and modifying programs. Prompts for acceptable commands are shown, and syntax errors are caught as they are entered. Once a lesson is created and saved, it is automatically added to the menu of lessons the students see each time they start E-Z PILOT.

E-Z PILOT is a bit slow to work with at times, and the documentation needs to be elaborated. However, it is very suitable for creating computerized lessons and could also be used to introduce students to programming. It requires an Apple II with Applesoft and one disk drive. At $34.95 it is a best buy. Available from TECK Associates, Box 8732, White Bear Lake, MN 55110.

APPILOT II allows you to use high and low resolution graphics, mix text and graphics on the screen, add music and even some speech if used in conjunction with the Muse Voice program (sold separately). It also lets you time answers to questions and set maximum times to allow a student to answer. It is easy for novices to use and comes with a good manual and demonstration programs. APPILOT II requires an Apple II with Integer BASIC and one disk drive. It is available for $99.95 from Muse Software, 347 N. Charles St., Baltimore, Maryland 21201.

Apple PILOT is a sophisticated package of tools for creating lesson programs. It contains many commands and options in addition to those of Core PILOT. These include special editors for creating your own graphics and sounds. Apple PILOT is an excellent language for experienced programmers. However, it is not designed for novice computer users and is not suitable for teaching children about programming. Apple PILOT requires a 48K Apple and two disk drives for creating lessons, one disk drive for using lessons. It is available for $150.00 from Apple dealers.

Mark-PILOT is part of the GENIS package from Bell & Howell. This package also includes an authoring system called CDS (Courseware Development System). Mark-PILOT is easy to use, but limited. It does not let you use high resolution graphics, lower-case letters, sound or timing. At $300.00 for Mark-PILOT and CDS, I feel the package is overpriced. No backup copy is provided; a replacement disk costs $100.00.

TRS-80 Micro-PILOT contains the standard PILOT commands plus extensions for using the low resolution graphics and large letter set of the TRS-80 Model I and Model III computers. A special editor lets you design screen displays and store them on disk, to be retrieved when needed in a program. Micro-PILOT is easy to use and comes with a complete manual and a demonstration program which the manual describes in detail. It has several features usually found in BASIC but not in PILOT, such as mathematical functions (square root, tangent, and so on). Unfortunately, Micro-PILOT has adapted BASIC conventions for variable names when the standard PILOT ones are preferable.

Micro-PILOT is suitable both for teachers writing tutorial programs and for children writing their own programs. Available for Model I and Model III computers with one disk drive, Micro-PILOT is sold by Radio Shack dealers for $79.95.

TRS-80 Color Computer PILOT is an interesting new version. I have received a preliminary copy without complete documentation and I have not explored it in detail, but it clearly contains some excellent features. For example, it lets you combine high resolution graphics with upper and lower-case letters of various colors and sizes. It even lets you design your own letter set and add music to your programs. This version of PILOT, soon available from Radio Shack dealers, requires only 16K and a cassette recorder. ©

Review

Two VIC Word Processing Programs

Harvey B Herman
Associate Editor

Not Just A Game Machine

Commodore has in its ads made much of the fact that the VIC is not a video game but a real computer. This truthful claim was only academic at first as there was little available VIC software. Today the number of advertised programs appears to be growing exponentially, and the advantages in owning a true computer are quite evident.

As much as I like the VIC, I am aware of its shortcomings. The basic unit has minimal memory (5K read/write memory, RAM) and a display with a short 22 character line. Optional hardware has recently become available to correct these deficiencies. Note, however, that the programs described here will even work with the original unexpanded computer. Pretty good for a machine that retails for less than $300!

Why A Word Processor?

I always felt that it was not practical to type my own letters as I made too many mistakes. Furthermore, sitting at a typewriter, correcting errors with Liquid Paper, and retyping whole pages was not my idea of a good time.

The situation has now changed completely since I started using a word processor (*PaperMate* from AB Computers) on an older, expanded 2001 PET. I still make errors when I type letters, but they can be easily corrected before the final copy is printed.

I did not appreciate the convenience of a word processor until I actually started using one. For example, a draft letter can be run off with a few corrections without having to retype the whole letter. Moreover, a somewhat disorganized person like myself can keep up with his filing. Because I save all correspondence on disk, I always know that I can find a given letter by looking through a limited number of printed disk directories. Nothing gets lost.

Recently, I was given two VIC word processing programs to review, but I was a little dubious. A PET makes an excellent word processor, but it was not clear that a VIC would, particularly a VIC with a limited amount of memory. For program evaluation, I used additional hardware – a 3K RAM expander and a VIC 1515 printer. (The extra RAM allowed longer text files, but was not essential.) If VIC word processing intrigues you, read on to learn more about the good and bad features of each program.

Un-Word Processor

The accompanying eight-page manual describes this program as a simple word processor. I would agree with that description and even add "very." This software may seem strange at first. It consists of two machine language text formatting programs, one for the VIC 1515 printer and the other for an RS-232 printer. The formatting program appropriate for your system is loaded and protected from BASIC with a few POKEs. The instructions are quite explicit even if the rationale might not be clear to a beginner.

The next order of business is the writing of what I would call a pseudo-BASIC program. The first statement is a SYS call to the formatting program, and the subsequent statements, a text file not normal BASIC, are numbered lines of text headed by a double quote. Corrections are made in the text file, using the VIC screen editor, after the line is entered. The file can be saved on cassette or disk the same as any other program.

The text file is printed by simply RUNning the pseudo program. Return and line feed are added automatically to the printout, about every 72 characters. The line length may be adjusted as desired. A second double quote can be used to prematurely terminate a line as at the end of a paragraph or to produce a blank line.

Although this program is admittedly simple, I don't mean to imply that it is not useful. In some cases simple is better. It would be particularly easy for elementary school students, who might be intimidated by a more comprehensive program. The only minor flaw that I found was a few easily recognized errors in the manual.

The VIC Typewriter

The 13-page manual that accompanies this program says, up front, that it is a very simple word processor. However, it is closer to a real one than the program just described. It consists of a combined BASIC driver and a machine language subprogram. Text files, written with the program, can be saved to either cassette or disk, and retrieved later. On command, the text file can be printed on the one supported printer, the VIC 1515.

The manual is slick and professionally done. A summary of the 18 direct commands and the seven indirect (embedded) commands precedes a detailed description of each. I found myself referring to this summary quite often at first, but I expect that this would not be necessary after extended use.

There are several features of this program that I really like. It continuously displays the maximum number of lines available (185 lines for the 3K expanded VIC) and the current line number. The VIC function keys are used to advantage. For example, one key can be used to delete a line and another to page forward in the text. Enhanced or reversed printed characters are easy to intersperse in the text using special symbols. A tab key (five spaces) is even provided.

Other features were difficult to use or were absent. One function key, corresponding to "delete text," is too easy to use, while another, "delete character," is tricky to get just right as it does not have instantaneous action. There is no page back key, so one must page forward and wrap around to the start through the back door. Support for an RS-232 printer is lacking. It could have been easily added as an option to be separately loaded. I could go on, but we would not expect it to contain all the bells and whistles of much more expensive programs.

In a comparison between the two programs, the *VIC Typewriter* is a clear winner. This program has so many more features that it would be the one to choose if you want a serious program. On the other hand, having more features makes it more difficult to learn. If you are a VIC owner in the market for a simple word processor, check these programs out at your closest VIC emporium. You will get a taste of an important application for personal computers. Choose the one that suits you best.

Please note – problems associated with the early release of Home Calculation Pack are being corrected; be sure you get the updated version.

Un-Word Processor
Midwest Micro Associates
P.O. Box 6148
Kansas City, MO 64110
$12.95

The VIC Typewriter (one of six programs in Home Calculation Pack VT 107A)
Commodore Business Machines
681 Moore Road
King of Prussia, PA 19406
$59.95

©

Review: BASIC A+

Charles Brannon
Editorial Assistant

What is your "wish list" for the ultimate BASIC? What statements would you have? How large could the language afford to be? Atari owners can choose from three BASICs: Atari 8K BASIC, Microsoft BASIC, or BASIC A+. The original BASIC was designed to fit into an 8K cartridge, limiting its potential, whereas the latter two languages load as machine language programs from disk, and can be as large as memory permits. These *softloaded* languages, therefore, require a disk drive and at least 32K of RAM. The increased size of these languages means less memory for writing programs, though. With a maximum 48K of memory, you have a bit over 20K remaining.

An Extended BASIC

BASIC A+ is upward compatible with Atari 8K BASIC. That is, programs written in Atari BASIC will run without modification in BASIC A+. Prior programming experience with Atari BASIC transfers to BASIC A+. In effect, BASIC A+ is an extension of Atari BASIC, adding a multitude of commands and improving others.

Most of the minor discrepancies and bugs of Atari BASIC have been corrected. For example, the RUN command did not clear out arrays or strings as it should, and does, in BASIC A+. You can now use a subscripted variable in a READ or INPUT statement, such as READ A(I).

Have Your Cake And Eat It Too...

BASIC A+ narrows the compatibility gap between Atari BASIC and the popular Microsoft BASIC with options such as prompts in INPUT statements. For example,

```
INPUT "What is your name?",NAME$
```

TAB is also supported, as in PRINT TAB(20);X(I). TAB even works with a printer. String concatenation is provided with ",". A$=A$+B$ becomes A$=A$,B$. Since BASIC A+ includes equivalent commands for most Microsoft statements (excluding strings), you get the best of both worlds!

BASIC A+ adds program debugging tools such as TRACE and TRACEOFF, which activate and cancel the display of line numbers during a program's execution, and LVAR, which lists all variables. Error messages are now non-cryptic English phrases, such as NO SUCH LINE # instead of ERROR - 12. BASIC error handling with TRAP is made easier with ERR(0), which returns the error number; ERR(1), which gives the offending line's number; and CONT, which continues execution on the line following the error (RESUME in other BASICs).

Extended I/O

BASIC A+ adds DOS commands to BASIC, making disk handling by a program much easier. Compare these two statements which delete a file from the disk:

```
XIO 33,#1,0,0,"D:TEMP"
ERASE "D:TEMP"
```

Also included are PROTECT and UNPROTECT (a.k.a. LOCK/UNLOCK), RENAME, and DIR. Yes, you can now list the directory without going to DOS!

BASIC A+ can save and load binary files. These files can be any section of memory. For example, a word processor could use BPUT to save a text file at top machine I/O speed or use BGET to recall another. This formerly required many POKEs and a machine language program to call the CIO. RPUT and RGET are useful for fixed-length files produced with NOTE and read with POINT. For example, to PRINT a number to such a file could produce anything from "0" to "10" to "3.1415927" to "1.562 E+42." RPUT will output the number as 7 bytes, using the internal storage format for numbers. RGET will read these bytes and reconstruct the number. This built-in "packing" and "unpacking" capability is vital for efficient record processing.

PRINT USING

BASIC A+ supports deluxe formatting via PRINT. For example, PRINT USING "###.##" would format a three-digit number in dollars and cents format, automatically padding unused digits with spaces (or zeroes to the right of the decimal point). So 33.345 would appear as "b33.35". (The "b" denotes a blank.) Notice that PRINT USING automatically rounds numbers to fit the format field. "Pi" printed in the same format would appear as "bb3.14". Other format characters are "$" (floating dollar sign), "," (comma formatting such as 1,000,000 for one million), "&" (fills unused digits with zeroes), "*" (pads blank digits with asterisks for number protection, such as check printing), and "+" or "-" (which force the appropriate sign to be printed where specified). String formatting is provided with "%" and "!" which perform right or left justification on the string.

An Ideal Language?

BASIC A+ is written by programmers who have both designed Atari BASIC and have used it extensively. Therefore, it reflects the "wish list" of these programmers. If you're a serious programmer, you'll find this wish list comes close to yours – BASIC A+ is a programmer's programming language. Examples of this are seen throughout the language. Most obvious of these are the structured programming constructs of "IF...ELSE...ENDIF" which permit multi-line, multi-clause comparison and evaluation, and "WHILE...ENDWHILE" which allow a structured loop without a GOTO. The SET function enables or disables certain features. For example, SET 0,128 disables the break key; SET 2,ASC(":") changes the INPUT prompt from a question mark to a colon; SET 5,1 will allow program entry in lower case (converts to uppercase). Believe it or not, there is even a SET for the ANSI-standard FOR/NEXT loop. SET 3, 1 will allow a FOR/NEXT loop to execute zero times. This statement:

```
FOR I=30 TO 1:PRINT I:NEXT I
```

would normally print "30" and stop – executing at least once, no matter what the FOR range is. Almost all microcomputer BASICs do this, but the SET statement will permit BASIC to ignore such loops, making the Atari one of the only microcomputers to comply with the ANSI standard.

Programming convenience is enhanced with the DPOKE and DPEEK commands which simulate 16-bit PEEK and POKE commands. For example, to find the start of screen memory with SAVMSC ($58), you would code:

```
SCR=PEEK(88)+256*PEEK(89) in Atari BASIC, or
SCR=DPEEK(88) in BASIC A+.
```

Obtaining the same address from the display list shows the convenience even more dramatically:

```
A=PEEK(560)+256*PEEK(561)+4
SCR=PEEK(A)+256*PEEK(A+1)
vs. SCR=DPEEK(DPEEK(560)+4)
```

DPOKE lets you place sixteen-bit results in memory without breaking the number into two 8-bit parts. For example, DPOKE 12,1536 replaces POKE 12,0:POKE 13,6 and DPOKE 12,A replaces this piece of code:

```
POKE 13,INT(A/256):POKE 12,A-PEEK(13)*256
```

Other memory-manipulation commands are FIND and MOVE. FIND will search a string, even a huge Atari "megastring" of thousands of characters, for any substring. This has instant application for database management programs, and has infinite minor potential, such as menu selection:

```
GOTO FIND("PDCJN",A$,0)*10+1000
```

where P,D,C,J,N are the first letters of a menu. FIND will return from 0-5. MOVE will move a block of memory from one place to another. This can be used to move the character set in ROM into RAM for modification (and BASIC A+ conveniently reserves 1024 bytes for the character set). MOVE can be used for insert/delete functions, and even page-flipping. Imagine how much easier player/missile graphics would be with MOVE, where you could easily move players around in player memory with this machine language-speed command. Ah, but that's another topic...

A User-Friendly Language

One of the most attractive features of BASIC A+ is the ease of access to hardware features. The controller functions STICK, PTRIG, etc., and the SETCOLOR command represent Atari BASIC's user friendliness. These functions could easily be replaced with PEEKs and POKEs, but Atari BASIC makes the machine easy to use without memorizing a memory map. BASIC A+ continues this tradition. PEN(0) will return the horizontal position of the light pen, if present. HSTICK and VSTICK return delta-X and delta-Y offsets for the joystick. HSTICK, for example, returns +1 if the joystick is pushed right, -1 if the joystick is to the left, and 0 if the joystick is "horizontally centered." The two functions can be combined to easily update motion.

Player/Missile Graphics

This ease of use continues with a complete set of commands to control player/missile graphics, one of the Atari's best features. Formerly, programming P/M graphics was a nightmare of PEEKs, POKEs (so many POKEs!), sloowww FOR/NEXT loops for initialization and vertical movements, and machine language routines that were clumsy at best. BASIC A+ makes P/M graphics as easy as PLOT and DRAWTO. Setting player colors is accomplished with SETCOLOR's analog, PMCOLOR. PMCLR clears out a player. PMWIDTH sets the width of a player. The headache of memory allocations and bit selection is replaced with PMGRAPHICS, a statement similar to GRAPHICS. PMGRAPHICS (abbreviated as PMG.) does all the setup for you. Cancelling P/M is done with PMGRAPHICS 0. After the initialization, PMADR will return the memory location of any player, so POKEing or MOVEing bytes into it is easy.

The heart of the P/M graphics commands is PMMOVE, which positions the player anywhere on the screen. The horizontal and vertical positions of the player can be set together or separately.

Send for our FREE Software Catalog for the Atari 400/800 Computer

ATARI

LARGE SELECTION
SAVE A MINIMUM OF 10%
ALL PROGRAMS PRETESTED
FREE SHIPPING
MOST MAJOR BRANDS

RANDOM ACCESS MICROWARE

P.O. BOX 1704
BREA, CA 92621
(714) 671-1093

STATISTICS FOR RESEARCHERS

R-STAT

Friendly, easy to use, well documented.

INCLUDES:

- Database with transformations
- ANOVA N – way
- Analysis of COvariance One – way
- Stepwise Multiple Regression (with Ridge-regression)
- Univariate Statistics
- Mulitple range tests
- Crosstabs
- t – tests
- Graphics
- Menu-driven

DISKETTE & MANUALS **$350.00**

Specify 8K, A+, or Microsoft

ATARI® DOS II

For a FREE Brochure or further information please call or write.

DSC CO.
P.O. BOX 1035
CENTRALIA, WA 98531
(206) 736-3240

VISA & MASTERCHARGE ACCEPTED

BYTE-A-BIT COMPUTING CO.

"The Software Specialists" offer...

★ THE STATE OF THE ART ★

IN COMPUTER SOFTWARE

● NUMBER STUMPER: 24K CAS. $19.95 32K DISK 24.95	AN ARCADE GAME FOR ONE OR TWO PLAYERS. USES JOYSTICKS.
● UTILITY PAK #1: 32K DISK $29.95	DOS UTILITIES, AUTORUN, SYS, RENUMBER, VARIABLE LISTER AND CHANGER, AUTOMENU.
● UTILITY PAK #2: 32K DISK $24.95	DISK-SPEED ADJUST, CONVERT DECIMAL-HEX-BINARY #'S, COLOR & SOUND PROGRAMMER'S AID, SAVE-A-SCREEN UTILITY.
● GAME PAK #1: 24K CAS. $14.95 32K DISK $19.95	SCOREFOUR, DOTS, FIFTEEN, PLUS SURPRISE GAME! (A TOTAL OF FOUR GAMES). USES JOYSTICKS.

BYTE-A-BIT OFFERS YOU **FREE:** •PROGRAMMING •CONSULTATION

CALL OR WRITE FOR A CATALOG OF OUR PROGRAMS AND DETAILED INFORMATION ON OUR FREE OFFERINGS. SPECIALISTS IN SOFTWARE FOR THE ATARI* COMPUTER. GAMES, UTILITIES AND CUSTOM PROGRAMMING AVAILABLE.

DEALERS INQUIRIES INVITED

"Our Software Is Guaranteed For As Long As You Own It" If Any Program EVER Fails To Run, We Will Replace It FREE!!

BYTE-A-BIT COMPUTING CO.
28 ORCHID ROAD
LEVITTOWN, N.Y. 11756
(516) 796-1212

ALL OUR SOFTWARE MAXIMIZES THE USE OF ATARI GRAPHICS AND SOUND

ADD $2.50 SHIPPING AND INSURANCE. N.Y. RESIDENTS ADD SALES TAX. C.O.D. OR BLUE LABEL SHIPMENT ADD'L.

Horizontal position is an absolute location from 0-255, but vertical positioning is done with a relative *offset*. PMMOVE 0;1 would move the first player up one scan line. PMMOVE 0,100;-4 would move the player to horizontal position 100 and then move it down 4 scan lines. Two other commands are MISSILE and BUMP. MISSILE creates a missile (the "missile" in player/missile graphics) which is moved with PMMOVE. BUMP will read the collision registers. BUMP(0,3) will check for a collision between player 0 and player 3.

Unique Documentation

The manual for BASIC A+ comes as a sheaf of loose-leaf pages to be inserted into the BASIC reference manual. A logical concept – BASIC A+ extends BASIC, so its manual extends BASIC's. It is a little confusing, all the paper-shuffling, but gives you a complete reference. All changes are noted with special pages, and the table of contents is replaced as well. Additional appendices are added, as well as two new chapters, including a tutorial on player/missile graphics and the related commands. The manual is very well written, concise, but complete. It even includes "secret" internal memory locations to permit the user to customize BASIC A+. It is apparently easy to add new commands to BASIC A+. On an included demo/utility disk are two commands, RENUM and "@" which respectively RENUMber your program and permit an Atari load, to let you load Atari programs into BASIC A+.

A Few Notes And Bugs

1. The pre-reserved memory for characters and P/M graphics cannot be used for your BASIC program, although you can store machine language programs there.

2. A GRAPHICS command can apparently wipe out some of the reserved character graphics area.

3. BASIC A+ runs noticeably faster than Atari BASIC. One reason for this is that BASIC A+ only checks for the break key at the end of each line, not after each command. Unfortunately, this can cause a lock-up on single-line commands such as

```
FOR I=0 TO 1E9:PRINT PEEK(764):NEXT I   and
300 GOTO 300 :REM FREEZE FULL-SCREEN
    GRAPHICS UNTIL BREAK
```

SYSTEM RESET will abort such a loop.

4. OSS sells BASIC A+ for the Apple, so software producers can sell software that will run on either machine.

5. OSS has a run-time BASIC A+ which presumably runs faster, uses less memory, and has security features. You must pay a royalty to use it, however (unless OSS distributes your software).

6. OSS apparently has excellent user support, with a newsletter, and inexpensive update disks available.

7. Get OS/A+ with the BASIC. Although space does not permit a complete review here, be assured it's a bargain. For an extra $70 you get a full-power "system" DOS with easy to use commands such as REName. Other commands can be accessed from disk, such as COPY and HELP. BASIC A+ is loaded with "BASIC." OS/A+ is always available – no DUP.SYS or MEM.SAV! Included are numerous system utilities and EASMD, the disk-based upward-compatible Editor Assembler – how's that for a free bonus?

BASIC A+ is a feature-packed, easy-to-use language. The scope and range of BASIC A+ make it a truly professional language, worth the consideration of any serious programmer.

BASIC A+
Optimized Systems Software, Inc.
10379 Lansdale Ave.
Cupertino, CA 95014
$80 (or $150 with OS/A+)

©

VIC Communications: The RS-232 Interface

Jim Butterfield and Jim Law
Toronto

The VIC has very good communications potential built in. It's versatile and sophisticated: you can set transmission speeds and other characteristics. Buffering gives you a nice bonus: after you deliver characters to the interface, you may go about your computing business and the characters will be sent at the proper time. Similarly, input characters can be collected while you are doing other things, allowing you to pick them up at your own convenience.

The interface is called RS-232 (more about that later). Its versatility makes it very useful, but there are a couple of drawbacks. First, you can't use the serial disk/printer port or the cassette tape while the RS-232 is in gear. You could stop the RS-232, fire something out to printer or disk, and then resume communications; but that's sometimes awkward to do. Secondly, the Parallel User Port (PUP) and the RS-232 interface are the same connections, so it's difficult to use them both at the same time. If you are planning to sense a bunch of switches on the PUP and report their status over a communications line, you'll have some headscratching to do.

The RS-232 Connection

RS-232 is a communications standard that defines a whole bunch of wires (25) for connecting a terminal device (that's your VIC) to a communications device (that's your modem). It defines what the wires do. It defines a standard connector (which the VIC doesn't have). It defines standard voltages and currents (which the VIC doesn't observe).

Although all the RS-232 connections are defined, they are rarely all used. The most important connections are:

Signal	Description	Edge Connector	RS-232 Connector
From Modem to VIC:			
Data Rcv	Serial data input	Pins B and C	Pin 2
DSR	Modem OK	Pin L	Pin 21
CTS	Modem ready to transmit	Pin K	Pin 4
From VIC to Modem:			
Data Xmt	Serial data out	Pin M	Pin 3
DTR	VIC OK	Pin E	Pin 5
RTS	VIC is ready to transmit	Pin D	Pin 6
Ground connections:			
Gnd	Signal ground	Pin N	Pin 7
Pro	Chasis ground	Pin A	Pin 1

We should mention that the VIC leaves RTS on all the time just to keep the modem on its toes. Two more connections are fitted to the VIC but not used: RI and DCD connect to pins F and H. But you won't need to worry about them unless you're heavily into communications.

Hardware

We mentioned that the PUP connector is not a standard RS-232 connector, and the voltages furnished by VIC are not standard RS-232 voltages. This means that you'll need an adaptor to hook into your modem. Commodore should make these available soon.

If you're not willing to wait or just want to make your own adapter, two possible circuits are described below. The first uses discrete parts and any well-stocked junk box should harbor the necessary items. The second has minimum parts, but the ICs may be hard to find. If you are not handy with a soldering iron, perhaps a friend or a local electronics buff could assemble the interface for you: it's not a difficult project.

The simplest way to connect the VIC is with just the two data lines and ground. This "three wire" connection will work with most modems and any printer where handshaking is not a problem. If you are using the "junk box" circuit, make only as many circuits as you need: one input and one output should be enough for a modem.

Figure 1. Output Circuits

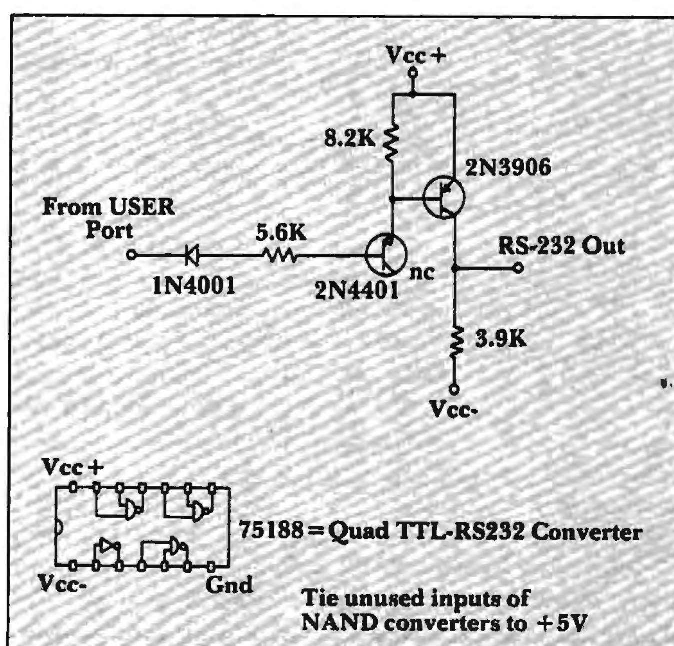


Figure 2. Input Circuits

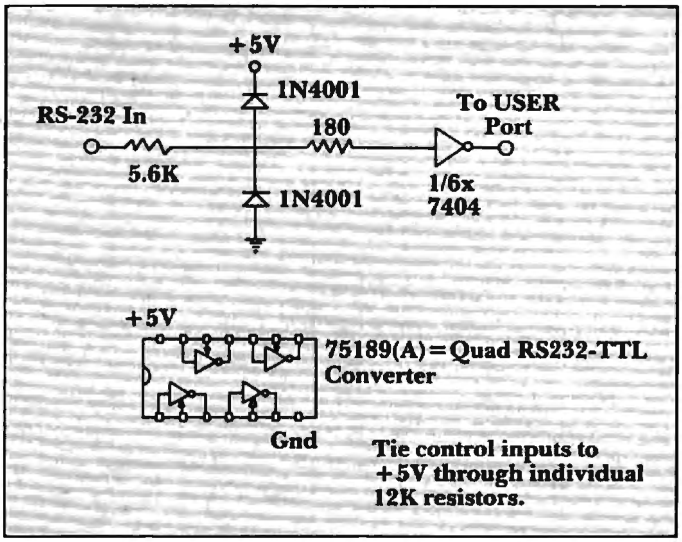


Figure 3. Power Supplies and Connection Diagram

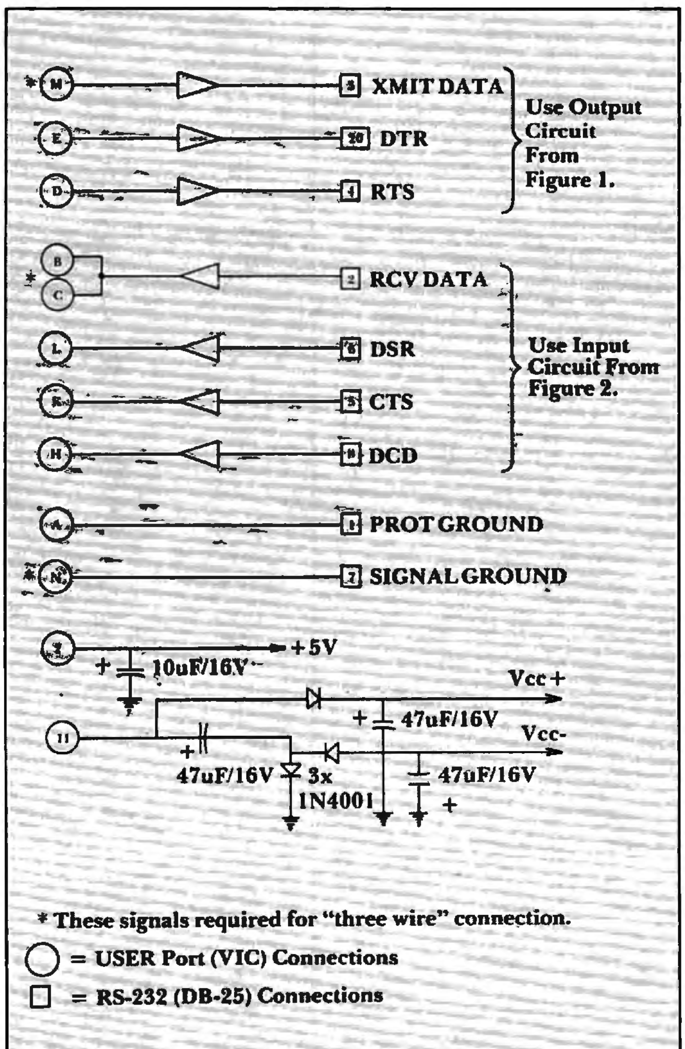
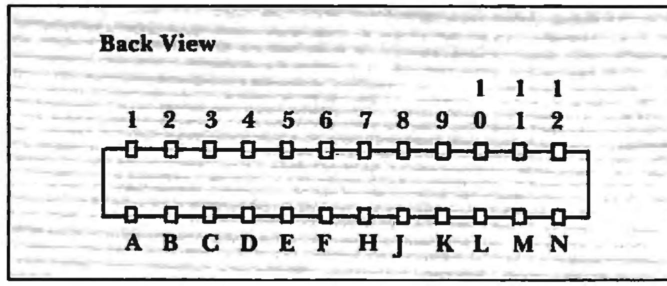


Figure 4. VIC User Port



The RS-232 interface is in place in your VIC as device 2. So all you need to do is to OPEN device 2 (for example, OPEN 1,2) and you may send and receive to your heart's content (PRINT#1,"ANYBODY THERE" : INPUT#1,R$). But you'd better know a few things about how it works before you do that.

Speed And Code And Other Things

The signals you are sending to the modem are serial. That means that a character is not sent all at once – the bits are sent one at a time at a certain speed. You must set that speed, and arrange a few other administrative details.

We need to signal speed, code, parity, and even the type of modem handshake we want. To open a 300 bit per second channel we would code:

```
OPEN 1,2,3,CHR$(6)+CHR$(0)
```

The value of 6 sets the speed to 300 bps; 5 would set 150 bps and 8 would set 1200 bps. We have turned off parity, assumed eight bits, and assumed ordinary ("simple 3-wire") RS-232 interfacing.

At the same time that we do this OPEN, something else happens: the VIC grabs two buffers for input/output use. It takes them from your available BASIC space. Try the above statement followed by a PRINT FRE(0) and watch 512 bytes disappear. The missing bytes will be returned to you when you say CLOSE 1.

There's another snag. These buffers are set up in the top of memory; if you have any strings stored up there you'll be in trouble. If you're going to use the RS-232, your program should OPEN it as its first command. That way there's no chance of confusion. Remember to CLOSE when you're done.

Working The Channel

You send with PRINT# and your program will be able to continue immediately while the data is sent. If you happen to fill up the buffer, the PRINT# will wait; you won't lose anything.

You can receive data with INPUT#, but it's a

little dangerous. INPUT of any sort always waits for a RETURN character to arrive; if it never arrives, your program will hang forever. Better to use GET#, which will give you back a character if it's there, otherwise it will return a null string (""). If you don't GET characters often enough, you will eventually end up with a full buffer and start losing things.

Errors are reported to you via the ST variable. This changes character completely; ST loses all of its previous meanings the moment you open the RS-232. There's a wide variety of things ST can report; for the moment, we'll make it simple by observing that if ST is not zero, there's something wrong. Each time you access ST, it will be cleared back to zero. You can tell if you're having communications problems and even count the errors if you like.

The Really Dumb Terminal Program

This program will talk to a modem connected as described above. Seven data bits and mark parity are assumed. Only uppercase letters are sent, but they will print on the VIC as lowercase because no conversions are done.

```
10 OPEN1,2,3,CHR$(38)+CHR$(160)
20 GET A$: IF A$="" THEN 60
30 IFA$=CHR$(147) THEN 90: REM CLEAR/HOME
   QUITS
40 A=ASC(A$) AND 127 : IF A=20 THEN PRINT#
   1, CHR$(8);: GOTO60
50 IF A>31 OR A=13 THEN PRINT#1, CHR$(A);
60 GET#1,A$ : IFA$="" THEN 20
70 A=ASC(A$) AND 127 : IF A=8 THEN PRINT C
   HR$(20); : GOTO20
80 IF A>31 OR A=13 THEN PRINT CHR$(A);
90 CLOSE1 : END
```

It's fun. It's sophisticated. But it is a little complex, and experience will be needed before you feel completely at home with VIC's communications features. ©

Keyprint, a fast machine language utility to copy the exact contents of a screen to a Commodore printer, was first published in **COMPUTE!**, *November/December, 1980, #7. Here are fourteen versions of this popular program, so you're sure to find the right one for your system. Included are complete descriptions of how to type in and use Keyprint.*

The Keyprint Compendium

Charles Brannon
Editorial Assistant

Keyprint's usefulness made it very popular. Within weeks, **COMPUTE!** began to receive many conversions, updates, and improvements to Keyprint. We've gathered them all together in this issue, with descriptive notes.

The BASIC programs can be entered and SAVEd as usual. RUN will PUT Keyprint into memory. The hex dumps are entered with the Machine Language Monitor.

To get to the monitor, type SYS 1024. You'll see something like:

```
     PC   IRQ   SR AC XR YR SP
.;  B780 E455 2C 34 3A 9D FA
```

Now list the block of memory occupied by the version of Keyprint you want to enter by typing .M 033A,03CA (or whatever is given in the listing). Then use the cursor to move over to the first two-digit (hex) number and begin typing, replacing the numbers on the screen. Press RETURN at the end of each line.

Now enter the correct ".save" line to SAVE Keyprint to tape or disk. For example,

```
.S "KEYPRINT",01,033A,03CB     (FOR TAPE)
.S "0:KEYPRINT",08,033A,03CB     (FOR DISK)
```

Be sure you SAVE Keyprint before you activate it. That way, if you made a mistake and the program "crashes," you can LOAD the program and proofread it. To activate Keyprint, enter the appropriate SYS, for example, SYS 826 [*033A in hex*]. You can then press the backslash " \ " (or whatever key is used) to dump the screen at any time to your printer.

The programs are keyed to the descriptions below:

1. Keyprint for Upgrade ROMs.

2. Eric Brandon's conversion of Keyprint to the Original ROM PET. Same operating instructions as Keyprint.

3. David Swaim has made a BASIC loader for Original ROM Keyprint for those who don't have a machine language monitor.

4. Mark W. Petersmeyer's conversion of Keyprint to the 4.0 ROM PET/CBM. Same operating instructions as Keyprint.

5. Jean Pierre Blanger has converted Keyprint to the 4.0 ROMs. SYS 826 to activate. Use the <SPACE> bar to print the screen, or SYS 843 under program control.

6. David Curtis has changed Keyprint so that it will

Program 1.

```
.M 033A 03CB
SYS 826
 033A 78 A9 03 85 91 A9 45 85
 0342 90 58 60 A5 97 C9 45 D0
 034A 03 20 51 03 4C 2E E6 A9
 0352 80 85 20 A9 00 85 1F A9
 035A 04 85 B0 85 D4 20 BA F0
 0362 20 2D F1 A9 19 85 22 A9
 036A 0D 85 21 20 D2 FF A9 11
 0372 AE 4C E8 E0 0C D0 02 A9
 037A 91 20 D2 FF A0 00 B1 1F
 0382 29 7F AA B1 1F 45 21 10
 038A 0B B1 1F 85 21 29 80 49
 0392 92 20 D2 FF 8A C9 20 B0
 039A 04 09 40 D0 0E C9 40 90
 03A2 0A C9 60 B0 04 09 80 D0
 03AA 02 49 C0 20 D2 FF C8 C0
 03B2 28 90 CB A5 1F 69 27 85
 03BA 1F 90 02 E6 20 C6 22 D0
 03C2 A6 A9 0D 20 D2 FF 4C CC
 03CA FF 72 21 61 3F 7F 76 57
```

Program 2.

```
.M 033A 03CB
SYS 826
 033A 78 A9 03 8D 1A 02 A9 47
 0342 85 90 90 58 60 AD 03 02
 034A C9 45 D0 03 20 54 03 4C
 0352 85 E6 A9 80 85 20 A9 00
 035A 85 1F A9 04 8D 64 02 85
 0362 F1 20 BA F0 20 32 F1 A9
 036A 19 85 22 A9 0D 85 21 20
 0372 D2 FF A9 11 AE 4C E8 E0
 037A 0C D0 02 A9 91 20 D2 FF
 0382 A0 00 B1 1F 29 7F AA B1
 038A 1F 45 21 10 0B B1 1F 85
 0392 21 29 80 49 92 20 D2 FF
 039A 8A C9 20 B0 04 09 40 D0
 03A2 0E C9 40 90 0A C9 60 B0
 03AA 04 09 80 D0 02 49 C0 20
 03B2 D2 FF C8 C0 28 90 CB A5
 03BA 1F 69 27 85 1F 90 02 E6
 03C2 20 C6 22 D0 A6 A9 0D 20
 03CA D2 FF 4C CC FF 67 54 00
```

work with his AXIOM EX801 printer and has added some new commands. Less than key ‹ prints the screen. Left arrow stops printing. Greater than symbol › deactivates Keyprint. It may also be useful to owners of printers other than Commodore's.

7. Melvin Field's version of Keyprint is for the 4.0 ROMs. Use SYS 634 to activate Keyprint. To access the screendump, use the backslash or SYS 657. He offers some alternatives to the use of the backslash. The direct mode statement FOR I = 1 TO

Program 3.

```
1 REM KEYPRINT LOADER PROGRAM
2 REM BY DAVID SWAIM
3 REM 2631 CALLAWAY RD
4 REM MARIETTA, GA  30060
5 REM THE FOLLOWING DATA IS THE DECIMAL
     EQUIVALENT OF
6 REM THE HEX LISTING BY ERIC BRANDON
7 REM COMPUTE! MARCH 1981 PAGE 92
10 DATA 120,169,3,141,26,2,169,71,141,25
    ,2,88,96,173,3,2,201,69
20 DATA 208,3,32,84,3,76,133,230,169,128
    ,133,32,169,0,133,31,169,4,141
30 DATA 100,2,133,241,32,186,240,32,50,2
    41,169,25,133,34,169,13,133,33
40 DATA 32,210,255,169,17,174,76,232,224
    ,12,208,2,169,145,32,210,255
50 DATA 160,0,177,31,41,127,170,177,31,6
    9,33,16,11,177,31,133,33,41,128
60 DATA 73,146,32,210,255,138,201,32,176
    ,4,9,64,208,14,201,64,144,10
70 DATA 201,96,176,4,9,128,208,2,73,192,
    32,210,255,200,192,40,144,203
80 DATA 165,31,105,39,133,31,144,2,230,3
    2,198,34,208,166,169,13,32,210
90 DATA 255,76,204,255,103,84,0,0
95 REM
96 REM POKE THE ML PROGRAM BEGINNING AT
    LOCATION 826 ( 033A )
99 REM
100 FOR I=826 TO 978
110 READ X
120 POKE I,X
130 NEXT I
140 END
```

Program 4.

.M 027A 030B
SYS 634

```
027A 78 A9 02 85 91 A9 85 85
0282 90 58 60 A5 97 C9 45 D0
028A 03 20 91 02 4C 55 E4 A9
0292 80 85 20 A9 00 85 1F A9
029A 04 85 B0 85 D4 20 D5 F0
02A2 20 48 F1 A9 19 85 22 A9
02AA 0D 85 21 20 D2 FF A0 11
02B2 AE 4C E8 E0 0C D0 02 A9
02BA 91 20 D2 FF A0 00 B1 1F
02C2 29 7F AA B1 1F 45 21 10
02CA 0B B1 1F 85 21 29 80 49
02D2 92 20 66 F2 8A C9 20 B0
02DA 04 09 40 D0 0E C9 40 90
02E2 0A C9 60 B0 04 09 80 D0
02EA 02 49 C0 20 D2 FF C8 C0
02F2 28 90 CB A5 1F 69 27 85
02FA 1F 90 02 E6 20 C6 22 D0
0302 A6 A9 0D 20 D2 FF 4C CC
030A FF 00 00 00 00 00 00 00
```

Program 5.

.M 033A 03CB
SYS 826

```
033A 78 A9 03 85 91 A9 45 85
0342 90 58 60 A5 97 C9 06 D0
034A 03 20 51 03 4C 55 E4 A9
0352 80 85 20 A9 00 85 1F A9
035A 04 85 B0 85 D4 20 D5 F0
0362 20 43 F1 A9 19 85 22 A9
036A 0D 85 21 20 D2 FF A9 11
0372 AE 4C E8 E0 0C D0 02 A9
037A 91 20 D2 FF A0 00 B1 1F
0382 29 7F AA B1 1F 45 21 10
038A 0B B1 1F 85 21 29 80 49
0392 92 20 D2 FF 8A C9 20 B0
039A 04 09 40 D0 0E C9 40 90
03A2 0A C9 60 B0 04 09 80 D0
03AA 02 49 C0 20 D2 FF C8 C0
03B2 28 90 CB A5 1F 69 27 85
03BA 1F 90 02 E6 20 C6 22 D0
03C2 A6 A9 0D 20 D2 FF 4C CC
03CA FF 72 21 61 3F 00 00 00
```

Program 6.

.M 033A 03ED
SYS 826

```
033A 78 A9 03 85 91 A9 45 85
0342 90 58 60 A5 97 C9 05 F0
034A 11 C9 0C F0 03 4C 2E E6
0352 A9 E6 85 91 A9 2E 85 90
035A D0 F3 20 62 03 4C 2E E6
0362 A9 80 85 20 A9 00 85 1F
036A A9 04 85 B0 85 D4 20 BA
0372 F0 20 2D F1 A9 19 85 22
037A A9 0A 85 21 20 D2 FF A9
0382 00 AE 4C E8 E0 0C D0 02
038A A9 00 20 D2 FF A0 00 B1
0392 1F 29 7F C9 1F F0 4C AA
039A B1 1F 45 21 10 0B B1 1F
03A2 85 21 29 80 49 92 20 D2
03AA FF 8A C9 20 B0 04 09 40
03B2 D0 1B C9 40 90 17 C9 60
03BA B0 11 AD 4C E8 C9 0C F0
03C2 06 8A 09 20 4C CF 03 09
03CA 80 D0 02 49 C0 20 D2 FF
03D2 C8 C0 28 90 BA A5 1F 69
03DA 27 85 1F 90 02 E6 20 C6
03E2 22 D0 95 A9 0D 20 D2 FF
03EA 4C CC FF EA EA EA EA EA
```

1000:PRINT PEEK(151):NEXT will print the coordinate code of the key you want to use. If you POKE 648,x then the key that corresponds to the coordinate code x will dump the screen. He suggests using the REV/OFF key, since it won't print any-

Program 7.

```
.M 027A 030B
SYS 634

027A 78 A9 02 85 91 A9 85 85
0282 90 58 60 A5 97 C9 45 D0
028A 03 20 91 02 4C 55 E4 A9
0292 80 85 20 A9 00 85 1F A9
029A 04 85 B0 85 D4 20 D5 F0
02A2 20 66 F2 A9 19 85 22 A9
02AA 0A 85 21 20 43 F1 A9 11
02B2 AE 4C E8 E0 0C D0 02 A9
02BA 91 20 66 F2 A0 00 B1 1F
02C2 29 7F AA B1 1F 45 21 10
02CA 0B B1 1F 85 21 29 80 49
02D2 92 20 66 F2 8A C9 20 B0
02DA 04 09 40 D0 0E C9 40 90
02E2 0A C9 60 B0 04 09 80 D0
02EA 02 49 C0 20 66 F2 C8 C0
02F2 29 90 CB A5 1F 69 27 85
02FA 1F 90 02 E6 20 C6 22 D0
0302 A6 A9 0D 20 66 F2 4C CC
030A FF 20 20 20 20 20 20 20
```

Program 8.

```
.M 033A 03FF
SYS 826
033A 78 AC 1A 02 20 8D 1F C0
0342 E6 D0 03 20 98 1F 58 60
034A AD 03 02 C9 45 F0 07 C9
0352 0F D0 06 4C CF 03 20 5E
035A 03 4C 85 E6 A9 80 85 72
0362 A9 00 85 71 20 78 1F A9
036A 19 85 74 9A 0D 85 73 20
0372 30 F2 A9 11 AE 4C E8 E0
037A 0C D0 02 A9 91 20 30 F2
0382 A0 00 B1 71 29 7F AA B1
038A 71 45 73 10 0B B1 71 85
0392 73 29 80 49 92 20 30 F2
039A 8A C9 20 B0 04 09 40 D0
03A2 0E C9 40 90 0A C9 60 B0
03AA 04 09 80 D0 02 49 C0 20
03B2 30 F2 C8 C0 28 90 CB A5
03BA 71 69 27 85 71 90 02 E6
03C2 72 C6 74 D0 A6 A9 0D 20
03CA 30 F2 4C 7D F2 A2 08 20
03D2 DF 03 4C 85 E6 A2 10 20
03DA 78 1F 4C B0 C5 A0 08 8C
03E2 0D 02 BD EE 03 99 0E 02
03EA CA 88 D0 F6 00 93 53 59
03F2 53 39 38 33 0D 93 53 59
03FA 53 39 37 32 0D 00 00 00
```

8. J. Michael McCormick has enhanced Eric Brandon's version of Keyprint for the Original ROMs. The at-symbol "@" will automatically list a BASIC program onto the printer.

9. This is Jerry Levitt's conversion of Keyprint for

Program 9.

```
.M 033A 03E0
SYS 826

033A 78 A9 03 85 91 A9 45 85
0342 90 58 60 A5 97 C9 45 D0
034A 03 20 51 03 4C 55 E4 A9
0352 00 85 D9 A9 01 85 D2 A2
035A 04 86 D4 A4 FF 84 D3 20
0362 63 F5 A2 01 20 FE F7 A9
036A 80 85 20 A9 00 85 1F A9
0372 19 85 22 A9 0D 85 21 20
037A D2 FF A9 11 AE 4C E8 E0
0382 0C D0 02 A9 91 20 D2 FF
038A A0 00 B1 1F 29 7F AA B1
0392 1F 45 21 10 0B B1 1F 85
039A 21 29 80 49 92 20 D2 FF
03A2 8A C9 20 B0 04 09 40 D0
03AA 0E C9 40 90 0A C9 60 B0
03B2 04 09 80 D0 02 49 C0 20
03BA D2 FF C8 C0 28 90 CB A5
03C2 1F 69 27 85 1F 90 02 E6
03CA 20 C6 22 D0 A6 A9 0D 20
03D2 D2 FF 20 CC FF A9 01 4C
03DA E2 F2 20 20 D2 FF 88 D0
```

Program 10a.

```
10 DATA120,169,3,133,145,169,69,133
20 DATA144,88,96,165,151,201,69,208
30 DATA3,32,81,3,76,46,230,169
40 DATA128,133,32,169,0,133,31,169
50 DATA4,133,176,133,212,32,186,240
60 DATA32,45,241,169,25,133,34,169
70 DATA13,133,33,32,210,255,169,17
80 DATA174,76,232,224,12,208,2,169
90 DATA145,32,210,255,160,0,177,31
100 DATA41,127,170,177,31,69,33,16
110 DATA11,177,31,133,33,41,128,73
120 DATA146,32,210,255,138,201,32,176~
~
130 DATA4,9,64,208,14,201,64,144
140 DATA10,201,96,176,4,9,128,208
150 DATA2,73,192,32,210,255,200,192
160 DATA40,144,203,165,31,105,39,133
170 DATA31,144,2,230,32,198,34,208
180 DATA166,169,13,32,210,255,76,204
190 DATA255,114,33,97,63,127,118,87
200 FOR I=826 TO 997
210 READV:POKEI,V
220 NEXTI
225 PRINT"{CLEAR}{10 DOWN}"
230 PRINT"TYPE SYS 826 TO ACTIVATE"
240 PRINT"{REV}{03 DOWN}THEN \ PRINTS~
   ~S FROM SCREEN EXACTLY!!!!"
250 END
```

the 4.0 ROMs. He gives some comments on customizing it. Keyprint uses logical file #1. To change this, POKE the new logical file number into memory locations 854, 869, and 984. POKE any different device number into 858. To change the key that dumps the screen, use the same procedure as in version **6.**

Program 10b.

```
10 DATA120,169,2,133,145,169,133,133
20 DATA144,88,96,165,151,201,69,208
30 DATA3,32,145,2,76,46,230,169
40 DATA128,133,32,169,0,133,31,169
50 DATA4,133,176,133,212,32,186,240
60 DATA32,45,241,169,25,133,34,169
70 DATA13,133,33,32,210,255,169,17
80 DATA174,76,232,224,12,208,2,169
90 DATA145,32,210,255,160,0,177,31
100 DATA41,127,170,177,31,69,33,16
110 DATA11,177,31,133,33,41,128,73
120 DATA146,32,210,255,138,201,32,176~
~
130 DATA4,9,64,208,14,201,64,144
140 DATA10,201,96,176,4,9,128,208
150 DATA2,73,192,32,210,255,200,192
160 DATA40,144,203,165,31,105,39,133
170 DATA31,144,2,230,32,198,34,208
180 DATA166,169,13,32,210,255,76,204
190 DATA255,114,33,97,63,127,118,87
200 FOR I=634 TO 785
210 READV:POKEI,V
220 NEXTI
225 PRINT"{CLEAR}{10 DOWN}"
230 PRINT"TYPE SYS 634 TO ACTIVATE"
240 PRINT"{REV}{03 DOWN}THEN \ PRINTS~
   ~S FROM SCREEN EXACTLY!!!!"
250 END
```

Program 11a. (4.0C)
.M 033A 03CB
SYS 826

```
033a 78 a9 03 85 91 a9 45 85
0342 90 58 60 a5 97 c9 45 d0
034a 03 20 51 03 4c 55 e4 a9
0352 80 85 20 a9 00 85 1f a9
035a 04 85 b0 85 d4 20 d5 f0
0362 20 48 f1 a9 19 85 22 a9
036a 0d 85 21 20 d2 ff a9 11
0372 ae 4c e8 e0 0c d0 02 a9
037a 91 20 d2 ff a0 00 b1 1f
0382 29 7f aa b1 1f 45 21 10
038a 0b b1 1f 85 21 29 80 49
0392 92 20 d2 ff 8a c9 20 b0
039a 04 09 40 d0 0e c9 40 90
03a2 0a c9 60 b0 04 09 80 d0
03aa 02 49 c0 20 d2 ff c8 c0
03b2 28 90 cb a5 1f 69 27 85
03ba 1f 90 02 e6 20 c6 22 d0
03c2 a6 a9 0d 20 d2 ff 4c cc
03ca ff 00 00 00 00 00 00 00
```

10. Timothy Dailey has given us two BASIC loader programs for Keyprint. He has also moved Keyprint to the first cassette buffer. You use SYS 634 to activate the latter version, and SYS 657 to dump the screen independently of Keyprint.

11. Joseph Holmes has supplied 4.0 versions of Keyprint for tape systems (4.0C), disk (4.0D), or for use on an 80-column CBM (80D).

Program 11b. (4.0D)
.M 027A, 030B
SYS 634

```
027a 78 a9 02 85 91 a9 85 85
0282 90 58 60 a5 97 c9 45 d0
028a 03 20 91 02 4c 55 e4 a9
0292 80 85 20 a9 00 85 1f a9
029a 04 85 b0 85 d4 20 d5 f0
02a2 20 48 f1 a9 19 85 22 a9
02aa 0d 85 21 20 d2 ff a9 11
02b2 ae 4c e8 e0 0c d0 02 a9
02ba 91 20 d2 ff a0 00 b1 1f
02c2 29 7f aa b1 1f 45 21 10
02ca 0b b1 1f 85 21 29 80 49
02d2 92 20 d2 ff 8a c9 20 b0
02da 04 09 40 d0 0e c9 40 90
02e2 0a c9 60 b0 04 09 80 d0
02ea 02 49 c0 20 d2 ff c8 c0
02f2 28 90 cb a5 1f 69 27 85
02fa 1f 90 02 e6 20 c6 22 d0
0302 a6 a9 0d 20 d2 ff 4c cc
030a ff 00 00 00 00 00 00 00
```

Program 11c. (80D)
.M 027A, 030B
SYS 634

```
027a 78 a9 02 85 91 a9 85 85
0282 90 58 60 a5 97 c9 dc d0
028a 03 20 91 02 4c 55 e4 a9
0292 80 85 20 a9 00 85 1f a9
029a 04 85 b0 85 d4 20 d5 f0
02a2 20 48 f1 a9 19 85 22 a9
02aa 0d 85 21 20 d2 ff a9 11
02b2 ae 4c e8 e0 0c d0 02 a9
02ba 91 20 d2 ff a0 00 b1 1f
02c2 29 7f aa b1 1f 45 21 10
02ca 0b b1 1f 85 21 29 80 49
02d2 92 20 d2 ff 8a c9 20 b0
02da 04 09 40 d0 0e c9 40 90
02e2 0a c9 60 b0 04 09 80 d0
02ea 02 49 c0 20 d2 ff c8 c0
02f2 50 90 cb a5 1f 69 4f 85
02fa 1f 90 02 e6 20 c6 22 d0
0302 a6 a9 0d 20 d2 ff 4c cc
030a ff 00 00 00 00 00 00 00
```

©

For PET/CBM BASICs 4.0 or Upgrade (3.0), 40 or 80 column screens, and disk drive. This short routine shows an easy way to transfer screen images to disk and back to the screen, in BASIC.

Screen Saver

David Wine
Philadelphia

This two-part program Screen Saver will SAVE the screen on your PET or CBM to disk, and then LOAD it back. The screen on your computer is mapped onto a continuous chunk of memory. Lines 230-240 trick BASIC into thinking the current program in memory starts at the top left corner of the screen and ends at the bottom right corner. There are two pointers in zero page that BASIC uses to tell it where the program is – the start-of-BASIC text pointer, and the start-of-variables (end of BASIC text) pointer. Screen Saver stores these and then points them to the start and end of the screen. On line 250, a simple SAVE stores the screen on disk. After this, the BASIC pointers used are restored their previous values by PEEKing them from the locations in the second cassette buffer where they were stored.

If you were to try to use variables to store the original pointers, you would run into trouble when it came time to put them back. BASIC gets very confused about variables when these pointers are redirected. Just for fun, try putting a STOP at line 245 and looking at a few variables.

The second part of the program LOADs the "screen" file back onto the screen. BASIC knows where to put it, because the first two bytes of a program file are written with the load address of the program in the computer's memory. Usually, this is 1024, but for the screen it's 32768. The end of a program on disk is followed by three zero bytes, so BASIC knows when to stop LOADing.

If line 310 were a normal LOAD, the program would stop execution right after LOADing the screen. In addition, out-of-memory errors would haunt you until you typed NEW. Luckily, there is a convenient BASIC firmware routine which LOADs the current file without disturbing BASIC's delicate pointers.

I can think of a couple of uses for screen saver. One is as part of an on-line help system. When the user asks for help, the current screen is saved and the help messages displayed. When he or she is done viewing the help screen, the old screen can be restored. Another use is for easy documentation of screen formats. Maybe even frame-by-frame animation? I would be interested in hearing of any other ideas.

```
100 REM TO USE THIS ON 40-COLUMN SCREENS, C
    HANGE LINE 240 TO:
101 REM POKE 42,232  AND POKE 43,131
102 REM
103 REM TO ADAPT TO UPGRADE (3.0) BASIC, CH
    ANGE LINE 310'S SYS TO 62242
104 REM
105 REM
200 REM SAVE IT
210 POKE900,PEEK(40):POKE901,PEEK(41)  : REM
     SAVE START BASIC TEXT
220 POKE902,PEEK(42):POKE903,PEEK(43)  : REM
     SAVE START VARIABLES
230 POKE40,0:POKE41,128                : REM
     POINT TO START SCREEN
240 POKE42,208:POKE43,135              : REM
     POINT TO END SCREEN
250 SAVE"@0:SCREEN",8
260 POKE40,PEEK(900):POKE41,PEEK(901)  : REM
     RESTORE POINTERS
270 POKE42,PEEK(902):POKE43,PEEK(903)
280 PRINTCHR$(147)
290 REM
300 REM LOAD IT
310 OPEN1,8,1,"0:SCREEN":SYS62294:CLOSE1
320 GETC$:IFC$=""THEN320
```

©

How to get 256 colors out of your Atari. Last month, this three-part series opened with a discussion of Atari Graphics. Part II examines techniques involving color indirection *and looks at the new GTIA chip in detail. If you have one of the older machines, your dealer should now have the new chip and can install it for you for about $60 (according to Atari). If your machine is still under warranty, the upgrade is free.*

Next month, this series concludes with several programs which put GTIA through its paces.

Part II:

Atari Video Graphics And The New GTIA

Craig Chamberlain
Birmingham, MI

Using Color Indirection

With color indirection, the number of different playfields is limited according to the number of bits per pixel, but the actual color/luminance of each playfield can be one of the 128 possibilities. The data bits are used as an index or offset into playfield color registers:

COLOR0 $02C4 708
playfield zero color register
COLOR1 $02C5 709
playfield one
COLOR2 $02C6 710
playfield two (used in modes 0 and 8)
COLOR3 $02C7 711
playfield three (used in color text modes)
COLOR4 $02C8 712
background color register

These playfield color registers use seven bits to select the color and luminance, as follows:

D7,D6,D5,D4	color
D3,D2,D1	luminance
D0	not used

BITS	VALUE	COLOR
0000	0	gray (no color)
0001	1	light orange
0010	2	orange
0011	3	red orange
0100	4	pink
0101	5	purple
0110	6	purple blue
0111	7	blue
1000	8	blue
1001	9	light blue
1010	10	turquoise
1011	11	blue green
1100	12	green
1101	13	yellow green
1110	14	orange green
1111	15	light orange

Atari BASIC allows you to select a playfield color to draw in by using the COLOR statement. The color register that corresponds to that playfield can be changed by using SETCOLOR.

Color indirection is a tool that should not be overlooked. It is possible to draw a detailed figure on the screen with one playfield, and then change the color of the entire figure with just one command. For example, a printed message can flash in colors to attract attention. A "glowing" effect can be created by rapidly changing the luminance of a playfield while maintaining the same color. Or, the playfield colors can all be set to the same color/luminance as the background. Figures drawn will not appear until the playfield color registers are changed. By changing the registers one at a time, an animation effect can be created. Color indirection may still not solve the problem of having many colors on the screen at the same time, but it does afford possibilities that otherwise would be difficult to achieve.

In special instances, playfield color registers can be changed during the horizontal blank, in which case all 128 color variations can be shown in one frame. This requires the use of machine language and still does not solve the problem of many colors on one scan line. Fortunately, experience has shown that, for many applications, three playfield colors will be sufficient.

Multiple Colors

Nevertheless, there are times when many colors would be desirable. This is where the GTIA steps in. It should now be apparent that 16 colors will require four bits per pixel. This is very expensive in terms of memory, so either pixel size or display memory will have to increase. Because ANTIC has a limit on how much memory it can access during one horizontal scan line, we have a limit on how much memory can be devoted to a screen. Therefore, resolution will have to suffer.

Before we see what the memory limit is, we should mention the two modes which are exceptions to the above rules. Three things distinguish modes zero and eight from the normal modes. Each pixel is a half color clock wide; a side effect of this is artifacting. The background color now becomes the border, and the main part of the screen is filled with playfield two. Finally, since the whole screen is now playfield two, the bit no longer tells which playfield to use, but which *luminance* to use.

MODE	BIT	LUMINANCE REGISTER
0,8	1	playfield one
0,8	2	playfield two (no image)

The color part of playfield one is ignored; only the luminance data is used. If the luminance values of playfields one and two are the same, the writing disappears. Modes zero and eight use this special "half color clock, one playfield color, two brightness" arrangement. Both modes have 320 distinct points of light horizontally and have single scan line resolution. The only difference between mode zero and mode eight is that the first is a text mode and the second is a direct mapping mode. Mode zero uses a character set and thereby saves memory; about 1K is required for this mode. Mode eight doesn't use a character set, and requires approximately 8K. That is our display memory limit. The Atari 400/800 is not capable of doing DMA to much more memory than the memory represented by one television frame.

Since the "half color clock, one color, two brightness" mode is used by graphics modes zero and eight, all the GTIA really does is provide three variations on this mode. They all use the maximum memory arrangement used by mode eight, so each of the three new modes requires 8K. All of the new modes use four bit pixels, so the horizontal resolution goes from 320 (half color clock) to 80 (two color clock, as in modes four and five). Therefore, the resolution for all three new modes is 80 by 192, for a total of 15360 points. One side effect of changing only the horizontal resolution is that the pixels are no longer square.

The ANTIC instruction register mode number for the maximum memory mode (the number you will find in the display list) is $0F, or decimal 15. It is important to understand that this number indicates not only mode eight, but also nine, ten, and eleven as well. In fact, the display list for any one of these modes is identical to the display list for any of the others.

Selecting Modes With PRIOR

How then does ANTIC know which of the four is the desired mode? The answer is that ANTIC neither knows nor cares; no matter which mode is being used, ANTIC still has to do the same work of fetching memory. It's the GTIA that processes the video signal; somehow the chip must be told which of the four modes is wanted. The GTIA hardware register PRIOR does exactly that.

GPRIOR	$026F	623	shadow
PRIOR	$D01B	53275	hardware

The two most significant bits (bits six and seven) of this register are the GTIA special mode select bits. Here's how they are set.

MODE	BITS	HEX	DECIMAL
8	00	00	0
9	01	40	64
10	10	80	128
11	11	C0	192

For example, it is possible to switch from any one of the four modes to another simply by changing the values of the two select bits.

Other bits in GPRIOR serve different functions, so care must be taken not to alter them. These other bits allow multi-color players (blending on overlap), set all missiles to the color of playfield three to form a fifth player, and establish player/missile and playfield priorities. See the *Hardware Manual* for further information.

Now that we know how the three new modes are similar, let's find out how they are different.

Mode 11 is the one luminance, 16 color mode. The overall luminance is set by the background color, which, for this mode, defaults to a luminance of six, rather than the usual zero. It is now easy to draw rather finely detailed shapes in several colors without having to fool around with the display list and machine code interrupt routines. The thing I am especially excited about is going to make Apple owners envious. The Apple has a 16 color mode with resolution of 40 by 48, called the "lo res" mode. The Atari now has a 16 color mode, but the resolution is eight times greater than the Apple's.

Sixteen colors do present a problem, though, since the GTIA has only four playfield color registers. Therefore, mode 11 does not allow color indirection. The color on the screen is determined directly by the bit data stored in memory, according to the chart given earlier in the section on color indirection. The values in the four color/luminance registers are ignored. Some may consider this a disadvantage, but there is a benefit too. Just as the playfield color registers are not used, neither are the player/missile color registers used, so by using players it is possible to have 21 colors on the screen at the same time, without using display list interrupts or other tricks.

Producing 256 Colors

Mode nine is the one color, 16 luminance mode. This mode will be used to create some excellent three dimensional effects and digitized pictures. The 16 luminances, when stacked vertically by the scan line with each line having the next brightest luminance, blend so well that it is very difficult to see the division from one to the other. The main color is set by the background color. Weird things happen when you change the luminance of the background. Another nice fact is that having 16 main colors with 16 luminance variations means that the Atari is capable of producing 256 colors.

One advanced application for mode nine is

NEW FROM

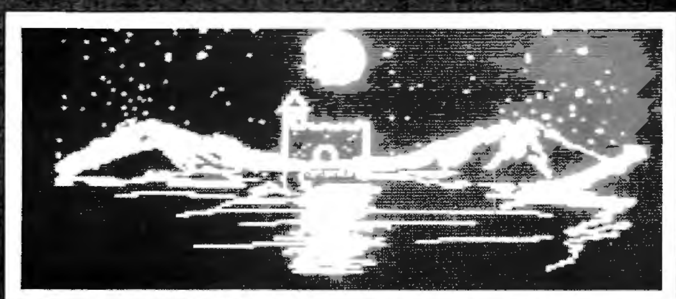
Synergistic Software

ACTION AND

ADVENTURE

PROBE ONE
THE TRANSMITTER

A new high-res adventure game filled with arcade action, color, and sound effects. You must battle guard droids while searching room to room in a remote research colony. The original game design has various skill levels and new game layout each time you play. Probe One uses paddle or joystick as well as keyboard commands............ $34.95



WARLOCK'S REVENGE

Two full disks of challenging high-res adventure. You lead a select group of adventurers on a quest to gather treasure and destroy the evil warlock, Oldorf. As you explore caverns and castles you use your group's various skills to gather riches and fight off creatures. Warlock's Revenge uses keyboard commands and you may save the game for later play....................................... $34.95

Both games require 40K, 800/400, disk, and ATARI BASIC.

TOLL FREE ORDER LINE — 800-426-6505

We accept checks, VISA, MASTER CHARGE, and C.O.D.
$2.00 postage and handling charge, except on prepaid orders.

Synergistic Software
830 N. Riverside Dr., Suite 201, Renton, WA 98055
(206) 226-3216

ATARI 800/400 are trademarks of ATARI, INC.

NOW AVAILABLE FOR THE APPLE II

Ali Baba and the forty thieves

By Stuart Smith

A fantasy role-playing adventure for Apple II and Atari Personal Computers.

Encounter sultans, thieves, fierce and friendly creatures as you guide your alter ego, Ali Baba, through the thieve's mountain den in an attempt to rescue the beautiful princess. Treasure, magic, and great danger await you! One or more human players can guide up to seventeen friendly characters through the many rooms, halls, and caves. Some characters wander around randomly, making each adventure a little different.

ALI BABA AND THE FORTY THIEVES is written in high resolution color graphics and includes music and sound effects. Adventures can be saved to disk and resumed at a later time.

Available for Apple II and Apple II Plus 48K or Atari 800 32K.

On diskette only — $32.95

FOR OUR COMPLETE LINE OF APPLE AND ATARI SOFTWARE PLEASE WRITE FOR OUR CATALOG

ASK FOR QUALITY SOFTWARE products at your favorite computer store. If necessary you may order directly from us. MasterCard and Visa cardholders may place orders by calling us at (213) 344-6599. Or mail your check or bankcard number to the address above. California residents add 6% sales tax. *Shipping Charges:* Within North America orders must include $1.50 for shipping and handling. Outside North America the charge for airmail shipping and handling is $5.00. Pay in U.S. currency.

QS QUALITY SOFTWARE

6660 Reseda Blvd., Suite 105, Reseda, CA 91335 (213) 344-6599 the display of digitized pictures. Digitization is a process by which a normal television picture, such as from a station or video recorder, can be analyzed and divided into different luminances. That information can be sent to the computer and stored on disk for later display. Mode nine, with 16 luminances and rather high resolution, is able to reproduce such pictures with impressive quality. Thus far we have seen only four digitized pictures. They were apparently made by some people at Atari, and two of the pictures were, uh, for mature viewers only. Standing from a short distance, however, it is very difficult to tell if any of these pictures is computer generated or not. I have never seen such quality on any other computer in the 400/800 price range without expensive additional equipment.

Mode ten is a cross between the other two modes; it allows eight colors plus the background, each with its own luminance as in the primary modes. Unlike the other two modes, this one allows color indirection, so it uses the playfield and player/missile registers for color/luminance information. This chart shows how data values correspond with playfield registers.

BITS	VALUE	REGISTER	PLAYFIELD
0000	0	704	PCOLR0
0001	1	705	PCOLR1
0010	2	706	PCOLR2
0011	3	707	PCOLR3
0100	4	708	COLOR0
0101	5	709	COLOR1
0110	6	710	COLOR2
0111	7	711	COLOR3
1000	8	712	COLOR4
1001	9	712	COLOR4
1010	10	712	COLOR4
1011	11	712	COLOR4
1100	12	708	COLOR0
1101	13	709	COLOR1
1110	14	710	COLOR2
1111	15	711	COLOR3

Only nine of the 16 possible data values correspond to different playfields. Data values greater than eight just repeat playfields. For some reason, the background color is no longer set by COLOR4, but instead by PCOLR0. The Atari BASIC statement SETCOLOR can't be used to change the player/missile color registers, so the equivalent POKE must be used. For any register, the data part of the POKE is the color choice number multiplied by 16, plus the luminance (refer to earlier chart).

The power of indirection is magnified when eight main drawing colors can be used. This mode is very useful for creating motion effects. With nine color/luminances and color indirection, mode ten may prove to be the most versatile of the three new modes.

Compatibility Between CTIA And GTIA

Remember that the GTIA only controls how the display is generated, so all programs written for the CTIA should run on a GTIA machine in the same way. There can be no such thing as incompatibility. We have, however, come across one discrepancy between the CTIA and GTIA. The video signal generated by the GTIA is shifted one half color clock, so colors produced by artifacting, such as in POOL 1.5 or Jawbreakers, will be different. That is just a minor visual difference; the important thing is that all software should be entirely compatible. Of course, you cannot expect a CTIA to generate these three new modes, but again the conflict is the display, not the program. In fact, I don't think it is even possible for the computer to tell whether it has a CTIA or GTIA in it.

Because of the half color clock shift, it is now possible for players and playfields to overlap perfectly, whereas with the CTIA they didn't.

There are some cases where software will not run on GTIA machines. This is due to the fact that some of the new computers with the GTIA also have a revised (no bugs) operating system in them. Atari has made very clear which memory locations and vectors are permanent and protected from any revisions. If a program does not run on a GTIA machine, it is the software's fault because illegal entry points were used.

One other conflict has appeared which really surprised me. We have discovered that a few programs written on CTIA machines carelessly set the GTIA special mode select bits of GPRIOR for no purpose. Since these two bits do nothing on the CTIA, there was no problem. But there was also no reason to involve them. When the same programs are run on GTIA, the accidental bit settings affect the display, even though modes nine, ten and eleven are not used. The function of those two bits has not been a secret. I figured out their function in July 1981, when I read the OS source listing before I bought my Atari 800. The *Hardware Manual* has described the three "new" modes in appendix H ever since the manual was released.

No Text Window

There is a difference between the normal modes and the three new modes – the three new ones do not allow split screen (text window at bottom) configurations. If you remember how modes eight and zero are related, you should understand why. The mode used in the text windows is mode zero, which follows the special "half color clock, one color, two luminances" arrangement. As stated above, having the mode select bits in GPRIOR set for a mode greater than eight causes mode zero to act funny. A split screen would only be possible if a display list interrupt were inserted just before the text window area. The interrupt routine would have to reset to zero the mode select bits in the hardware register PRIOR, not the shadow register. The hardware register will then be reset to the value of GPRIOR during the vertical blank service routine.

The three new modes seem to handle player/missile to playfield collisions a little differently. In modes zero and eight, a playfield two collision is flagged when a player or missile hits a pixel whose luminance is controlled by COLOR1 rather than the COLOR2 for the main playfield. From what I have been able to tell thus far, there is no kind of playfield collision at all in modes nine and eleven. Mode ten collisions work only for playfield colors that correspond to the usual playfield registers (COLOR1 through COLOR3). Also, the fact that the background in this mode is set by PCOLR0 affects the priority of players and playfields in some cases. In priority, mode ten playfield colors PCOLR0 through PCOLR3 behave like players.

The GTIA still allows only eight luminances on the normal modes.

All new Atari computers are being shipped with the GTIA at no extra cost. The CTIA is no longer being produced. The new machines with the GTIA have little yellow or white stickers that have the letter "G" on them. Those of us who have older machines with the CTIA can replace it with a GTIA. The part number is C014805.

If you want to do it yourself, it will be a simple matter to replace the CTIA. The CTIA is on the CPU card that plugs into the motherboard inside the Atari case. It's not soldered in, so the replacement operation should take only 30 minutes if you have taken your computer apart before. Instructions are supplied with the chip. In the meantime, if you don't have the GTIA, don't fret. It will be a while before much software requiring the chip is available.

Do You Already Have The GTIA?

If you want to quickly see if your computer has a GTIA, try this: POKE 623,64 (while in the default mode, zero).

If you have the GTIA, the screen will go black. Otherwise, there will be no change and you'll know you've got the CTIA. If you have the GTIA and want to see 16 colors, try this.

```
10 GRAPHICS 11
20 FOR K=0 TO 79
30 COLOR K
40 PLOT K,0
50 DRAWTO K,191
60 NEXT K
70 GOTO 70
```

©

Telecommunications:

Choosing A Modem: Part II

Michael E Day
Chief Engineer
Edge Technology
West Linn, OR

When considering the purchase of a modem, one important issue is compatibility – what other modems can you communicate with?

There are three major types of modems currently available to the general user. The 202 type, the 103, and a modem that has been enjoying a recent increase in popularity despite its high cost, the 212A.

The 202 modem has been around for some time. It has fallen out of favor recently due to the complexity of its operation. It requires some amount of computer control. Its main advantage is its speed of operation – up to 1200 bits per second (bps), which is equal to 120 characters per second. Because it has never received wide acceptance in the general user market, it also tends to cost far more than it should.

A Cassette Interface Is Practically A Modem

Of all the types of modems, the 202 has the potential for being the least expensive due to its minimal hardware required to convert the computer data to be sent over the phone lines. The greater degree of computer control over a 202 modem's operation also lowers its cost. It is actually possible to build a 202 for less than $25 in parts. This would translate to around $50 in high volume production, and would be even less if it were incorporated into a computer as a basic part of its design. In fact, many computers actually have a similar circuit in them already: the cassette interface.

Although the exact method used for placing data on a cassette varies between manufacturers, the basic conversion requirements are the same. In fact, with a little bit of work, it would be quite possible to convert many of the cassette interfaces to communicate over the phone line.

A disadvantage of the 202 modem (in addition to its requiring a high degree of computer control) is that it must be connected directly to the phone line. It is not feasible to acoustically connect the 202 type modem through the telephone handset. The distortion caused by the conversion to sound and back effectively limits the speed of communication. Because of the wideband transmission characteristics of the 202 type modem, this problem is made even worse.

Although it is *possible* to communicate acoustically with the 202 type modem, the distortion effects limit the communication speed to 330 bps or less. This makes it less efficient than the 103 type modem, which can also communicate at 300 baud acoustically, but does not have the extra computer control requirements.

The 103 modem, unlike the 202, requires little or no computer control. In its simplest form – as an acoustic coupler – no control at all is required. Placing the phone call and initiating the communications are all performed by the operator. The 103 is very attractive to the general user because little or no programming is needed.

The 103 could also be connected directly to the phone line, but, again, it has the advantage of requiring little or no control by the computer to operate it. The only real computer control that would be needed would be some software to generate a call automatically if that were wanted. Some modems do not allow automatic calling, while others provide for computer control, and other functions as well.

The Most Popular Model By Far

The 103 modem is the most popular modem available by quite a margin. This is a direct result of its simplicity and ease of use. The 103 has an upper speed limit of operation of 300 bps normally. Some direct connect types, however, can go up to 600 bps. The typical cost of the 103 is between $100 to $200, depending on quality and complexity. Some of the better direct connect 103's can cost $300 to $400 and offer many functions as well as increased reliability.

The 103 cannot communicate with the 202 type. The two modems do not use the same communications scheme. The 202 is generally a half duplex while the 103 is a full duplex modem. While the 202 can communicate in full duplex, it must have two phone lines to do so (one for each direction of communication). The 103 provides the full duplex communications over a single phone. However, it needs more circuitry to perform full duplex, and this is the reason for its increased cost over the 202.

A More Expensive Design

The third main modem design is the 212A. This modem is more expensive than the others, but it combines the functionality of the 103 with the speed capability of the 202. The 212A cannot be acoustically connected to the phone line through the telephone handset, but it does not require any special computer control (unless that's desired).

The 212A cannot communicate with the 202. It does, however, have a mode of operation which allows it to communicate with the 103 design. In fact, it is two modems in one: the 103 communications method is entirely different from the 212A method. This ability to communicate with the 103 type modem as well as with another 212A is part of the reason for its popularity in spite of its high cost.

The 212A can cost from $500 to $1000. This higher cost is due primarily to the method by which it communicates. It is in actuality a computer and modem combination. This is necessary because the 212A not only converts the data sent to it into the audio signals which go over the phone line, but it also must change the way the data is transmitted. The 212A internally transforms asynchronous data that it receives into a synchronous data stream.

The 103 and 202 modems use a data conversion scheme called FSK (*Frequency Shift Keying*). The 212A, however, uses a different method called PSK (*Phase Shift Keying*). FSK does not require any special handling other than checking that the maximum speed of operation is not exceeded.

The PSK method, however, requires that the data to be sent be synchronized to the audio signal to be sent. By doing this, the 212A is able to make more effective use of the phone line and to allow for true full duplex communications at 1200 bps. The limitation here is that the communications must occur at exactly 1200 bps due to the conversion requirements.

When a 212A is operating in the 103 mode, it reverts to the FSK method to communicate with the 103 design. When placing a call with the 212A, the user must tell the modem which mode it is to use (103 or 212A). When the 212A is receiving a call, however, it will automatically determine which type of modem it is communicating with, switch to that mode (unless it is told otherwise), and tell the user which mode it has selected.

The 212A cannot be acoustically connected to the phone line for the same reason that the 202 type cannot – too much distortion is caused by the conversion telephone handset and by sound conversion. Such a connection would probably be possible if a condenser microphone were installed in the phone, but, since no one is manufacturing acoustically coupled 212A modems, there is not much point in trying this.

Racal Vadic's 3400

There is one manufacturer which has attempted to solve this problem, however. Racal Vadic builds a modem they call a 3400 series. This modem, while able to communicate acoustically over a telephone handset, can only do so to another 3400. The 3400 does this by using yet another communications scheme which is not compatible with any of the previously mentioned modems. The 3400 uses a PSK type of transmission (like the 212A), but it uses a specially designed structure which minimizes the distortion caused by the telephone handset.

The 3400 can also be directly connected to the phone line, and some models, able to communicate with the 103, also have a mode which allows them to communicate with 212A's. The 3400 is not in very widespread use at this time, which might be due to its incompatibility with the 212A.

When deciding which modem to buy, it might be best not to consider the 202 unless you have a specific need for it. It is not in general use and can only communicate with another 202. The 103 is the most common and least expensive, but the 212A, while it costs more, has greater functionality. For portability, the 103 acoustic coupler is probably the best choice since it can communicate with either another 103 or with a 212A. If the portability is unnecessary, and it is acceptable to have the modem directly connected to the phone line, then it becomes simply a matter of deciding how much you are willing to spend for functionality when deciding between a 103 or a 212A design. ©

For the TRS-80 Color Computer, 16K, with either Extended or Non-Extended BASIC, disk or tape – this program will show you the effects of home energy conservation.

Energy Monitor

Linton S. Chastain
Greensboro, NC

Since energy costs have been of major concern to many people in the past few years, here's a BASIC program that has helped me evaluate my energy costs and consumption. The program helps you to determine if those conservation changes that you may have made over the past year are meaningful.

The program keeps track of energy cost and consumption. The first thing you will probably notice is that when energy consumption has remained the same from year to year the cost of that energy has increased. This awareness is enough in itself to inspire conservation measures. Major changes to a home (storm windows, weather stripping, more insulation) can be validated with this program. Pick periods that have the same number of days with similar heating or cooling.

The program was originally written on a 16K Radio Shack Color Computer without Extended BASIC, and it used a cassette recorder for DATA storage. Energy Monitor now uses a disk drive. However, the program should work with few changes on any computer that has at least 8K of user memory available and uses Microsoft BASIC.

If you don't use a disk drive, I will point to what modifications you will have to make to use a cassette recorder. These modifications will be directed toward the Color Computer with at least 8K of user memory with or without Extended BASIC. If you have a different computer, please check your manual on how to store data on cassette.

Here are the changes for those who have cassette recorders:

For Cassette Users.

```
120 PRINT"6-READ OLD MASTER FILE":PRINT:"FR
    OM CASSETTE"
140 PRINT"8-WRITE NEW MASTER FILE":PRINT:"T
    O CASSETTE"
650 PRINT"PRESS THE RECORD AND":PRINT"PLAY
    KEY ON CASSETTE"
670 OPEN"O",#-1,T$:PRINT#-1,K:K=1:L=N
700 FORJ=KTOL:PRINT#-1,D$(J),A(J),B(J),C(J)
    ,D(J),E(J),F(J),G(J):PRINTJ:NEXTJ
710 CLOSE
720 PRINT:PRINT"PRESS THE CASSTTE":PRINT"ST
    OP KEY"
750 PRINT"PRESS THE PLAY KEY":PRINT"ON THE
    CASSETTE"
790 OPEN"I",#-1,T$:PRINT"READING FILE: "T$:
    INPUT#-1,N
820 FORJ=1TON:INPUT#-1,D$(J),A(J),B(J),C(J)
    ,D(J),E(J),F(J),G(J):PRINTJ:NEXTJ
840 CLOSE
850 PRINT:PRINT"PRESS THE CASSTTE":PRINT"ST
    OP KEY"
```

```
10 REM UTILITIES
20 REM BY STEVE CHASTAIN 1/31/81
30 CLEAR 200
40 MW=20:MR=20:N=0
50 DIMD$(MR),A(MR),B(MR),C(MR),D(MR),E(MR)
   ,F(MR),G(MR),W(MR),WW(MR),X(MR),XX
   (MR),Y(MR),YY(MR),Z(MR),U(MR)
60 R=0:S=0:W=0:WW=0:X=0:Y=0:YY=0:Z=0
65 CLS:PRINT"UTILITIES":PRINT:PRINT"COMMAN
   D LIST # 1"
70 PRINT"1-DISPLAY WATER COST AND UNITS"
80 PRINT"2-DISPLAY GAS COST AND UNITS"
90 PRINT"3-DISPLAY ELECTRIC COST AND"
95 PRINT"UNITS"
100 PRINT"4-DISPLAY TELEPHONE COST"
110 PRINT"5-DISPLAY UTILITIES COSTS AND"
115 PRINT"UNITS"
120 PRINT"6-READ OLD MASTER FILE FROM DISK"
130 PRINT"7-INPUT NEW DATA"
140 PRINT"8-WRITE NEW MASTER FILE TO DISK"
150 PRINT"9-TERMINATE PROGRAM":PRINT
160 INPUT"ENTER COMMAND BY NUMBER";R:IFR<1
    OR R>9 THEN 60
170 ON R GOSUB 970,1170,1370,1570,520,740,1
    80,630,870:GOTO60
180 IFN=MR THEN510
190 PRINT:PRINT"ENTER THE FOLLOWING DATA AS
     REQUESTED"
200 PRINT"-DATE(1/31/81)"
210 PRINT"-WATER COST"
220 PRINT"-WATER UNITS"
230 PRINT"-GAS COST"
240 PRINT"-GAS UNITS"
250 PRINT"-ELECTRIC COST"
260 PRINT"-ELECTRIC UNITS"
270 PRINT"-TELEPHONE COST"
280 N=N+1:PRINT:INPUT"DATE";R$:R$=LEFT$(R$,
    8);D$(N)=R$
290 INPUT"WATER COST";R:A(N)=R;IFR<0 THEN30
    0
300 INPUT"WATER UNITS";R:B(N)=R;IFR<0 THEN3
    10
310 INPUT"GAS COST";R:C(N)=R;IFR<0 THEN320
320 INPUT"GAS UNITS";R:D(N)=R:IFR<0 THEN330
330 INPUT"ELECTRIC COST";R:E(N)=R;IFR<0 THE
    N340
340 INPUT"LECTRIC UNITS";R:F(N)=R;IFR<0 THE
    N350
350 INPUT"TELEPHONE COST";R:G(N)=R;IFR<0 TH
    EN 350
360 PRINT:PRINTTAB(1);"CHECK";TAB(7);"DATE:
    ";D$(N)
370 PRINTTAB(7);"WATER COST:";A(N)
380 PRINTTAB(7);"WATER UNITS:";B(N)
390 PRINTTAB(7);"GAS COST:";C(N)
400 PRINTTAB(7);"GAS UNITS:";D(N)
410 PRINTTAB(7);"ELECTRIC COST:";E(N)
420 PRINTTAB(7);"ELECTRIC UNITS:";F(N)
430 PRINTTAB(7);"TELEPHONE COST:";G(N)
```

```
440 PRINT:PRINTTAB(7)"-IS INPUT O.K.?-":PRI
    NT
450 INPUT"(Y=YES,N=NO,F=YES AND FINISHED)";
    R$:R$=LEFT$(R$,1)
460 IFR$="N" THEN N=N-1:PRINT:PRINT"REDO LA
    ST DATA":GOTO280
470 IFR$="F" THEN RETURN
480 IFR$<>"Y" THEN440
490 IFN=MR THEN510
500 GOTO280
510 PRINT:PRINT"*** NO MORE DATA ALLOWED***
    ":GOSUB940:RETURN
520 IFN<1 THEN PRINT:PRINT"*** NOT ENOUGH D
    ATA***":GOSUB940:RETURN
530 FORJ=1 TO N
540 W=W+W(J):WW=WW+WW(J):X=S+X(J):XX=XX+XX(
    J):Y=Y+Y(J):YY=YY+YY(J);Z=Z+Z(J)
550 U=(W+X+Y+Z)
560 NEXTJ:K=-1:L=0
570 K=K+2:L=L+2:IFL>N THEN L=N
580 CLS:FORJ=K TO L:PRINT@96,"GAS";TAB(14)"
    COSTS";TAB(26)"UNITS":PRINT@64,
    "WATER";TAB(14);W;TAB(26);WW:PRINT@96,"
    GAS";TAB(14);X;TAB(26);XX:PRINT@128,
    "ELECTRIC";TAB(14);Y;TAB(26);YY
585 PRINT]160,"TELEPHONE";TAB(14);Z
590 FORQ=0 TO 31:PRINTCHR$(45);:NEXT
595 PRINT]224,"TOTALS";TAB(13);U
600 NEXTJ:PRINT
610 IFL<=N THENPRINT"HIT ANY KEY FOR COMMAN
    D MODE":GOSUB950:RETURN
620 PRINT"HIT ANY KEY TO CONTINUE":GOSUB950
    :GOTO570
630 IFN<1 THEN PRINT:PRINT"*** NO DATA TO W
    RITE ***":GOSUB940:RETURN
640 R$="WRITING":PRINT
660 INPUT"NAME FOR FILE";T$:K=N:IFN>MW THEN
    K=MW
670 OPEN"O",#1,T$:WRITE#1,K:K=1:L=N
680 IFN>MW THENK=N-MW+1:PRINT"-ONLY LAST";M
    W"VALUES WILL BE WRITTEN"
690 PRINT"WRITING FILE:";T$:PRINT"    RECOR
    DS #";
700 FORJ=K TO L:WRITE#1,D$(J),A(J),B(J),C(J
    ),D(J),E(J),F(J),F(J),G(J):PRINTJ:
    NEXTJ
710 CLOSE#1
730 PRINT"PRESS THE KEYBOARD'S ENTER KEY.":
    GOSUB950:RETURN
740 R$="READING":PRINT
780 INPUT"NAME OF FILE ";T$
790 OPEN"I",#1,T$:PRINT"READING FILE: ";T$:
    INPUT#1,N
800 IFN>MR THEN PRINT"*** TOO MANY FILES ON
     DISK ***":END
810 PRINT"READING RECORDS # ";
820 FORJ=1 TO N:INPUT#1,D$(J),B(J),C(J),D(J
    ),E(J),F(J),G(J):PRINTJ:NEXTJ
830 PRINTN ;" DATA RECORDS READ"
840 CLOSE#1
860 PRINT"PRESS THE KEYBOARD'S ENTER KEY.":
    GOSUB950:RETURN
870 END
940 FORQ=1 TO 1000:NEXTQ:RETURN
950 B$="":R$=INKEY$-IFR$=B$ THEN950
960 RETURN
970 CLS:PRINT"WATER":PRINT:PRINT"COMMAND LI
    ST # 2"
980 PRINT"1-DISPLAY WATER"
990 PRINT"2-RETURN TO COMMAND LIST # 1"
1000 INPUT"ENTER COMMAND BY NUMBER";R:IFR<1 ~
     OR R>2 THEN970
1010 ON R GOSUB 1020,1110:GOTO970
1020 IFN<1 THEN PRINT:PRINT"*** NOT ENOUGH D
     ATA ***":GOSUB1140:RETURN
1030 FORJ=1 TO N
1040 R=A(J):S=B(J):W(J)=R:WW(J)=S
1050 W=W(J):WW=WW(J)
1060 NEXTJ:K=-3:L=0
1070 K=K+4:L=L+4:IFL>N THENL=N
1080 CLS:PRINT"DATE";TAB(14);"COST";TAB(26);
     "UNITS"
1085 FORJ=K TO L:PRINTD$(J);TAB(14);A(J);TAB
     (26);B(J);NEXTJ:PRINT
1090 IFL=N THEN PRINT"HIT ANY KEY FOR COMMAN
     D MODE ":GOSUB1150:RETURN
1100 PRINT"HIT ANY KEY TO CONTINUE":GOSUB115
     0:GOTO1070
1110 GOTO60
1120 R$=INKEY$:IFR$=B$ THEN 1120
1130 RETURN
1140 FORQ=1 TO 1000:NEXTQ:RETURN
1150 B$="":R$=INKEY$:IFR$=B$ THEN 1150
1160 RETURN
1170 CLS:PRINT"GAS":PRINT:PRINT"COMMAND LIST
      # 3"
1180 PRINT"1-DISPLAY GAS"
1190 PRINT"2-RETURN TO COMMAND LIST # 1"
1200 INPUT"ENTER COMMAND BY NUMBER";R:IFR<1 ~
     OR R>2 THEN 1170
1210 ON R GOSUB 1220,1310:GOTO1170
1220 IFN<1 THEN PRINT;PRINT"*** NOT ENOUGH D
     ATA ***":GOSUB1340:RETURN
1230 FORJ=1 TO N
1240 R=C(J):S=D(J):X(J)=R:XX(J)=S
1250 X=X(J):XX=XX(J)
1260 NEXT J:K=-3:L=0
1270 K=K+4:L=L+4:IFL>N THENL=N
1280 CLS:PRINT"DATE";TAB(14);"COST";TAB(26);
     "UNITS"
1285 FORJ=K TO L:0PRINTD$(J);TAB(14);C(J);TA
     B(26);D(J):NEXTJ:PRINT
1290 IFL=N THEN PRINT"HIT ANY KEY FOR COMMAN
     D MODE":GOSUB1350:RETURN
1300 PRINT"HIT ANY KEY TO CONTINUE":GOSUB135
     0:GOTO1270
1310 GOTO60
1320 R$=INKEY$:IFR$=B$ THEN 1320
1330 RETURN
1340 FORQ=1 TO 1000:NEXTQ:RETURN
1350 B$="":R$=INKEY$:IFR$=B$ THEN 1350
1360 RETURN
1370 CLS:PRINT"ELECTRIC":PRINT:PRINT"COMMAND
      LIST # 4"
1380 PRINT"1-DISPLAY ELELCTRIC"
1390 PRINT"2-RETURN TO COMMAND LIST # 1"
1400 INPUT"ENTER COMMAND BY NUMBER";R:IFR<1 ~
     OR R>2 THEN 1370
1410 ON R GOSUB 1420,1510:GOTO1370
1420 IFN<1 THEN PRINT:PRINT"*** NOT ENOUGH D
     ATA ***":GOSUB1540:RETURN
1430 FORJ=1 TO N
1440 R=E(J):S^F(J):Y(J)=R:YY(J)=S
1450 Y=Y(J):YY=YY(J)
1460 NEXTJ:K=-3:L=0
1470 K=K+4:L=L+4:IFL>N THEN L=N
1480 CLS:PRINT"DATE";TAB(14);"COST";TAB(26);
     "UNITS"
1485 FORJ=K TO L:PRINTD$(J);TAB(14);E(J);TAB
     (25);F(J):NEXTJ:PRINT
1490 IFL=N THEN PRINT"HIT ANY KEY FOR COMMAN
     D MODE":GOSUB1550:RETURN
1500 PRINT"HIT ANY KEY TO CONTINUE":GOSUB155
```

```
    0:GOTO1470
1510 GOTO60
1520 R$=INKEY$:IFR$=B$ THEN1520
1530 RETURN
1540 FORQ=1 TO 1000:NEXTQ:RETURN
1550 B$="":R$=INKEY$:IFR$=B$ THEN 1550
1560 RETURN
1570 CLS:PRINT"TELEPHONE":PRINT:PRINT"COMMAN
    D LIST # 5"
1580 PRINT"1-DISPLAY TELEPHONE"
1590 PRINT"2-RETURN TO COMMAND LIST # 1"
1600 INPUT"ENTER COMMAND BY NUMBER";R:IF R<1
     OR R>2 THEN1570
1610 ON R GOSUB 1620,1710:GOTO1570
1620 IFN<1 THEN PRINT:PRINT"*** NOT ENOUGH D
    ATA ***":GOSUB1740:RETURN
1630 FORJ=1 TO N
1640 R=G(J):Z(J)=R
1650 Z=Z(J)
1660 NEXTJ:K=-7:L=0
1670 K=K+8:L=L+8:IFL>N THEN L=N
1680 CLS:PRINT"DATE","COST"
1685 FORJ=K TO L:PRINTD$(J),G(J);NEXTJ:PRINT
1690 IFL=N THEN PRINT"HIT ANY KEY FOR COMMAN
    D MODE":GOSUB1750:RETURN
1700 PRINT"HIT ANY KEY TO CONTINUE":GOSUB175
    0:GOTO1670
1710 GOTO60
1720 R$=INKEY$:IFR$*B$ THEN1720
1730 RETURN
1740 FORQ=1 TO 1000:NEXTQ:RETURN
1750 B$="":R$=INKEY$:IFR$=B$ THEN1750
1760 RETURN
```

©

Animate your Atari players – this set of programs creates the illusion of motion using only four drawings. And it's simple to add additional players which can each move independently.

Animation And P/M Graphics

Tom Sak and Sid Meier
Baltimore

You're already familiar with the Atari's ability to rapidly move a player from one location to another. But there are many times when you would like to do more than simply move a player; you'd like to give it lifelike motion, or animation. Spend a few minutes and learn how you can achieve these effects with far less effort than you might have imagined.

The art of bringing life to still pictures is much older than many of us realize. The production of books which contained moving pictures was well established before the invention of the motion picture camera and projector. The effect of moving pictures was typically accomplished by rapidly flipping the pages of a booklet containing simple character drawings, making them seem to spring to life.

Walt Disney and numerous other animators have produced this illusion of motion by drawing series of pictures in which each picture differs from the previous one only in a very small detail, a subtle displacement of each moving element. The pictures are then photographed for subsequent projection.

For example, an animator draws a man who appears to raise his arm away from his side, using a sequence of drawings. The first drawing would show the man facing you with both arms at his sides. The second picture differs only in that one arm is now slightly away from the man's side. The next picture shows the arm slightly further away, and so on through the sequence of drawings.

Animate With Only Four Drawings

As each picture in the series is viewed in rapid succession, by flipping through the stack of drawings, the figure appears to be raising his arm away from his side. A motion picture film consists of an analogous sequence of pictures which also provide the illusion of motion when they are projected and viewed in rapid succession.

As you can well imagine, a very large number of drawings is required to produce even a relatively short motion picture sequence. Since you're not about to adapt *Fantasia* for the small screen attached to your Atari, we will show you a way to use only four drawings, repeated in a cyclical pattern, to produce the illusion of motion. This is a very effective shortcut which makes it practical to adapt the animator's techniques to your BASIC program.

Now for some Atari animation. There is no question that our artistic creativity and graphic talents may never rival those of Walt Disney, but we will endeavor to adapt the basic animation technique which he popularized in order to move four "cowboys" from right to left across your television screen, totally out of step with each other.

For illustrative purposes we'll begin by moving only one cowboy. Program 1 accomplishes this objective by using the automatic player-missile graphic manipulation of the vertical blank interrupt routine which we discussed in **COMPUTE!**, February 1982, #21. Those of you who have entered the example program in that article will be pleased to know it already contains the animation features described here.

Program 2 adds complexity to the one cowboy program, illustrating the asynchronous movement of four players. Developing an understanding of the more complex program won't be too difficult once you've grasped the concepts in Program 1.

Reviewing Vertical Blank Interrupts

An elementary understanding of our vertical blank interrupt routine, VBLANK PM, is a prerequisite. Here we will review highlights of our previous article.

VBLANK PM is a machine language subroutine which occupies a portion of memory page six. It is initialized by a single BASIC USR function call which causes VBLANK PM to notify the operating system of both its presence and its desire to be automatically invoked during each vertical blank interrupt.

Prior to initialization, a 2K (2048) byte memory allocation must be made for the storage of players, and the players must be drawn. Following initialization, a POKE of the x-axis (horizontal) and y-axis (vertical) screen coordinates is all that is required to cause a player to be automatically moved during the next vertical blank period, or approximately every 1/60 of a second.

Not mentioned in the previous article is the fact that VBLANK PM has an animation feature just waiting to bring life to your players. All you need do is supply a few more drawings. The drawings and the current display image are contained in the 2K byte storage block.

Players Are Stored As Separate Images

Figure 1 depicts the memory allocated for the storage of players (see line 1030 in Program 1; memory allocation is explained in our earlier article). The current displayed image of player zero resides at locations PMBASE + 1024 through PMBASE + 1279; player one's homestead is PMBASE + 1280 through PMBASE + 1535, and so on for the other two players.

To achieve the animation, you need more than one image of each player, so the lower 1K (1024) locations (PMBASE through PMBASE + 1023) of the 2K byte storage block are used to hold the necessary set of drawings. Each player's drawings are stored in an area of memory beginning at a location which is 1K bytes below (lower memory address) the player's position in the upper 1K portion of the 2K byte storage block. A drawing is copied to the upper 1K portion by VBLANK PM when it is to be displayed. As a matter of fact, you won't draw anything at all in the upper 1K locations but will let VBLANK PM look after this chore for you.

For example, all of the player zero drawings reside at the 256 locations beginning at PMBASE. The currently displayed image of player zero resides at locations PMBASE + 1024 through PMBASE + 1279. The drawings for player zero are stored 1024 locations below this point, which is equal to PMBASE + 1024 minus 1024, or simply PMBASE. The player one drawings begin at PMBASE + 256, or (PMBASE + 1280)-1024, and so on for players two and three at locations PMBASE + 512 and PMBASE + 768, respectively.

A note of caution: we mentioned in the previous article that you could use the lower 1K bytes for your own purposes without disturbing anything. This is true only when the VBLANK PM animation feature is not going to be used. We hope that you've not been led too far astray!

At the risk of stating the obvious, we'd like to mention that as soon as you've decided to use more than one drawing per player – which you must do in order to achieve the animation – you can no longer have a player which is 255 lines tall. This is true because there are only 256 locations in which to store all of the drawings necessary to animate a single player. The first position, location zero, of each storage bin is reserved for a reason discussed later.

Initialize The Vertical Blank Routine

Now let's turn our attention to Program 1. Line numbers ending in zero are unchanged since the February article; and, for those who previously keyed the lengthy DATA statements containing VBLANK PM, we've made no changes to the machine language subroutine.

Lines 105 through 205 are the main program which causes our ragtag cowboy to meander across the screen. The BASIC code required to load and initialize VBLANK PM is found on lines 1000 through 1110. The VBLANK PM machine language subroutine is represented as DATA in lines 2000 through 2100. Finally, lines 3005 through 3045 contain the four drawings, used to describe a single player.

Before reviewing the main program, we'll go over the initialization subroutine which performs three functions: load VBLANK PM, load the player's drawings, and initialize VBLANK PM.

Lines 1010 and 1020 cause VBLANK PM to be read from DATA statements and POKEd into memory page six. A more memory-efficient method of representing VBLANK PM is the use of a string variable instead of DATA statements. Using this alternative, you continue to POKE the VBLANK PM code into page six, but from the string variable instead of from DATA statements.

You would save memory because only a single byte of memory is required in the string variable assignment statement to represent a byte of machine language code. In the DATA statement, as many as three bytes may be required for the same thing. For certain other machine language code applications, you can directly execute from the string, eliminating the need to POKE the code into another memory location. If you're interested in more on this topic, look for the article "Creating and Using Program Storage Strings" in this issue.

How The Animation Works

Line 1030 acquires the 2K byte memory storage block and line 1040 assures that the upper 1K byte display portion is cleared. Lines 1045 through 1065 are responsible for reading and storing the player's drawings in the lower 1K byte portion of the storage block. The four drawings of a cowboy are illustrated in Figure 2; you see now why Disney Studios can rest easy!

Notice that in line 1045 the first location in which the first drawing is stored is established as one byte above PMBASE; you will learn why this is necessary in a minute. The FOR statement on line 1055 assures that four drawings (zero through three) are read and stored. Each drawing is 24 lines tall, so we begin the FOR loop on line 1065 with the base of the first drawing offset by 24 bytes for each previous drawing stored. Since each drawing consists of 24 bytes, the loop is completed by adding 23 to the starting point.

Line 1075 designates the player's color. Line 1080 establishes the locations to be POKEd to change the player's x-axis and y-axis screen coordinates (PLX and PLY) and to set the length (height)

The Hard & Soft of It: ATARI from ASAP

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

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

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

OPTIONAL ACCESSORIES	PRICE
ATARI® 410™ Program Recorder	80.00
ATARI® 810™ Disk Drive	470.00
ATARI® 822™ Thermal Printer	299.00
ATARI® 820™ 40-column Dot Matrix Impact Printer	279.00
ATARI® 825™ 80-column Dot Matrix Impact Printer	645.00
ATARI® 830™ Acoustic Modem	159.00
ATARI® 850™ Interface Module	175.00
ATARI® Paddle (CX30-04) and Joystick (CX40-04) (Pair)	17.95
ASAP 16K RAM Module	45.00

COMPLETE SOFTWARE LIBRARY INCLUDES THESE POPULAR UNITS:

Atari®

Video Easel ROM	26.00
Music Composer ROM	44.00
Assembler/Editor ROM	45.00
Mortgage Loan Analysis Cassette	15.95
Stock Analysis Disk	19.95
Stock Charting	22.95
Bond Analysis Disk	22.95
Mailing List Cassette	19.95
Touch Typing (2 Cass)	19.95
Graph It (2 Cass)	17.95
Word Processor	119.00
Personal Finance	64.95
Microsoft Basic	75.00
Basketball ROM	27.00
Super Breakout ROM	33.00
Computer Chess ROM	32.00
3D Tic Tac Toe ROM	26.00
Star Raiders ROM	37.00
Kingdom Cassette	12.95
Blackjack Cassette	12.95
Biorhythm Cassette	12.95
Energy Czar Cassette	12.95
Telelink ROM	24.00
Space Invaders ROM	33.00
Scram	18.95
Asteroids	32.00
Missile Command	32.00
Caverns of Mars Disk	32.00
Centipede Disk	35.00
Pac-Man	35.00

Arcade Plus

Ghost Hunter Cassette	23.50
Ghost Hunter Disk	27.50

Datasoft

Atari Mailing List Disk	19.95
Atari Character Generator Disk	16.95
Text Wizard Disk	89.95
Micropainter Album 1 Disk	16.95
Micropainter Album 2 Disk	16.95
Le Stick Accessory	30.00

On-Line Systems

HI-RES Adv #0 — Mission: Asteroid Disk	19.95
HI-RES Adv #2 — Wiz & Princess Disk	25.00
Roadwork Disk	29.95
Jawbreaker Disk	25.00
Softporn Adventure (X-rated) Disk	25.00
The Next Step Disk	29.95

Personal Software

Visicalc Disk	169.00

FOR HOME ENTERTAINMENT SYSTEMS:

Activision

Dragster Cartridge	17.95
Boxing Cartridge	17.95
Checkers Cartridge	17.95
Fishing Derby Cartridge	17.95
Skiing Cartridge	17.95
Bridge Cartridge	22.00
Tennis Cartridge	17.95
Laser Blast Cartridge	17.95
Freeway Cartridge	17.95
Kaboom! Cartridge	17.95
Stampede Cartridge	17.95
Ice Hockey Cartridge	22.00

asap computer products, inc.
1198 E. Willow St., Signal Hill, CA 90806

ALL PRICES SUBJECT TO CHANGE WITHOUT NOTICE. CALL FOR BEST PRICE
ASAP offers a 15-day buyer protection policy: full money-back guarantee if not totally satisfied.
Ordering Information: name, address, phone; ship by: UPS or Mail. Shipping charge: add $2.90 up to 1 lb. (UPS blue), U.S. Mail add $1.50 (U.S. only) ($25.00 minimum order).
Terms: We accept cash, check, money orders, Visa and Master Charge (U.S. funds only). Tax: 6% Calif. res., COD's and terms available on approval (School PO's Accepted).

Toll free outside California: **(800) 421-7701.** Inside California: **(213) 595-6431 (714) 891-2663**
Inside Canada call **ASAP Computer Products, LTD.**, 116 Viceroy Rd., D-12 Concord, Toronto, Ontario, Canada L4K 1A9 **(416) 738-0500 (800) 268-1996**

of the player (PLL).

The x-axis screen display position for players zero, one, two, and three are indicated by POKEs to PLX, PLX+1, PLX+2 and PLX+3, respectively. The analogous situation is true for setting the player's y-axis coordinate (PLY, PLY+1, ...) and the player's height (PLL, PLL+1, ...), and for selecting the next drawing to be displayed (PDR, PDR+1, ...).

PDR is defined on line 1085 and is used to select the next drawing to be used as the player's current display image. VBLANK PM is responsible for copying the drawing to the appropriate location in the upper 1K byte portion of the 2K byte storage block. A value in the range of one to 255 is POKEd into PDR to indicate the bottom-most line of the selected drawing. The most recent value POKEd into PLL indicates the number of bytes (the height of the player) to be copied.

VBLANK PM Must Announce Itself

A value of zero POKEd into PDR signals VBLANK PM to continue to display the current image. This is why we were careful to avoid location zero when loading the first drawing. VBLANK PM sets PDR to zero automatically after it copies a drawing to the upper 1K byte display area.

Location 1771, POKEd in line 1085, is a location in VBLANK PM which must contain the memory page number of the first page in which drawings are stored. Location 1788, referenced on line 1090, is also in VBLANK PM, and must contain the page number of the beginning of the upper 1K byte current display portion. (These parameters afford even greater flexibility to VBLANK PM, features which are beyond the scope of this discussion.)

The other POKEs on line 1090 are associated with the Atari's player-missile graphics mechanism which is described in numerous other articles including our February article.

VBLANK PM is initialized on line 1100. This is the only explicit BASIC function call to VBLANK PM which is required. As a result of this call, VBLANK PM will register its intention to become a part of the vertical blank interrupt process with the operating system.

Inside The Main Routine

Turning our attention to the main program, we start with line 105, which establishes the television screen background, or playfield. It is important that you always define a graphics mode (execute a graphics statement) before you initialize VBLANK PM; if you fail to follow this sage advice, you are likely to be plagued by a strange flashing vertical bar on your screen.

It doesn't matter which graphics mode is specified since Atari players are independent of the mode. Graphics mode one is chosen to provide a text window to serve as a walkway for our strolling cowboy. Line 125 sets the y-axis position of the cowboy so he appears to walk on top of the text window. The player's height is also established on line 125.

The animation is performed by lines 135 through 205. These lines should be relatively easy to comprehend once you have a mental picture of the way in which the drawings were stored during the initialization procedure. The variable DRAW, initialized as one on line 135, selects the next drawing to be used as the current display image.

Lines 145 and 165 control the right to left motion of the cowboy by using the index variable I as the x-axis coordinate of the player. The POKE to PDR on line 185 selects the next drawing to be displayed, and the calculation on line 195 results in the selection of the drawing to be used in the next cycle when the cowboy takes his next step.

The IF statement on line 195 assures that after the fourth drawing is used, the program will cycle and begin anew with the first drawing. The FOR loop on line 205 controls the speed with which the cowboy strolls across the screen. A maximum value of 30 results in a movement which you might describe as a brisk walk. The larger the maximum value of this delay loop, the slower the pace of the player.

The cowboy will continue to walk across the screen until you stop the program. Incidentally, the program does not gracefully turn off the Atari's player-missile graphics mechanism, so you are well advised to press SYSTEM RESET to remove the undesirable residue from the screen. (POKE 53277,0 turns off the player-missile gracefully.) Be patient when the program is started, since it takes more than ten seconds for the initialization procedure.

Four Heads Are Better Than One

And that's almost all there is to animation! Are you ready to tackle a little bit more challenging project? Program 2 represents enhancements to the program we've been reviewing. It uses all four players and, while it causes them to walk out of step with each other, it employs only the same four drawings.

Program 2 modifies seven lines and adds two more. The changed lines are: 125, 165, 185, 205, 1045, 1055 and 1075; lines 155 and 175 are new.

Line 1045 now includes a FOR statement to cause the drawings to be READ and POKEd in the storage area associated with the additional three players. Note also that the calculation of DRWBAS is revised to reflect the additional players. DRWBAS contains the address of the first byte of the drawing storage area containing the first drawing for the current player. As the value of the variable, I, in the FOR loop is indexed from 0 to 3, DRWBAS will take the values 1, 257, 513 and 769. The first byte, location 0, of each storage area is skipped for the reason mentioned earlier.

A RESTORE statement is added to line 1055 which resets the DATA pointer to reread the same drawings for each player. The modification to line 1075 is simply the addition of player colors for the new players.

Looking at the main program, line 125 now establishes the y-axis and height for four players rather than one. Line 155 is added to cycle through the x-axis movement and picture selection for all players.

In line 165 we've added a calculation to the x-axis positioning POKE to maintain a separation between the cowboys which is equal to slightly more than the width of a single player as measured from the leftmost edge of one player to the leftmost edge of the following player.

Still Only Four Drawings

Line 175 is added to assure that a different drawing is used as the current display image for each player. The variable DRAW continues to determine the

Figure 1.

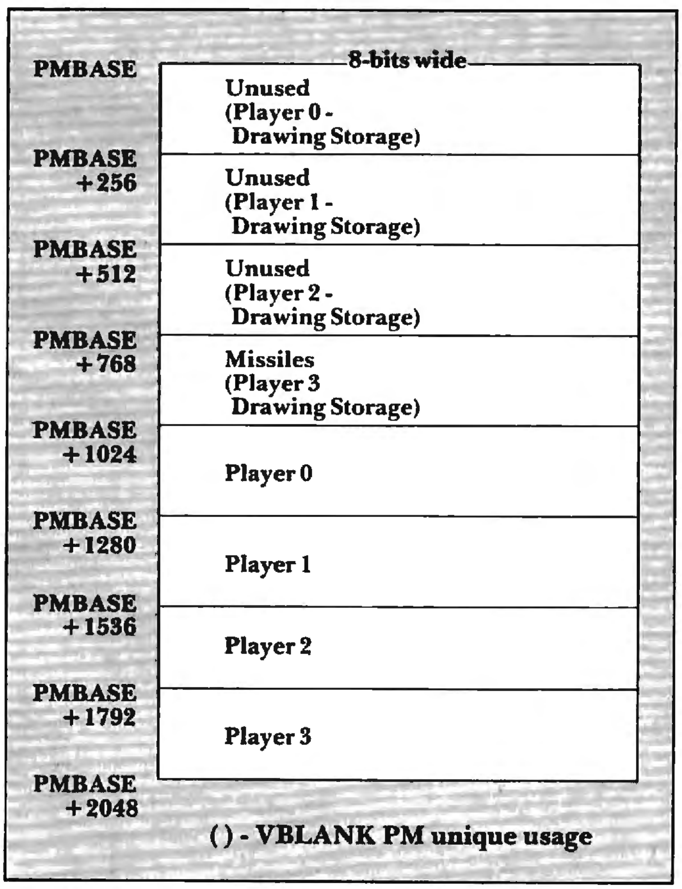
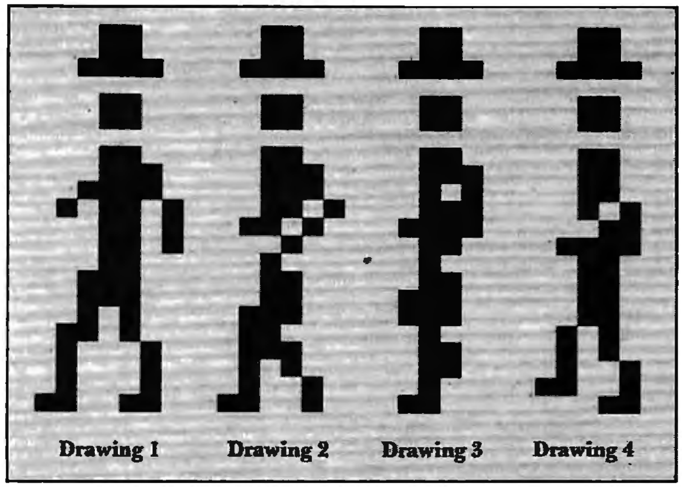


Figure 2.



drawing to be selected for player zero. Study the statement, and you will discover that each player will be depicted by the drawing following that used for the previous player. That is, if player zero is pictured by the first drawing, then player one is illustrated by the second, player two by the third, and, finally, player three is displayed as the fourth drawing. A circular assignment is used so that the fourth drawing is followed by the first.

The delay loop is omitted from line 205 because the additional calculations needed for the added players consume sufficient time to maintain a reasonable pace for all four cowboys. You might want to experiment with a delay loop to further slow the action; better yet, consider using GET to accept a keystroke instead of employing a delaying FOR loop. The GET will allow you to step the players across the screen in order to study the animation technique.

Don't you agree that animation makes a world of difference in the use of player-missile graphics? I was fascinated when my more talented partner, Sid, gave me a half dozen lines of cryptic BASIC statements to turn into an animation tutorial. The first time I saw them execute I was mesmerized. Go ahead, type either program into your Atari; you'll be addicted too.

Program 1.

```
5 REM .... P R O G R A M{4 SPACES}O N E ....
105 GRAPHICS 1:SETCOLOR 2,1,8:SETCOLOR 4,8,4
    :POSITION 5,3:? #6;"animation":POSITION
    3,5:? #6;"demonstration"
120 GOSUB 1000:REM initialize vb routine
125 POKE PLY,169:POKE PLL,24
135 DRAW=1
145 FOR I=212 TO 10 STEP -1:REM move rt to l
    ft horiz
165 POKE PLX,I:REM new position
185 POKE PDR,DRAW:REM new drawing
195 DRAW=DRAW+24:IF DRAW>73 THEN DRAW=1:REM
    select next drawing
```

```
205 FOR DELAY=1 TO 30:NEXT DELAY:NEXT I:GOTO
    145
1000 REM INITIALIZE VBLANK PM SUBR
1010 FOR I=1536 TO 1706:READ A:POKE I,A:NEXT
     I
1020 FOR I=1774 TO 1787:POKE I,0:NEXT I
1030 PM=PEEK(106)-16:PMBASE=256*PM
1040 FOR I=PMBASE+1023 TO PMBASE+2047:POKE I
     ,0:NEXT I
1045 DRWBAS=PMBASE+1
1055 FOR J=0 TO 3:REM four drawings
1065 FOR K=DRWBAS+J*24 TO DRWBAS+J*24+23:REA
     D X:POKE K,X:NEXT K:NEXT J
1075 POKE 704,12
1080 PLX=53248:PLY=1780:PLL=1784
1090 POKE 559,62:POKE 623,1:POKE 1788,PM+4:P
     OKE 53277,3:POKE 54279,PM
1095 PDR=1772:POKE 1771,PM
1100 X=USR(1696)
1110 RETURN
2000 REM vblank interupt routine
2010 DATA 162,3,189,244,6,240,89,56,221,240,
     6,240,83,141,254,6,106,141
2020 DATA 255,6,142,253,6,24,169,0,109,253,6
     ,24,109,252,6,133,204,133
2030 DATA 206,189,240,6,133,203,173,254,6,13
     3,205,189,248,6,170,232,46,255
2040 DATA 6,144,16,168,177,203,145,205,169,0
     ,145,203,136,202,208,244,76,87
2050 DATA 6,160,0,177,203,145,205,169,0,145,
     203,200,202,208,244,174,253,6
2060 DATA 173,254,6,157,240,6,189,236,6,240,
     48,133,203,24,138,141,253,6
2070 DATA 109,235,6,133,204,24,173,253,6,109
     ,252,6,133,206,189,240,6,133
2080 DATA 205,189,248,6,170,160,0,177,203,14
     5,205,200,202,208,248,174,253,6
2090 DATA 169,0,157,236,6,202,48,3,76,2,6,76
     ,98,228,0,0,104,169
2100 DATA 7,162,6,160,0,32,92,228,96
3005 REM drawings 0, 1, 2 and 3
3015 DATA 0,12,12,30,0,12,12,0,12,14,30,45,1
     3,13,12,28,28,20,52,34,34,34,102,0
3025 DATA 0,12,12,30,0,12,12,0,12,14,14,13,2
     6,4,8,12,12,28,24,28,20,18,50,0
3035 DATA 0,12,12,30,0,12,12,0,12,14,10,14,3
     0,12,8,12,28,28,8,12,12,8,24,0
3045 DATA 0,12,12,30,0,12,12,0,12,12,12,10,6
     ,30,12,12,12,12,20,20,18,50,6,0
```

Program 2.

This program uses the Vertical Blank Player/Missile routine, so add lines 2000-3045 of Program 1 when you type it in.

```
5 REM .... P R O G R A M{4 SPACES}T W O ....
105 GRAPHICS 1:SETCOLOR 2,1,8:SETCOLOR 4,8,4
    :POSITION 5,3:? #6;"animation":POSITION
    3,5:? #6;"demonstration"
120 GOSUB 1000:REM initialize vb routine
125 FOR J=0 TO 3:POKE PLY+J,169:POKE PLL+J,2
    4:NEXT J
135 DRAW=1
145 FOR I=212 TO 10 STEP -1:REM move rt to l
    ft horiz
155 FOR J=0 TO 3:REM four players
165 POKE PLX+J,I+J*10:REM new positioN, main
    tain separation
175 NXTDRW=DRAW+J*24:IF NXTDRW>73 THEN NXTDR
    W=NXTDRW-96:REM select different drawing
    for each player
185 POKE PDR+J,NXTDRW:NEXT J
195 DRAW=DRAW+24:IF DRAW>73 THEN DRAW=1:REM
    select next drawing
205 NEXT I:GOTO 145
1000 REM INITIALIZE VBLANK PM SUBR
1010 FOR I=1536 TO 1706:READ A:POKE I,A:NEXT
     I
1020 FOR I=1774 TO 1787:POKE I,0:NEXT I
1030 PM=PEEK(106)-16:PMBASE=256*PM
1040 FOR I=PMBASE+1023 TO PMBASE+2047:POKE I
     ,0:NEXT I
1045 FOR I=0 TO 3:DRWBAS=PMBASE+I*256+1:REM
     four players
1055 RESTORE 3015:FOR J=0 TO 3:REM four draw
     ings
1065 FOR K=DRWBAS+J*24 TO DRWBAS+J*24+23:REA
     D X:POKE K,X:NEXT K:NEXT J:NEXT I
1075 POKE 704,12:POKE 705,128:POKE 706,48:PO
     KE 707,192
1080 PLX=53248:PLY=1780:PLL=1784
1090 POKE 559,62:POKE 623,1:POKE 1788,PM+4:P
     OKE 53277,3:POKE 54279,PM
1095 PDR=1772:POKE 1771,PM
1100 X=USR(1696)
1110 RETURN
```

©

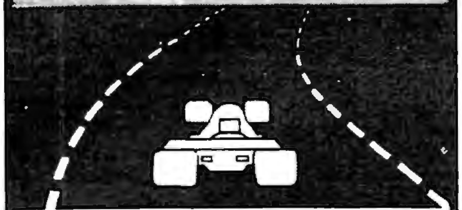
Royal Software

YOUR MARKETPLACE FOR:

*ATARI

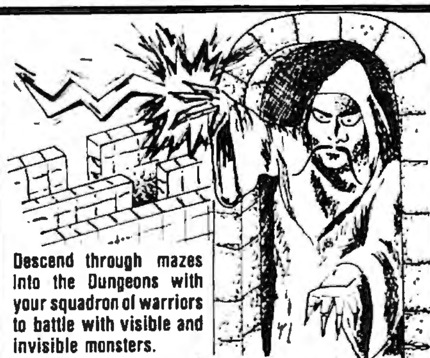
WE DEAL EXCLUSIVELY IN PRODUCTS FOR THE ATARI (THE BEST) COMPUTER

NIGHT RALLY



Test your driving skills to their limits. Climb into the driver's seat as you race against some of the best computer drivers in the world.

16K DISK $31.40 TAPE $26.90



Descend through mazes into the Dungeons with your squadron of warriors to battle with visible and invisible monsters.

Wizard of Wor

16K DISK ONLY $35.10

PRO-FOOTBALL

Play against the computer or another player. Over 25 offense and defense plays. The BEST football game yet.

16K
$26.90 TAPE
$31.40 DISK

HARDWARE

800 Computer 16K	$648
800 Computer 48K	$757
400 Computer 16K	$338
810 Disk Drive	$448
850 Interface	$178
410 Recorder	$ 78
MX80FT+ Printer	$636
16K Ram module	$ 69
32K Ram module	$109
Graphics Tablet	$278

Trade your 400 or older 800 for the new model 800 — Call for prices

SOFTWARE

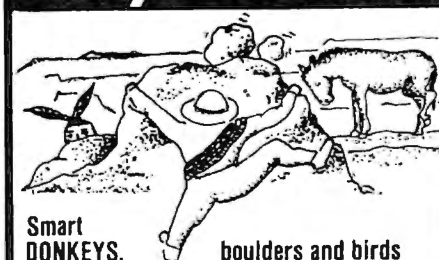
PAC MAN (cart.)	$39.50
Centipede (cart.)	$39.50
Microsoft Basic (D) 32K	$79.10
Ghost Hunter (T) 16K	$25.50
Ali Baba (D) 32K	$28.90
Protector (T) 32K	$22.45
Apple Panic (D) 48K	$26.90
Threshold (D) 40K	$33.90
GORF (D) 16K	$35.10
Crypts of Terror (D,T) 16K	$26.30
Text Wizard (D) 32K	$87.90
Dodge Racer (T) 16K	$20.20
Chicken (T) 16K	$20.20
De Re Atari Book	$17.90
Atari Games Book	$13.90
Personal Finance (D) 40K	$67.50
Zork I or II (D) 32K	$35.90
Deluxe Invaders (D) 16K	$31.40
Eastern Front (T,D)	$25.40
Mousekattack (D) 32K	$31.40
Galactic Chase (T) 16K	$22.40
Bug Attack (T,D) 32K	$26.90
Pacific Coast Hwy (T,D) 16K	$26.90
Alien Swarm (T) 16K	$26.90
Midnight Magic (D) 48K	$31.45
Intruder (T) 16K	$26.90
Nautilus (T,D)	$20.20

Call or Write for

FREE

ATARI PRODUCTS CATALOG
with hundreds of items

Canyon Climber



Smart DONKEYS, boulders and birds dropping rocks try to stop you from reaching the top. A real challenge. Arcade quality.

16K
DISK $26.90 TAPE $22.40

CROSSFIRE

The aliens have landed and it's your job to save the city. (If you can!) TOP RATE GAME, ARCADE QUALITY, HI-RES Graphics & Sound.

16K TAPE
32K DISK
$26.90

FROGGER

This is the genuine FROGGER game that you see in the arcades. Made by the same people that made Jawbreaker (One of the top ten sellers.)

16K DISK OR TAPE
$31.40

VISA | master charge

TO ORDER CALL TOLL FREE 800-452-8013

FOR INFORMATION CALL 503-683-5361

HOW TO ORDER: Send check or money order or call our toll free number and use your Visa Card. Shipping on software is $2.00 per order anywhere in USA. Hardware shipping call for cost. Add 3% for VISA or MC. Equipment subject to price change and availability without notice.

Store Hours
8 am — 6 pm
Mon. — Sat.

Royal Software, P.O. Box 10303, Eugene, Oregon 97440

A Monthly Column

Machine Language: Shreds And Patches

Jim Butterfield
Toronto

When you write a program, you expect it to be perfect. Sometimes it misses the perfection you expected, and you have to fix it.

After serious debugging you isolate the fault – or one of them, at least. Occasionally, it's a single instruction that's wrong – that LDA (Load A) should have been LDY (Load Y). You can fix it by correcting the hexadecimal Op Code and immediately go for another run. Rarely, it's code that you don't need: instead of storing zero into location 96, do nothing. Again, you fix it by overwriting. Change the unwanted Store instruction into a do-nothing NOP (No Operation) – or, more accurately, a series of NOP's – and the program is ready to go.

The annoying problem is the most common – you've left something out of the code and need to shoehorn it in somehow. You need to find empty space within your program – and there's no space there.

Classic Correction

The classic answer to program repairs is to redo the program. If you have a symbolic assembler, this isn't hard. You add the missing instructions, call in the assembler, and a new program is generated. Unused space is closed up, new space is created as necessary, and all the branches, jumps, and calls are recalculated. It's ideal, but we don't always go that way.

Why not always reassemble? There are a number of valid reasons.

Sometimes the owner of a small computer doesn't have an assembler; perhaps his system isn't big enough to support one. He'll be assembling by hand, or by using tiny assemblers like the one in Supermon. A new assembly means a lot of work.

Even when an assembler is available, the user can perceive it as a lot of trouble to use during a debugging session. The test program must be thrown out and the assembler loaded; a new "object" program – that's a machine language program – must be created. The clincher is a paperwork problem: to reassemble and do the job right, the new program should be dated and version-numbered; and then a program listing should be generated. That's potentially a lot of paper and a lot of printing time. Yet it's needed, since the programmer will need to know where the code is located during testing.

What's the alternative? A simple procedure known as "patching" can add corrections to a program without the work of a full assembly.

Patches

The principle of a patch is this: to add new code, you must destroy some of the old code by overwriting it with a Jump instruction. The Jump will take you to a fresh part of memory (the patch area) in which the old code will be reconstructed and the new code added. Finally, the patch program will Jump back and allow normal program execution to resume.

Let's do this with a simple example. At address hex 027A, we have the following two instructions: LDY #0:LDA #$20 .. followed by more program. During testing we discover that we have forgotten an important step – say, printing a carriage return character (hex 0D). There obviously isn't room to insert the missing code into the program; how do we handle it?

First, we look around for a patch area. In this case, we might find that there is free memory starting at address $0300. We know that we will want to insert a JMP $0300 instruction – three bytes long – into our code. Since our LDY and LDA instructions are two bytes each in length, we're going to clobber both of them (which means that we'll need to rebuild them both).

OK, at location $0300 we can code: LDA #$0D:JSR $FFD2 to print the carriage return. Now we must rebuild the butchered code with LDY #0:LDA #$20 and finally return to the continuation point with JMP $027E. The last step is to place the JMP $0300 instruction at $027A and activate the patch.

All this must be done in machine language, so hand assembly is necessary. It's not really hard.

Our coding at $0300 works out to A9 0D 20 D2 FF A0 00 A9 20 4C 7E 02; and at 027A we place the code 20 00 03. Note that there's a "left-over" byte at address 027D, but that doesn't matter. Now we can take another shot at our program and see if all the problems are corrected.

One more thing: you need to make careful records of your patches. As patches are written, the memory they occupy must be marked off so that you won't try to use those locations again. The patch itself must be written out carefully – you may need it during debugging. If you find a bug in a patch, it's better not to try to "patch the patch." Just write a brand new corrected patch program somewhere else.

Wrapping It Up

Patches are usually temporary activities during a debugging session. Testing takes place; a bug is found; a patch is written; testing resumes; more bugs, more patches, etc. Eventually, when the program behaves satisfactorily, you'll want to clean up and reassemble. The patches have done their job; they've allowed you to whip the program into running shape. Now you'll want to clean up, document, and so forth. The patches look ungainly; you'll want them out of there.

Occasionally, however, patches are left in place permanently. If a program has been released and users have come to depend on certain "entry points," it would be unwise to reassemble, which would move things around and cause problems.

Patch Points

Certain places are easy to patch. If there's a JMP or JSR (or, for that matter, almost any three-byte instruction) at a convenient place in your code, you can quite easily slip in extra code with little dislocation.

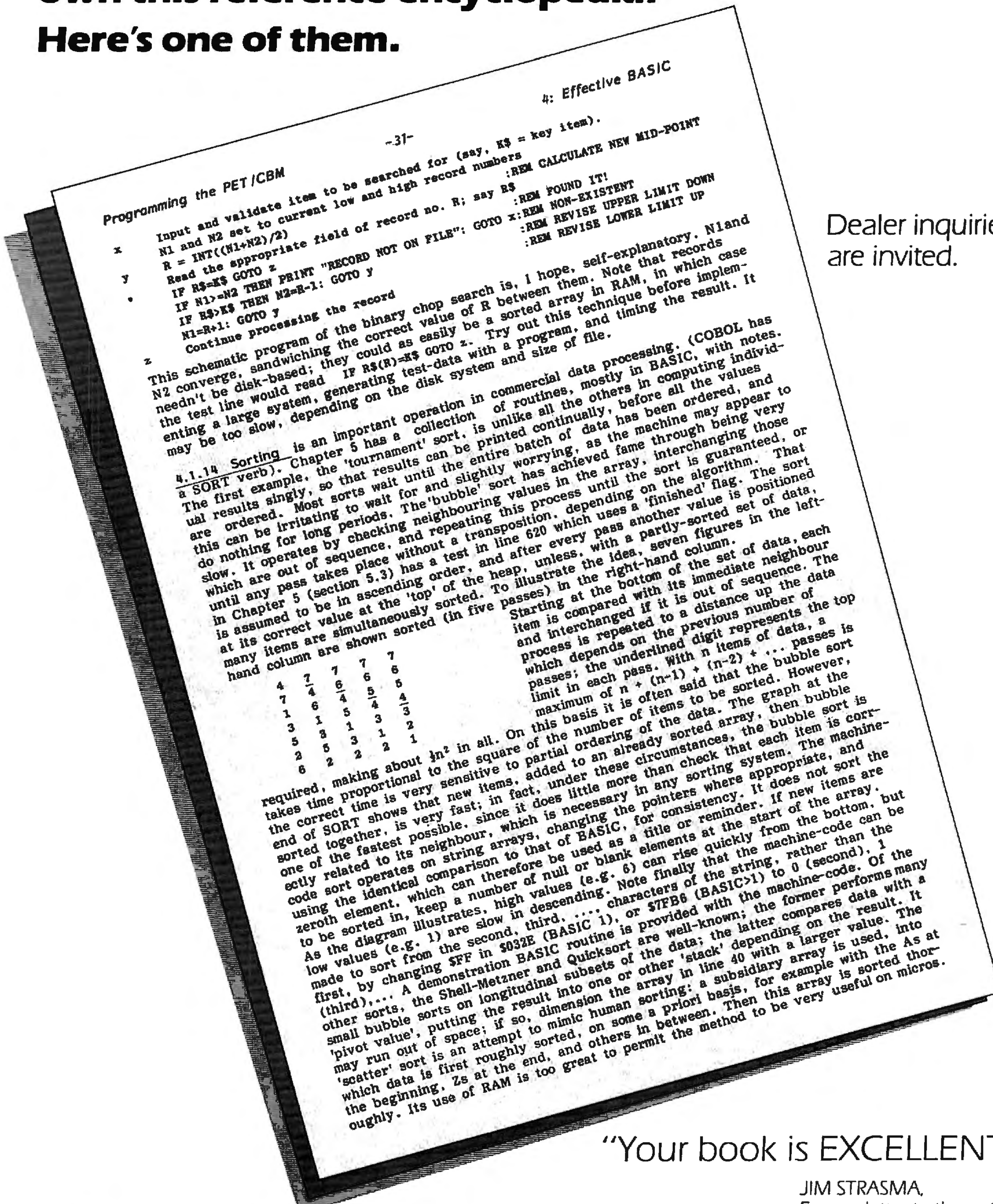
Other code is quite difficult. The 6502 Branch instructions are relative, and will only reach 120-odd locations either way. If you tried to overwrite a Branch instruction, you might have troubles rebuilding it in your patch – it probably wouldn't reach.

Some programmers make provision for patches as they write code. Every once in a while, they throw in a group of three NOP instructions, which do nothing but provide space for hooking in a patch for correction or testing.

Most of us, however, forge ahead in the expectation that our coding will be perfect the first time around. Occasionally we're right. When we're wrong, we reach for a patch. ©

There are over 500 reasons to own this reference encyclopedia.

Here's one of them.



Programming the PET/CBM -31- *4: Effective BASIC*

```
x     Input and validate item to be searched for (say, K$ = key item).
      N1 and N2 set to current low and high record numbers
      R = INT((N1+N2)/2)                            :REM CALCULATE NEW MID-POINT
      Read the appropriate field of record no. R; say R$
y     IF R$=K$ GOTO z                               :REM FOUND IT!
      IF N1>=N2 THEN PRINT "RECORD NOT ON FILE": GOTO x:REM NON-EXISTENT
•     IF R$>K$ THEN N2=R-1: GOTO y                  :REM REVISE UPPER LIMIT DOWN
      N1=R+1: GOTO y                                :REM REVISE LOWER LIMIT UP
z     Continue processing the record
```

This schematic program of the binary chop search is, I hope, self-explanatory. N1and N2 converge, sandwiching the correct value of R between them. Note that records needn't be disk-based; they could as easily be a sorted array in RAM, in which case the test line would read IF R$(R)=K$ GOTO z. Try out this technique before implementing a large system, generating test-data with a program, and timing the result. It may be too slow, depending on the disk system and size of file.

4.1.14 Sorting is an important operation in commercial data processing. (COBOL has a SORT verb). Chapter 5 has a collection of routines, mostly in BASIC, with notes. The first example, the 'tournament' sort, is unlike all the others in computing individual results singly, so that results can be printed continually, before all the values are ordered. Most sorts wait until the entire batch of data has been ordered, and this can be irritating to wait for and slightly worrying, as the machine may appear to do nothing for long periods. The 'bubble' sort has achieved fame through being very slow. It operates by checking neighbouring values in the array, interchanging those which are out of sequence, and repeating this process until the sort is guaranteed, or until any pass takes place without a transposition, depending on the algorithm. That in Chapter 5 (section 5.3) has a test in line 620 which uses a 'finished' flag. The sort is assumed to be in ascending order, and after every pass another value is positioned at its correct value at the 'top' of the heap, unless, with a partly-sorted set of data, many items are simultaneously sorted. To illustrate the idea, seven figures in the left-hand column are shown sorted (in five passes) in the right-hand column.

				7
			7	6
		7	6	5
4	7	6	5	4
7	4	4	4	3
1	6	5	3	2
3	1	1	1	1
5	3	3	2	
2	5	2		
6	2			

Starting at the bottom of the set of data, each item is compared with its immediate neighbour and interchanged if it is out of sequence. The process is repeated to a distance up the data which depends on the previous number of passes; the underlined digit represents the top limit in each pass. With n items of data, a maximum of n + (n-1) + (n-2) + ... passes is required, making about $\frac{1}{2}n^2$ in all. On this basis it is often said that the bubble sort takes time proportional to the square of the number of items to be sorted. However, the correct time is very sensitive to partial ordering of the data. The graph at the end of SORT shows that new items, added to an already sorted array, then bubble sorted together, is very fast; in fact, under these circumstances, the bubble sort is one of the fastest possible, since it does little more than check that each item is correctly related to its neighbour, which is necessary in any sorting system. The machine-code sort operates on string arrays, changing the pointers where appropriate, and using the identical comparison to that of BASIC, for consistency. It does not sort the zeroth element, which can therefore be used as a title or reminder. If new items are to be sorted in, keep a number of null or blank elements at the start of the array. As the diagram illustrates, high values (e.g. 6) can rise quickly from the bottom, but low values (e.g. 1) are slow in descending. Note finally that the machine-code can be made to sort from the second, third, ... characters of the string, rather than the first, by changing $FF in $032E (BASIC 1), or $7FB6 (BASIC>1) to 0 (second), 1 (third),... A demonstration BASIC routine is provided with the machine-code. Of the other sorts, the Shell-Metzner and Quicksort are well-known; the former performs many small bubble sorts on longitudinal subsets of the data; the latter compares data with a 'pivot value', putting the result into one or other 'stack' depending on the result. It may run out of space; if so, dimension the array in line 40 with a larger value. The 'scatter' sort is an attempt to mimic human sorting: a subsidiary array is used, into which data is first roughly sorted, on some a priori basis, for example with the As at the beginning, Zs at the end, and others in between. Then this array is sorted thoroughly. Its use of RAM is too great to permit the method to be very useful on micros.

Dealer inquiries are invited.

"Your book is EXCELLENT!"

JIM STRASMA,
From a letter to the author.

A **COMPUTE! Books** Publication
In conjunction with Level Limited.

Apple Manager:

An Alphanumeric Data Manager

Robert Jacques Beck
Minneapolis, MN

I began writing a data management program as part of a classroom assignment, but I finished it only because I had become obsessed with fitting the pieces of the puzzle together. I learned that the ideal data management system does everything under the sun and will never be invented. The data manager described in this article is designed primarily for string data, although numeric applications are possible.

It has two advantages. First, you get a listing. Second, it's written in BASIC and you get some explanation of how it works. If you find that it doesn't meet all your needs, and you don't want to modify it, you'll have some valuable knowledge if you go looking for another data manager. The program is written for the Apple – hence the name Apple Manager – but many ideas collected in it can be used elsewhere. The rest of this section is a somewhat theoretical discussion, so if you want to get into the particulars of the program just skip ahead.

Computers are great at keeping track of large masses of information. That's what data management is all about, so asking "Why do we need data managers?" is like asking "Why do we need computers?" But that is a kind of circular definition. Maybe we should ask, "What should a data manager program do?"

I like to think of data managers as two-way transportation systems between my diskettes and me or, more technically, between the storage device and the information source. To store data we must input it, but that's not enough to give us control over the contents of our data files: we might want to come back later and modify or delete something. Similarly, information retrieval is not just a matter of pulling the stuff out as fast as we can; we may be interested in one kind of information on Monday and another on Tuesday. You'll see flexibility come alive when you try the program.

From Aardvarks to Ziggurats

Since data is stored in files, a data manager is first and foremost a file manager. You can use the same data manager program to deal with files from Aardvarks to Ziggurats because each file will have the same general structure, even though individual components may vary. You may not think about it, but you will definitely take advantage of similarities in file structure when you write additions to a data manager or interface your files with other programs.

Just as books are made up of pages, files are made up of records. Records can be subdivided into fields, much like the sentences on a page. So we can define a record as a logical grouping of several individual data items. If each record in a file is identical (that is, if it has the same size and makeup), several records hypothetically placed adjacent to each other will look like a rectangle. A rectangular file structure is easy to program.

Another possibility is a hierarchical structure. Hierarchical files have records that are built from the same group – but not necessarily the same number – of components. Hierarchies occur naturally in many applications. Suppose you want to keep tabs on the books in your library and you want to cross-index them by one or more topics. One approach is a rectangular file with each record storing information about one book.

But there's a slight problem. You will need a field for author, another for title, and one additional field for each topic. Because you need to know how many fields to allot, you'll have to decide in advance on a reasonable maximum number of topics. On the other hand, a hierarchical file doesn't lock you into a fixed design. Imagine a hierarchy with author at the top. Titles are second in status, with each title being linked to its author's name. Topics are linked, in turn, to titles. In both cases, there is no set number of linkages.

Though this program is based on a rectangular organization, I just wanted to point out that there are alternative ways to set up data bases.

Module Structure

The more a program does, the more likely it is to grow to an unmanageable size. One way to cope with this is to break your program into chunks called modules. The main driver (lines 91-92) prints a menu and lets your choose one of the five modules: Files, Records, Reports, Select File, or Utilities. With the exception of Select File, all of the modules are multi-functional so the first thing you see when you enter them is another menu.

Menus allow you to move away from the main drive and into the tangled depths of the program, but how do you return? Whenever Apple manager requests input, if you type CONTROL O (for Out), plus a RETURN if necessary, you'll back up one level in the program hierarchy. This is a handy escape from any operation. It's the only way you exit from functions that don't automatically return to menu. To switch from one module to another, type CONTROL O to back up to the main menu, then select the new module.

To conserve memory, I used a lot of subroutines and multiple statement lines. Program lines are numbered consecutively, instead of adding 10 to each line number as is usually done. There are so many GOSUBs and GOTOs that I saved about 300 bytes this way. Variable names are short and variables are used and reused whenever possible. Look at Table 1 to sort out some of the confusion.

Diskette Data Bases

Apple Manager assumes every diskette is an independent filing system. Each diskette has one title file, which is a sequential file containing the number of data files on the diskette and their names. When the program starts up it reads the names into an array (T$, line 89). Whenever a file name is input later on, it can be checked against this array to see if it is a valid file name for the diskette (lines 57-60). Data is kept in random access files. The data manager has to know how to relate to these files. Somebody has to tell it things like what size record to use, how many fields, what their names are, and so on. The easiest way to do it is to put the information into a file.

Rather than put it all in one file, I set up a separate file for the description of each data file. The description file is read into two arrays, one containing the names of the fields and the other containing the field lengths in characters (or bytes, since one character is stored per byte). You might want to look at the Atari Data Manager in **COMPUTE!** (November, 1981, #18) where the same concept is implemented somewhat differently. Fields are referred to as "items" by Apple Manager, so I'll use the two terms interchangeably. To summarize, each diskette has one title file, and for each data file there will be a description file.

Files

Most of this module is pretty easy to use once you get it running. The Catalog function is simple: it lists the title file array (lines 99-100). Describe File is similar in that it prints the description file arrays (lines 101-103). Create File is a bit more complicated – here's where new file structures are born. First type in the file name, then the number of items per record. All items are alphanumeric, in other words: strings. (Numbers are stored on diskette as strings anyway, one byte per number, because that's how Apple DOS formats diskette storage.)

After you finish entering a name and length for each item, you'll fall into the file editor. Record length is the sum of the field lengths plus the number of fields (because there is a return character after each field).

The file editor (lines 113-131) is basically a list editor. In BASIC, lists are virtually synonymous with arrays. It is the arrays holding the file description (L$ and L%) that get manipulated here. The edit menu uses abbreviations. (Replace the pound sign [#] with an integer.) I suggest you make a few simple typographical errors to see how the program responds. This is what the abbreviations mean:

S – saves/creates a data file and a description file.

R – review. Prints the file description.

A# – add # new items to the description (i.e., A3 = add three items).

D# – delete item # (i.e., D2 = delete item number 2).

I# – insert an item into position #.

N# – change the name of item number #.

L# – change the length of item number #.

Deleting and inserting items is done by shifting both description arrays; changing a name or length is done by entering a replacement for an element of one of the two arrays.

I once read somewhere that file maintenance consists of content changes and structural changes. The record editor described below takes care of content changes. *Evolvability*, the capacity to respond to changing needs, is accomplished through the file editor. If a check (line 114: is B>0?) shows that the file has data in it, you can still edit the file structure, although the program works a little harder. First, the original description is copied into some temporary arrays (line 114 again). When you're done monkeying around and you choose the Save option, the old and new descriptions are compared (lines 129-131). Next a scratch, or temporary, file is written to meet the new specifications (line 131).

Adding new items or changing an item name presents no problem. If an item is deleted it won't be rewritten; if an item is shortened, any instance of it that's too long gets truncated from the right. After the scratch file has been successfully completed, the old file is deleted and the scratch file is renamed. Apple Manager uses scratch files in a couple of other places, namely when sorting a file or deleting records. These routines also write a new, updated file before deleting the old file. You could run into a DISK FULL error if there weren't enough room. By the way, I've chosen the unlikely name of "A control D" for the scratch file's name, so it shouldn't interfere with any of your files.

The Other Choices

Perhaps you've asked yourself, "How does Apple Manager know when a file has data?" The method is simple. Record zero, the first record in a random access file, stores the number of records. This number is updated whenever records are added or deleted (line 64). A newly created or emptied file (see below) is actually one record on the diskette with a 48 (the ASCII code for zero) in the first byte.

No data manager would be complete without the ability to get rid of unwanted files. Apple Manager deletes the data file and the corresponding description file and removes the file name from the title array (lines 132-134). The title array may change several times in one run. Rather than rewrite the file each time, a flag variable, F, is set. The title file on the diskette is updated (line 66) when you exit the Files module – if the flag is set (see lines 5 and 223). This is why the disk drive light may come on when you switch modules.

If you don't want to remove a file (just reuse it), then the Empty function at lines 135-136 is for you. This section deletes a data file but not its description file, and opens a new data file of the same name. The net effect is to empty a file of data while preserving the file structure. Copy creates a new data file by using an already existing description file.

A few words about limits and error handling are undoubtedly in order here. The dimension statement in line two defines the first three of these somewhat arbitrary limits, so they can be easily changed:

- 25 files per diskette.
- 50 fields per record.
- 1000 records per file.
- 115 bytes = maximum field length.
- 20 characters maximum in a file or item name.

The last two limits, as well as many errors, are avoided by checking input: line 32 (Is a number out of range?), line 35 (Has the return key been pressed without first typing something?) and line 36 (Is a string too long?), are examples. But what if you try to make the program do something illegal, such as read data that doesn't exist? ONERR is meant for just such cases, though it does have the drawback of stopping the Apple's excellent error messages.

Here's how I compromised. If there's an error in line 89 – e.g., if there's no title file because it's the first time you're using a diskette – you jump to line 90 where the error flag for the rest of the program is set. From here on in, unless your error is one of the DOS errors dealt with by lines 220 and 221, the POKE 216,0 at line 222 cancels ONERR. RESUME causes the error to recur so you get an Apple message. Control C (program interrupt; error code =255) is handled differently; it still works, but without the expected BREAK message.

Using A Directory

Let's assume we've got an imaginary file defined and that data has been entered into it. Let's also assume we want to extract information about an author named Kilroy. One way to do it is to search the entire file until we find the Kilroy record. But disk access is slow and we are impatient, so let's use a directory to locate the record instead.

When a file is selected, Apple Manager opens the description file and reads it into arrays (line 67, called as a subroutine from line 94). The data file is also opened and the first field of every record is read into the array D$ (line 94). It's faster to search this directory array than to search the file. There is a one-to-one correspondence between array elements and records (Figure 6): if *Kilroy* is the seventh array element then record seven is the record we want.

What we have done is to define the first field – in this case *author* – as the record identifier. Record IDs are used for rapid access in Delete, Print, and Change (described below). A subroutine beginning at line 43 requests the ID and searches the directory. You don't have to enter the complete ID. For instance, repeatedly typing "K" will locate, in sequence, all authors whose name begins with K.

Assuming 48K of memory, there are about 17,000 free bytes for the directory. (The exact amount depends on how much is used up by the title array, the description array, and other string variables.) If the first item is a long one, it may not be possible to have 1000 records in the file without disabling the directory. The same memory problem may arise when you sort, since the D$ array holds the item being sorted.

You now have an outline of how the program works. We could step slowly and leisurely through the code, but I don't want to send the editors into apoplexy. The rest of this writeup is a guide to using the program. A good way to start is by creating a file or two. Then go to the Records module, enter some fictitious data, sort it, and edit it. Next try a report. Then go back to Files and change the file structure. Now generate another report to see how stored data has been affected.

Apple Manager makes a good, if rudimentary, stab at most data management functions. One omission is computed variables. This is not a short program, so I'll make copies for anyone who sends $3, a diskette, and a stamped, self-addressed mailer to: Robert Beck, 2101 21 Ave. S., #W15, MPLS, MN 55404. To those who are typing it in, I wish a steady hand and a steadier eye. Happy data managing.

Records Module

ENTER (lines 147-150) – Initializes each item to an asterisk (*) – so missing data is not a problem – then goes to the record editor (lines 8-27), which has five options:

1) Retype – type in a new value.
2) Control O – exit to menu.
3) Control B – back up one field in the file.
4) Control F – forward to next record.
5) RETURN – the return key must be pressed after each of the above options. Pressing the return key alone does not affect the item displayed; it moves you to the next item in the record.

Record number appears in parentheses to the left of the item name. Use Control B to backspace through a file; use RETURN to move forward through a file.

DELETE (lines 139-146) – marks records you choose to delete by placing a Control E in the first byte. When you exit this option via Control O, Apple Manager rewrites the file without marked records.

CHANGE (lines 151-158) – allows a "window" in each record to be set by selecting a starting and an ending item number. Once a starting record in the file is chosen, the record editor, which works as previously described, is called.

PRINT (lines 159-160) – prints one record at a time. This is the fastest way to retrieve a record (Figure 7).

SORT (lines 161-165) – reads the sort key (item to sort by) into the array D$, then sorts, in either ascending or descending order, by the bubble method. It's really a tag sort because the record numbers (in S%) are also sorted. Once record numbers are properly ordered, the file is rewritten. The sort is alphanumeric, so "17" is placed before "7" and after "07".

Report Module

1. Retrieve (lines 169-171, 190-198) – formats the report in tabular form if the printout from one record fits on one line (Figure 8), otherwise prints one item per line. A variable number of fields can be retrieved, and in any order.

2. With Sums – same report as Retrieve, with the addition (pun intended) that a sum for each item is printed.

3. Frequencies (lines 199-204) – counts the number of times each value of an item occurs. First, sort the file by that item.

4. Case Selection (lines 172-188) – a technique that lets you retrieve information by its characteristics – you can pick out a subset of the file. All you have to do is input selection criteria in the form of minimum and maximum item values. The values are stored in arrays (M$ and N$) and may be ORed together in groups of five or less. Up to five of these groups can be ANDed together:

(a1 OR b1 OR c1 OR d1 OR e1) AND ... AND (a5 OR b5 OR c5 OR d5 OR e5).

Each a1, b1, c1, etc. is of the form MIN(A)<=A=<MAX(A).

Got that? Well, here's how to work it:

1) Select an item by typing its number.

2) Select a range for that item by typing a minimum and a maximum. Pressing the return key without typing anything sets a minimum to the null string or a maximum to CHR$(95).

3) Terminate a series of ORs and go on to the next AND by typing a slash (/).

4) Terminate the whole thing at any point, including the very beginning, by typing a period.

Note: Each record is checked against the criteria stored in arrays (lines 76-79) and ignored if it doesn't meet them (i.e., if G=5). For an alternative, and very interesting, method of introducing changeable functions into your program, see "Algebra String – a Self-altering Program" in **COMPUTE!** (September, 1981, #16).

Utilities Module (lines 212-218)

Upload – Each record's fields are laid down sequentially within the record. There will be null bytes at the end of the record if any item is shorter than its defined byte length. Upload removes all null bytes from a file by adding blanks where needed. The name "Upload" comes from the fact that some mainframe computers interpret null bytes as end-of-record marks; to send diskette files to them null bytes must be removed.

Download – Does the opposite of Upload.

Drive Select – Use to switch from one disk drive to another. Each diskette remains an independent data base.

```
1  REM    APPLE MANAGER    **  ROBERT JAC
     QUES BECK   **
2 D$ =  CHR$ (4): FOR I = 768 TO 777: RE
     AD J: POKE I,J: NEXT : DIM L$(50),
     F$(4),T$(25),L%(50),D$(1000),R$(50)
     ,S%(1000),S(50),M$(25),N$(25): FOR
     I = 0 TO 4: READ F$(I): NEXT : GOTO
     88
3  VTAB 23: PRINT "PRESS RETURN TO CONT
     INUE";: CALL  - 958
4  POKE  - 16368,0: INPUT "->";Z$:H =
     RIGHT$ (Z$,1) =  CHR$ (15)
5  IF E = 7 AND F = 1 AND H THEN  GOSUB
     66
6  IF H THEN  POP : CALL 768: ON E GOTO
```

```
    96,104,166,212,137,152,91,115,144,
    156,172
7 Z = VAL (Z$): RETURN
8  FOR Y = O TO K
9  VTAB 19: HTAB 1: PRINT " (R"I")" SPC
    ( 2)Y". "L$(Y)":": PRINT : PRINT R
    $(Y)
10  HTAB 1: VTAB 21: INPUT "";Z$: IF Z$
```

Table : Partial list of variables.

Name	Description
A$	File name.
D$	Control D (DOS commands).
D$(1000)	Directory and sort array.
F$(4)	Option titles (Records).
L$(50)	Item (field) names.
L%(50)	Item lengths.
M$(25)	Minima of criteria (Report).
N$(25)	Maxima of criteria (Report).
R$	Current value of item being counted (frequencies).
R$(50)	Fields of current record.
S(50)	Item sums (Report).
S%(1000)	Sort array (holds record numbers).
S%(1-50)	Item numbers being reported.
S%(200-250)	Criteria item numbers (Report).
S%(400-450)	Spaces allotted to an item in report printout.
Z$	Input.
A	Number of fields per record.
B	Number of records.
C	Maximum permissible value.
	Frequency count of an item (frequencies).
D	Input error flag.
	Number of lines on screen.
E	Module flag.
F	Title file update flag.
G	Option flag (Files).
	G<>5 if record meets criteria (Report).
	Total sum (Report).
H	Control 0 flag.
I,J,K,Q	Temporary indices.
K	Ending item number (edit window).
L	String or item length.
N	Number of files.
O	Starting item number (edit window).
	Number of criteria (Report).
P	P = 1 if file name is in title file.
	Number of dashes to print.
Q	Record number.
	Larger of item length and length of item name (Line 74).
R	Record length.
S	Option selected (Report).
V	Drive number.
	Number of printer columns (Report).
W	Slot number.
X	Number of spaces needed for tabular report.
	First record to edit (Records).
Y	Number of items being reported.
	Item number being counted (frequencies).
	Error code.
Z	Numeric value of input.

```
    = "" THEN 15
11  IF Z$ =  CHR$ (2) THEN 23
12  IF Z$ =  CHR$ (6) OR Z$ =  CHR$ (15
    ) THEN X =  ASC (Z$): GOTO 18
13 L = L%(Y): VTAB 17: GOSUB 36: IF D T
    HEN 10
14 R$(Y) = Z$: VTAB 18: CALL  - 868
15  GOSUB 27: NEXT Y
16 Q = I: GOSUB 48:D$(I) = R$(1): IF X
    = 15 THEN  GOSUB 64: CALL 768: GOTO
    137
17  RETURN
18  IF V = 6 THEN 16
19 H = Y = 1 AND I = B + 1 AND R$(1) =
    "*": IF H AND X = 6 THEN 10
20  IF H AND X = 15 THEN  GOSUB 64: CALL
    768: GOTO 137
21  IF X = 15 THEN B = B + 1
22  GOSUB 27: GOTO 16
23  IF I = 1 AND Y < 2 THEN  VTAB 21: PR
    INT R$(Y): GOTO 10
24  IF Y > 1 THEN  GOSUB 27:Y = Y - 1:
    GOTO 9
25 Q = I: GOSUB 48:D$(I) = R$(1):I = I -
   - 1: IF V = 5 THEN  GOSUB 27:Y = K:
    GOSUB 47: GOTO 9
26  VTAB 6: CALL  - 958: PRINT "* ID= "D
    $(I): POKE 34, PEEK (37) + 1:Y = K:
     GOSUB 47: GOTO 9
27  HTAB 1: VTAB 21: PRINT R$(Y): FOR E
     = 1 TO  LEN (R$(Y)) / 40 + 4: CALL
      - 912: NEXT : RETURN
28  PRINT "** WHICH?";: POKE  - 16368,0:
      GET Z$: IF Z$ < >  CHR$ (15) THEN
      Z =  VAL (Z$): RETURN
29 H = 1: GOTO 5
30 C =A
31  GOSUB 4
32  IF Z < 1 OR Z > C OR  INT (Z) < > Z
      OR ( VAL ( RIGHT$ (Z$,1)) =0 AND
      RIGHT$ (Z$,1) < > "0") THEN PRINT
       "TYPE AN INTEGER FROM 1 TO "C;:W
      = 1: GOTO 31
33  RETURN
34  GOSUB 4
35 D = 0: IF Z$ = "" THEN  PRINT : PRINT
      "TYPE SOMETHING BEFORE PRESSING
      RETURN!":D = 1: RETURN
36 D = 0: IF  LEN (Z$) > L THEN  PRINT :
      PRINT "THAT IS TOO LONG, TRY AG
      AIN":D =1
37  RETURN
38 I =  VAL ( MID$ (Z$,2)): IF (I > A)
      THEN PRINT "NUMBER OUT OF RANGE-
       START OVER!": POP : GOTO 116
39 I =  VAL ( MID$ (Z$,2)): IF I = 0 OR
      ( VAL ( RIGHT$ (Z$,1)) = 0 AND R
      IGHT$ (Z$,1) < > "0") THEN PRINT
       "TYPO, TRY AGAIN!": POP : GOTO
      116
40  RETURN
41  PRINT : PRINT "NAME OF ITEM "I"?";:
      GOSUB 34:L$(I) = Z$: IF D = 1 TH
      EN 41
42  PRINT "ITEM LENGTH?";:C = 115: GOSUB
      31:L%(I) = Z:R = + Z + 1: RETURN
43  PRINT :L = 115: PRINT "RECORD ID?";:
```

```
        GOSUB 34:U = I:J = LEN (Z$): IF
        D = 1 THEN 43
44  IF U = B THEN U = 0
45  FOR I = U + 1 TO B: IF  LEFT$ (D$(I)
        ,J) < > Z$ THEN  NEXT : IF U > 0
        THEN U = 0: GOTO 45
46  IF I > B THEN  PRINT : PRINT "ID NOT
        FOUND, TRY ANOTHER!": GOTO 43
47  PRINT D$"READ"A$",R"I: FOR J = 1 TO
        A: INPUT R$(J): NEXT : PRINT D$:
        RETURN
48  PRINT D$"WRITE"A$",R"Q: FOR J = 1 TO
        A: PRINT R$(J): NEXT : PRINT D$:
        RETURN
49  PRINT D$"WRITEA"D$",R"Q: FOR J = 1 T
        O  A: PRINT R$(J): NEXT : PRINT
        D$: RETURN
50  IF W = 0 AND D / 15 < >  INT (D / 15
        ) THEN  PRINT : PRINT : GOSUB 3:
        VTAB 23: HTAB 1: CALL  - 868
51  RETURN
52  PRINT
53 D = D +1: IF W = 0 AND D / 15 =  INT
        (D / 15) THEN  PRINT : PRINT : P
        RINT : GOSUB 3: VTAB 21: HTAB 1:
        CALL  - 958
54  RETURN
55  PRINT L$(Q)": "R$(Q): FOR J = 1 TO
        INT (( LEN (L$D(Q) + R$(Q))) / 4
        0) + 1: GOSUB 53: NEXT :RETURN
56 M$ = D$(J):D$(J) = D$(J - 1):D$(J - 1
        ) = M$:K = S%(J):S%(J) = S%(J -
        1):S%(J - 1) = K:K = 1: RETURN
57  PRINT : PRINT "FILE NAME?";:L = 20:
        GOSUB 34: IF D = 1 THEN 57
58 A$ = Z$: FOR I =1 TO N: IF Z$ = T$(I)
        THEN P = 1: RETURN
59  NEXT :P = 0: IF G < 2 THEN  PRINT :
        PRINT Z$" ISN'T IN THE TITLE": P
        RINT "FILE, TRY AGAIN.": GOSUB 3
60  RETURN
61  GOSUB 63: IF B >  0 THEN  PRINT : P
        RINT "FILE "Z$" HAS DATA -": IF
        P < 3 THEN  POP : GOSUB 3: GOTO
        96
62  RETURN
63  GOSUB 65: PRINT D$"READ"A$",R0": IN
        PUT B : PRINT D$: RETURN
64  GOSUB 65: PRINT D$"WRITE"A$",R0": P
        RINT B: PRINT D$: RETURN
65  PRINT : PRINT D$"OPEN"A$",L"R: RETU
        RN
66  PRINT : PRINT D$"OPEN TITLE FILE":
        PRINT D$"WRITE TITLE FILE": PRI
        NT N: FOR I = 1 TO N: PRINT T$(
        I): NEXT : PRINT D$"CLOSETITLE
        FILE":F =0: RETURN
67  PRINT D$"OPENDES "A$: PRINT D$"READ
        DES "A$: INPUT A,R: FOR I = 1 T
        O A: INPUT L$(I),L%(I): NEXT :
        PRINT D$"CLOSE": RETURN
68  HOME : PRINT "FILE NAME: "A$: PRINT
        : PRINT A" ITEMS PER RECORD": P
        RINT : PRINT "RECORD LENGTH: "R
69  PRINT : PRINT "#" SPC( 6)"ITEM" SPC
        ( 20)"LENGTH":P = 40: GOSUB 82:
        PRINT : RETURN
70  GOSUB 68: FOR I = 1 TOA: PRINT I"."
        SPC( 6 -  LEN ( STR$ (I)))L$(I)
        SPC( 26 -  LEN (L$(I)))L%(I): P
        RINT : IF (I - 6) / 8 =  INT ((
        I - 6) / 8) AND (I < > A) AND W
        = 0 THEN  GOSUB 3: HOME : GOSUB
        69
71  NEXT : PRINT D$"PR#0": RETURN
72  FOR W = U TO O * 5:M$(W) =  CHRS$ (
        95):S%(200 + W) = 0: NEXT : RET
        URN
73  VTAB 23 - C: CALL  - 958: PRINT : R
        ETURN
74 J = S%(I):Q = L%(J): IF  LEN (L$(J))
        > L%(J) THEN Q =  LEN (L$(J))
75 X = Q + X + 2:S%(400 + I) = Q: RETURN
76 G = 0: GOSUB 47: IF O = 0 THEN  RETU
        RN
77  FOR K = 1 TO O:G = 0: FOR J = 1 TO
        5:U = (K - 1) * 5 + J:Q = S%(20
        0 + U): IF R$(Q) < M$(U) OR R$(
        Q) > N$(U) THEN G = G + 1
78  NEXT : IF G = 5 THEN  RETURN
79  NEXT : RETURN
80  PRINT : PRINT "SEND PRINTOUT TO SLO
        T NUMBER?":P = 12: GOSUB 82: PR
        INT "DEFAULT = TV": GOSUB 82: V
        TAB  PEEK (37) - 3: HTAB 30 : G
        OSUB 4:W = Z: CALL  - 958: RETU
        RN
81  PRINT D$"PR#"W: RETURN
82  FOR J = 1 TO P: PRINT "-";: NEXT :
        PRINT : RETURN
83  HOME : VTAB 2: PRINT  TAB( 13)F$(Z
        - 1)" RECORDS"
84  VTAB 4: PRINT "FILE NAME: "A$: POKE
        34,5: VTAB 7: RETURN
85  IF B = 0 THEN  RETURN
86  FOR J = 1 TO V: IF L$(I) < > D$(J)
        THEN  NEXT : RETURN
87 D$(J) = Z$: RETURN
88 V = 1: ONERR  GOTO 90
89  PRINT D$"OPENTITLE FILE,D"V: PRINT
        D$"READTITLE FILE": INPUT N: IF
        N > 0 THEN  FOR I = 1 TO N: INP
        UT T$(I): NEXT
90  PRINT D$"CLOSE": ONERR  GOTO 219
91  TEXT : HOME : VTAB 2: PRINT  SPC( 1
        0)  APPLE MANAGER   ": VTAB 7:
        HTAB 3: PRINT "1  FILES" SPC( 1
        2)"4  REPORTS": PRINT : HTAB 3:
        PRINT "2  SELECT FILE" SPC( 6)"
        5 UTILITIES": PRINT : HTAB 3: P
        RINT "3  RECORDS" SPC( 10)"6 QU
        IT"
92 E = 7: VTAB 20: GOSUB 28: ON Z GOTO
        96,93,137,166,212,223: PRINT :
        PRINT "YOU CAN'T CHOOSE THAT! T
        RY AGAIN": GOTO 92
93 O = 1:E = 7: HOME : VTAB 4:G = 0: PR
        INT  TAB( 11)"-- SELECT A FILE
        --": VTAB 6: GOSUB 57: IF P = 0
        THEN 93
94  GOSUB 67: GOSUB 63: IF B > 0 THEN F
        OR I = 1 TO B: PRINT D$"READ"A$
        ",R"I: INPUT D$(I): NEXT : PRIN
```

```
     T D$
95  GOTO 91
96 O = 2:L = 20:G = 0:P = 0: HOME : VTA
     B 2:E = 7: PRINT  SPC( 13)" <<
     FILES  >>": VTAB 6: PRINT : PRI
     NT "1  CATALOG" SPC( 10)"5 EDIT
     DESCRIPTION": PRINT : PRINT "2
     DESCRIBE FILE" SPC( 4)"6  EMPTY
     FILE": PRINT : PRINT "3  CREATE
     FILE" SPC( 6)"7  COPY DESCRIPTI
     ON"
97  PRINT : PRINT "4  DELETE FILE" SPC(
     6)"8  QUIT"
98  VTAB 22: GOSUB 28:E = 1: ON Z GOTO
     99,101,104,132,113,135,106,223:
     FLASH : PRINT : PRINT "TYPE A N
     UMBER FROM 1 TO 8": NORMAL : GO
     TO 98
99  HOME : PRINT  TAB( 9)"CATALOG OF DA
     TA FILES": PRINT :D = 0:W = 0:
     IF N = 0 THEN  GOSUB 3: GOTO 96
100  FOR I = 1 TO N: PRINT T$(I): GOSUB
     53: NEXT : GOSUB 50: GOTO 96
101  HOME : PRINT  TAB( 9)"** DESCRIBE
     FILE **": PRINT : GOSUB 57: IF
     P = 0 THEN 101
102  GOSUB 67: GOSUB 63: GOSUB 80: GOSU
     B 81: GOSUB 70: IF W = 0 THEN
     GOSUB 3
103  GOTO 96
104 E = 1: HOME : PRINT  TAB( 7)"** CRE
     ATE A NEW FILE **": PRINT :G =
     2: GOSUB 57: IF P = 1 THEN  GOS
     UB 67: GOSUB 61
105  PRINT "NUMBER OF ITEMS PER RECORD?
     ";:C = 50: GOSUB 31:A = Z:R = 0
     : FOR I = 1 TO A: GOSUB 41: NEX
     T :B = 0: GOTO 115
106  HOME : PRINT  TAB( 18)"COPY": PRIN
     T : PRINT "OLD";: GOSUB 57: IF
     P = 0 THEN 106
107  GOSUB 67: PRINT :G = 7:B = 0: PRIN
     T "NEW";: GOSUB 57: IF P = 1 TH
     EN  GOSUB 61
108  IF B > 0 THEN 129
109  VTAB 24: FLASH : PRINT "CREATING F
     ILE "A$: NORMAL
110  PRINT D$"OPENDES "A$: PRINT D$"DEL
     ETEDES "A$: PRINT D$"OPENDES "A
     $: PRINT D$"WRITEDES "A$: PRINT

     A: PRINT R: FOR I = 1 TO A : PR
     INT L$(I): PRINT L%(I): NEXT :
     PRINT D$"CLOSE"
111  FOR I = 1 TO N: IF A$ < > T$(I) TH
     EN  NEXT :N = N + 1:T$(N) = A$
     :F = 1
112  GOSUB 64: GOTO 96
113 G = 1: HOME : PRINT  TAB( 10)"## ED
     IT DESCRIPTION ##": GOSUB 57: I
     F P = 0 THEN 113
114  GOSUB 67:P = 3: GOSUB 61: PRINT :
     IF B > 0 THEN FOR I = 1 TO A:D$
     (I) = L$(I):S%(I) = L%(I): NEXT
115 R$ = "SRADINL":P = 39: GOSUB 82: PR
     INT  TAB( 3)"SAVE, REVIEW, ADD,
     DELETE, INSERT,": PRINT  TAB( 3
     )"CHANGE NAME, OR CHANGE LENGTH
     ......"
116 E=G: PRINT "(S, R, A#, D#, I#, N#,
     OR L#)";: GOSUB 4:E = 8:I = 1
117  IF  LEFT$ (Z$,1) =  MID$ (R$,I,1)
     OR I > 7 THEN 119
118 I = I + 1: GOTO 117
119  ON I GOTO 108,120,121,123,126,127,
     128: FLASH : VTAB 23: PRINT "HE
     Y!";: NORMAL : GOTO 116
120  GOSUB 70: GOTO 116
121  GOSUB 39: PRINT : PRINT "ADD "I" I
     TEMS:":Y = A + I: IF Y > 50 THE
     N  PRINT "YOU CAN'T HAVE MORE T
     HAN 50 ITEMS!": GOTO 116
122  FOR I = A + 1 TO Y: GOSUB 41:A = I
     : NEXT : GOTO 116
123  GOSUB 38: PRINT : PRINT : PRINT "D
     ELETE: "L$(I) TAB( 28)"LENGTH:
     "L%(I):R =R - L%(I) - 1: IF (I
     = A) THEN 125
124  FOR J = I TO A - 1:L$(J) = L$(J +
     1):L%(J) = L%(J + 1): NEXT
125 A = A - 1: GOTO 116
126  GOSUB 38:M$ = L$(I):Y = L%(I): GOS
     UB 41:A = A + 1: FOR J = A TO I
     + 2 STEP  -1:L$(J) = L$(J - 1):
     L%(J) = L%(J - 1): NEXT : L$(I
     + 1) = M$:L%(I + 1) = Y: GOTO 1
     16
127  GOSUB 38: PRINT "OLD NAME: "L$(I):
     PRINT : PRINT "NEW NAME?";: GOS
     UB 34: GOSUB 85:L$(I) = Z$: GOT
     O 116
128  GOSUB 38: PRINT L$(I)"  >>>  LENGT
     H IS "L%(I): PRINT : PRINT "NEW
     LENGTH?";:C = 115: GOSUB 31:R =
     R - L%(I) + Z:L%(I) = Z: GOTO 1
     16
129  HOME : FLASH : PRINT "REWRITING":
     NORMAL : FOR I = 1 TO A:S%(100
     + I) = 0: FOR J = 1 TO V: IF D$
     (J) = L$(I) THEN S%(100 + I) =
     J: GOTO 131
130  NEXT J
131  NEXT I:D$(0) = "*": PRINT D$"OPENA
     "D$",L"R: FOR Q = 1 TO B: PRINT
     D$"READ"A$",R"Q: FOR J = 1 TO V
     : INPUT D$(J): NEXT : FOR J = 1
     TO A:R$(J) =  LEFT$ (D$(S%(100
     + J)),L%(J)): NEXT : GOSUB 49:
     NEXT : PRINT D$"DELETE"A$: PRIN
     T D$"RENAMEA"D$","A$: GOTO 110
132  HOME : PRINT : PRINT  TAB( 9);: FL
     ASH : PRINT "###";: NORMAL : PR
     INT "  DELETE FILE  ";: FLASH :
     PRINT "###": NORMAL : VTAB 8: G
     OSUB 57: IF P = 0 THEN 132
133  IF I < > N THEN  FOR J = I TO N -
     1:T$(J) = T$(J + 1): NEXT
134  PRINT D$"OPEN"A$: PRINT D$"DELETE"
     A$: PRINT D$"OPENDES "A$: PRINT
     D$"DELETEDES "A$:F = 1:N = N -
     1: GOTO 96
135  HOME : PRINT : PRINT  TAB( 10)"EMP
     TY A FILE": GOSUB 57: IF P = 0
```

```
        THEN 135
136  PRINT D$"OPEN"A$: PRINT D$"DELETE"
        A$: GOSUB 67:B = 0: GOSUB 64: G
        OTO 96
137  PRINT : PRINT D$"PR#0": TEXT : HOM
        E : VTAB 2:E = 7: PRINT  SPC( 1
        1)"<<<  RECORDS  >>>": VTAB 8:
        PRINT "1 ENTER RECORDS" SPC( 6)
        "4 PRINT RECORDS": PRINT : PRIN
        T "2 DELETE RECORDS" SPC( 5)"5
        SORT": PRINT : PRINT "3 CHANGE
        RECORDS" SPC( 5)"6  QUIT"
138  VTAB 22: GOSUB 28: PRINT :E =5: ON
        Z GOTO 147,139,151,159,161,223:
        FLASH : PRINT : "TYPE A NUMBER
        FROM 1 TO 6!": NORMAL : GOTO 138
139 E = 9: GOSUB 83:I = 0
140  GOSUB 43: PRINT : PRINT "DELETED:
        ";: FOR J = 1 TO A: PRINT R$(J)
        " ";: NEXT : PRINT : PRINT "IS
        THIS WHAT YOU WANT DELETED?"
141  PRINT "(Y OR N)";: GOSUB 4: IF Z$
        = "Y" THEN  PRINT D$"WRITE"A$",
        R"I: PRINT  CHR$ (5): PRINT D$:
        GOTO 140
142  IF Z$ = "N" THEN 140
143  FLASH : PRINT "HEY!";: NORMAL : GO
        TO 141
144  HOME : HTAB 12: FLASH : PRINT "REW
        RITING": NORMAL : PRINT D$"OPEN
        A"D$",L"R:K = B:B = 0: FOR I =
        1 TO K: GOSUB 47: IF  LEFT$ (R$
        (1),1) < >  CHR$ (5) THEN B = B
        + 1:Q = B: GOSUB 49:D$(B) = R$(
        1)
145  NEXT
146  PRINT D$"DELETE"A$: PRINT D$"RENAM
        EA"D$","A$: GOSUB 64: GOTO 137
147  GOSUB 83
148 V = 5:0 = 1:K = A: FOR I = 1 TO A:R
        $(I) = "*": NEXT : I = B + 1
149  GOSUB 8: IF I < B + 1 THEN I = I +
        1: GOSUB 47: GOTO 149
150 B = B + 1: GOTO 148
151  GOSUB 83
152 E = 5: PRINT : PRINT "STARTING ITEM
        NUMBER? (DEFAULT= 2)";: GOSUB 4
        :E = 6: IF Z$ = "" THEN O = 2:
        GOTO 154
153 C = A: GOSUB 32:O = Z
154  PRINT - PRINT "ENDING ITEM NUMBER?
        ": PRINT "(DEFAULT= LAST ONE)";
        : GOSUB 4: IF Z$ = "" THEN K =
        A: GOTO 156
155 K = Z:C = A: GOSUB 32: IF K <
        O THEN PRINT : PRINT "THE LAST
        ITEM NUMBER MUST BE AT LEAST "O
        : GOTO 154
156 E = 6: PRINT : PRINT "STARTING ID?
        (DEFAULT= FIRST ONE)";: GOSUB 4
        :E = 10: IF Z$ = "" THEN X = 1:
        GOTO 158
157 J = LEN (Z$): FOR X = 1 TO B: IF  L
        EFT$ (D$(X),J) < > Z$ THEN  NEX
        T : PRINT : PRINT "ID NOT FOUND
        , TRY ANOTHER!": GOTO 156
158 V = 6: FOR I = X TO B: GOSUB 47: VT
        AB 6: CALL  - 958: PRINT "*ID=
        "D$(I): POKE 34, PEEK (37) + 1:
        GOSUB 8: NEXT I: GOSUB 64: GOTO
        137
159  GOSUB 83: GOSUB 80:I = 0
160  PRINT D$"PR#0": GOSUB 43: GOSUB 81
        : PRINT : PRINT :Z =0:D =0: FOR
        Q = 1 TO A: GOSUB 55: NEXT : GO
        TO 160
161  GOSUB 83: PRINT "NUMBER OF ITEM TO
        SORT BY?";: GOSUB 30: PRINT : P
        RINT "I WILL SORT BY "L$(Z): PR
        INT :V = Z - 1: PRINT : PRINT "
        ASCENDING (1)": PRINT "OR DESCE
        NDING (2) ORDER?";:C = 2: GOSUB
        31
162  HOME : FLASH : PRINT "SORTING": NO
        RMAL : FOR I = 1 TO B:S%(I) = I
        : PRINT D$"READ"A$",R"I: PRINT
        D$"POSITION"A$",R"V: PRINT D$"R
        EAD"A$: INPUT D$(I): NEXT : FOR
        I = B TO 2 STEP  - 1:K = 0: FOR
        J = 2 TO I: IF Z = 1 AND D$(J -
        1) > D$(J) THEN  GOSUB 56
163  IF Z = 2 AND D$(J) > D$(J - 1) THE
        N  GOSUB 56
164  NEXT : IF K THEN  NEXT
165  PRINT D$"OPENA"D$",L"R: FOR Q = 1
        TO B:I = S%(Q): GOSUB 47: GOSUB
        49:D$(Q) = R$(1): NEXT : GOTO 1
        46
166  PRINT : PRINT D$"PR#0": TEXT :E =
        7: HOME : HTAB 15: PRINT "+ REP
        ORTS +": GOSUB 84: PRINT : HTAB
        9: PRINT "1  RETRIEVE": PRINT :
        HTAB 9: PRINT "2  WITH SUMS": P
        RINT : HTAB 9: PRINT "3  FREQUE
        NCIES": PRINT : HTAB 9: PRINT "
        4 QUIT"
167  VTAB 21: GOSUB 28:C = 4: PRINT : G
        OSUB 32:S = Z: IF Z = 4 THEN 22
        3
168  HOME :E = 3: IF Z = 3 THEN 199
169 X = 0: PRINT "REPORT TITLE?";: GOSU
        B 4:M$ =Z$: PRINT "HOW MANY ITE
        MS TO PRINT?": PRINT "(DEFAULT
        = ALL)";: GOSUB 4: IF Z$ < > ""
        THEN 171
170 Y = A: FOR I = 1 TO Y:S%(I) = I:S(I
        ) = 0: GOSUB 74: NEXT : GOTO 172
171 C = A: GOSUB 32:Y = Z: PRINT "TYPE I
        N ITEM NUMBERS ONE AT A TIME.":
        PRINT : FOR I = 1 TO Y: CALL  -
        868: PRINT  SPC( 16)"<-";: HTAB
        13:F =  PEEK (37): GOSUB 30: VT
        AB F + 1: HTAB 1: CALL  - 868:
        HTAB 7: PRINT L$(Z):S(I) = 0:S%
        (I) = Z: GOSUB 74: NEXT
172 P = 40: GOSUB 82: HTAB 13: PRINT "S
        ELECT CASES":E = 3: PRINT : PRI
        NT "TYPE AN ITEM NUMBER,THEN IT
        S MINIMUM ANDMAXIMUM VALUES (IN
        THAT ORDER!). PRESS  RETURN AFT
        ER EACH  ONE.": PRINT : PRINT "
        TYPE A SLASH (/) TO SKIP THE 'O
```

```
        R'."
173  PRINT "TO IGNORE A MINIMUM OR A MA
        XIMUM, MERELYPRESS RETURN.": PR
        INT : PRINT "TO FINISH, TYPE A
        PERIOD (.), THEN PRESS  RETURN.
        ":E = 3:O = 0: PRINT
174 O = O + 1: PRINT : FOR K = 1 TO 5:U
        = (O - 1) * 5 + K: PRINT O"." S
        PC( 3)"ITEM NUMBER?";: GOSUB 4:
        E = 11: IF Z$ = "." THEN C = 2:
        ON K GOTO 186: GOSUB 73: GOTO 1
        88
175  IF Z$ = "/" AND K > 1 THEN C = 3:
        GOSUB 73: GOSUB 72: GOTO 184
176 C = A:W = 0: GOSUB 32:S%(200 + U) =
        Z:L = L%(Z): VTAB 23 - W: HTAB
        5: CALL  - 958: PRINT "** "L$(Z)
177  PRINT  TAB( 7)"MINIMUM?";: GOSUB 4
        : GOSUB 36: IF D = 1 THEN  PRIN
        T : GOTO 177
178 M$ (U) = Z$: IF Z$ = "." THEN C = 3
        : ON K GOTO 186: GOSUB 73: GOTO
        188
179  PRINT  TAB( 7)"MAXIMUM?";: GOSUB 4
        : GOSUB 36: IF D = 1 THEN PRINT
        : GOTO 179
180 N$(U) = Z$: IF Z$ = "" OR Z$ = "."
        THEN N$(U) =  CHR$ (95)
181  IF Z$ = "." THEN U = U + 1: ON K G
        OTO 188,188,188,188,189
182  IF K < 5 THEN PRINT : PRINT TAB( 8
        )"- OR -": PRINT
183  NEXT
184  IF O < 5 THEN PRINT : PRINT TAB( 8
        )"+ AND +": GOTO 174
185  IF O = 5 THEN 189
186 O = O - 1: IF O = 0 THEN C = 1
187  GOSUB 73: GOTO 189
188  GOSUB 72
189  IF S = 3 THEN 200
190  GOSUB 80:V = 40: IF W > 0 THEN PRI
        NT "HOW MANY COLUMNS PER LINE?"
        ;: GOSUB 4:V =Z
191  GOSUB 81: PRINT : PRINT : PRINT :
        PRINT  SPC( 10)M$: PRINT: PRINT
        :PRINT :D = 0:I = 0: IF X < V T
        HEN P = X: FOR J = 1 TO Y: PRIN
        T L$(S%(J)) SPC( 2 + S%(400 + J
        ) -  LEN (L$(S%(J))));: NEXT :
        PRINT : GOSUB 82
192 I = I + 1: GOSUB 76: IF G = 5 THEN
        195
193  IF X < V THEN  FOR K = 1 TO Y:Q =
        S%(K): PRINT R$(Q); SPC( 2 + S%
        (400 + K) -  LEN (R$(Q)));:S(K)
        = S(K) + VAL (R$(Q)): NEXT : GOS
        UB 52: GOTO 195
194  GOSUB 52: GOSUB 52: GOSUB 52: FOR
        K = 1 TO Y:Q = S%(K): GOSUB 55:
        S(K) = S(K) + VAL (R$(Q)): NEXT
195  IF I < B THEN 192
196  IF X < V AND S = 2 THEN  GOSUB 82:
        GOSUB 53: GOSUB 52: FOR K = 1 T
        O Y:G = INT (S(K) * 100 + .5) /
        100: PRINT G; SPC( S%(40  0 + K
        ) + 2 - LEN ( STR$ (G));: NEXT
        : GOSUB 52: GOTO 205
197  IF S = 2 THEN GOSUB 52: GOSUB 52:
        GOSUB 52: PRINT SPC( 10)"SUMS:"
        : GOSUB 53:GOSUB 52: GOSUB 52:
        FOR K = 1 TO Y:Q = S%(K) : PRIN
        T L$(Q)": " INT (S(K) * 100 + .
        5) / 100: GOSUB 53: NEXT : GOTO
        205
198  GOTO 205
199  PRINT : PRINT "ITEM TO COUNT?";: G
        OSUB 30:Y = Z: PRINT : PRINT A$
        " MUST BE SORTED": PRINT "BY ";
        : FLASH : PRINT L$(Y): NORMAL :
        GOTO 172
200  GOSUB 80: GOSUB 81:R$ = "":D = 0:C
        = 0:S%(I) = Y:X = 0: GOSUB 74:X
        = X + 5:P = X + 6: IF W = 0 THE
        N P = 40
201  PRINT : PRINT : PRINT L$(Y) SPC( X
        - 3 - LEN (L$(Y)))"FREQUENCY":
        GOSUB 82: PRINT: FOR I = 1 TO B
        : GOSUB 76: IF G = 5 THEN 204
202  IF R$(Y) < > R$ AND I < > 1 THEN P
        RINT R$ SPC( X -  LEN (R$))C:R$
        = R$(Y): GOSUB 53:C = 1: GOTO
        204
203 C = C + 1: IF I = 1 THEN R$ = R$(Y)
204  NEXT : PRINT R$ SPC( X - LEN (R$))
        C: GOSUB 53
205  GOSUB 50: IF O < 1 THEN 166
206  PRINT : PRINT : PRINT : FOR Q = 1
        TO 30: PRINT "*";: NEXT : PRINT
        : PRINT "THESE CRITERIA WERE US
        ED:": PRINT : PRINT :I = 0
207 I = I + 1: FOR K = 1 TO 5:U = (I -
        1) * 5 + K:Q = S%(200 + U): PRI
        NT L$(Q): PRINT SPC( 5)"MIN:"M$
        (U): PRINT SPC( 5)"MAX:"N$(U):D
        = 14: GOSUB 52: IF S%(201 + U)
        = 0 THEN 210
208  IF K < 5 THEN  PRINT : PRINT  SPC(
        8)"- OR -": PRINT
209  NEXT K
210  IF I < O THEN PRINT : PRINT  "----
        - AND": PRINT : GOTO 207
211  GOTO 166
212  HOME :E = 7: PRINT  TAB( 15)"UTILI
        TIES": VTAB 7: HTAB 9: PRINT "1
        UPLOAD": PRINT : HTAB 9: PRINT
        "2 DOWNLOAD": PRINT : HTAB 9: P
        RINT "3 DRIVE SELECT": PRINT :
        HTAB 9: PRINT "4 QUIT"
213  VTAB 18: GOSUB 28:C = 4: PRINT : G
        OSUB 32: ON Z GOTO 214,216,218,
        223
214  GOSUB 84: GOSUB 65: FOR I = 1 TO B
        : GOSUB 47: FOR J = 1 TO A: IF
        LEN (R$(J)) < L%(J) THEN FOR K
        = 1 TO L%(J) - LEN (R$(J)):R$(J
        ) = R$(J) + " ": NEXT
215  NEXT :Q = I: GOSUB 48: NEXT : GOSU
        B 65: GOTO 91
216  GOSUB 84: GOSUB 65: FOR I = 1 TO B
        : GOSUB 47: FOR J = 1 TO A: FOR
        K = L%(J) TO 1 STEP  - 1: IF MI
        D$ (R$(J),K,1) < > " " THEN R$(
```

```
      J) = LEFT$ (R$(J),K):K=1
217  NEXT : NEXT :Q = I: GOSUB 48: NEXT
      : GOSUB 65: GOTO 91
218  HOME : VTAB 6:E = 4: PRINT TAB( 5)
      "DISK DRIVE FOR DATA FILES?";:C
      = 2: GOSUB 31:V = Z:N = Ø: GOTO
      89
219 Y =  PEEK (222): IF Y = 255 THEN 223
220  IF Y = 5 THEN HOME : VTAB 1Ø: PRIN
      T A$" IS EMPTY!": GOSUB 3: CALL
      768: GOTO 91
221  TEXT : IF Y = 6 OR Y = 11 OR Y = 2
      OR Y = 3 THEN CALL 768: GOTO 93
222  POKE 216,Ø: RESUME
223  IF F = 1 THEN GOSUB 66
224  TEXT : CALL  - 868: HTAB 17: FLASH
      : PRINT "SO LONG": NORMAL
225  DATA 1Ø4,168,1Ø4,162,223,154,72,15
      2,72,96,ENTER, DELETE, CHANGE,
      PRINT, SORT
```

©

If you have an Original or an Upgrade BASIC PET, you'll find this repeating-keys program frequently useful, especially when you need to make corrections to a large program.

PET Auto Repeat

Art Hunkins
School of Music
University of North Carolina at Greensboro
Greensboro, NC

These programs were adapted from several sources. SYS889 enables them both. The same command also *disables* the repeat function, which, incidentally, works for *all* keys. Auto repeat must be disabled for cassette functions to operate.

In the Original ROM version, POKE914,(30) specifies the .5 second delay time (hold time) before the character begins to repeat. POKE932, (4) specifies the repeat rate. Either value can be changed. For Upgrade ROMs, the locations are, respectively, POKE927,(30) and POKE945,(4).

Both programs store in the second cassette buffer. Whenever the need for extensive program editing arises, an auto repeat function is a real timesaver.

Program 1.

```
1ØØ REM FOR ORIGINAL ROMS
11Ø DATA12Ø,56,169,233,237,26,2,141,26
12Ø DATA2,88,96,173,3,2,2Ø1,255
13Ø DATA2Ø8,12,169,Ø,141,119,3,169
14Ø DATA3Ø,141,12Ø,3,2Ø8,3Ø,238,119
15Ø DATA3,173,12Ø,3,2Ø5,119,3,176,19
16Ø DATA169,4,141,12Ø,3,169,Ø,141,3,2
17Ø DATA141,119,3,169,2,141,37,2,24
18Ø DATA76,133,23Ø
19Ø FORI=889TO952:READJ:POKEI,J:NEXT
```

Program 2.

```
1ØØ REM FOR UPGRADE ROMS
11Ø DATA12Ø,56,169,233,229,145,133,145
12Ø DATA165,144,2Ø1,46,2Ø8,6,169,147
13Ø DATA133,144,2Ø8,4,169,46,133,144
14Ø DATA88,96,165,151,2Ø1,255
15Ø DATA2Ø8,12,169,Ø,141,119,3,169
16Ø DATA3Ø,141,12Ø,3,2Ø8,28,238,119
17Ø DATA3,173,12Ø,3,2Ø5,119,3,176,17
18Ø DATA169,4,141,12Ø,3,169,Ø,133,151
19Ø DATA141,119,3,169,2,133,168,24
2ØØ DATA76,46,23Ø
21Ø FORI=889TO963:READJ:POKEI,J:NEXT
```

©

A grab bag of tricks and VIC techniques to make programming easier and safer.

VIC Curiosities

Doug Ferguson
Elida, OH

Here's a potpourri of odd things I discovered by accident on the keyboard of the VIC-20. I hope you find something useful.

Cold Start By SYS 64802

You, too, may hate turning your VIC off and on to clear all the funny POKEs you've made or to get a clean start after strange happenings. Save your power switch by typing SYS 64802. Of course, you'll still have to power down the usual way in the event of a crash or lock-up.

One-handed RUN

It is already generally known that some operations may be initiated using one hand. You can stop a program by merely hitting the RUN/STOP key. And you might know the quick way to LOAD: just hold down the left shift key with your thumb and touch the RUN/STOP key. But how does one initiate a RUN with one hand?

Easy. Type quotation marks followed by left-shifted RUN/STOP. I usually hold down the left shift key with my thumb and touch the "2" key and then the RUN/STOP key in sequence with my middle finger of the same hand. Try it! You'll see a quick flash of an error and then a RUN will begin. A lot easier than typing RUN and hitting RETURN all of the time. Incidentally, if you want a mysterious-looking two-handed RUN, try SYS 50830.

LIST Killer

Would you like to prevent nosy people from reading your program? Add a line to POKE 755,200. Unfortunately, it only works if they RUN it before they try to LIST it. By the way, don't let this trick prevent *you* from listing your own stuff. POKE 755,199 to restore the LIST function.

SAVE Killer

This is the LIST killer's big brother. (Like its sibling, it only works if the program is first RUN.) It also kills RUN/STOP and the RESTORE key in order to prevent easy reversal. To make it work, add this line to your program: POKE 802,0: POKE 803,0: POKE 818,165. This should discourage the casual thief. Remember, you cannot use your RUN/STOP key. To undo this little trick POKE 802,243: POKE 803,243: POKE 818,133. ©

A Light Pen For Under $10

William Hale, Albuquerque, NM

The light pen capabilities of the VIC-20 can be put to use for less than $10 (or for less than $2 if you have a connector to mate with game port connector). All that is needed is a ballpoint pen case or fine tip marker pen case, a photodiode or phototransistor, a resistor, three to four feet of shielded cable, and a 9-pin female connector.

The phototransistor can be a Radio Shack Catalog No. 276-138 ($.89) or a Sylvania ECG-3038 ($1.25). The ECG-3038 is the smaller of the two and fits closer to the fine tip of a marker pen. A 1/8-watt resistor, 1K for the ECG-3038 and 100K for the RS 276-138, is connected between the collector of the phototransistor and +5V located on Pin 7 on the game port connector. The collector of the phototransistor is also connected to the light pen input on Pin 6 of the game port connector and its emitter to ground, located on Pin 8.

The phototransistor is mounted in the tip of a pen case as near to the opening as possible and is tied to the connector via a shielded cable. Solder the collector to the center conductor and the emitter to the shield. A pushbutton can be added between the emitter and the shield at the pen to prevent false triggering. The other end of the shielded cable will, of course, be connected to the game port mating connector where the load resistor should be located between Pins 6 and 7. A one shot multivibrator could be added between the phototransistor output and Pin 6 in case the CRT of the user's monitor or TV screen is weak. In my case, I found it was not necessary with either of the light pens I constructed.

The principle of operation is quite simple. With the pen touching the screen, the scanning beam of the CRT causes the phototransistor to produce a negative pulse. This pulse is used by the VIC-20 to latch into memory locations 36870 and 36871 the horizontal and vertical position of the scan line via numbers ranging from 0 to 128. I was able to obtain repeatable readings out of the vertical location (36871), but the horizontal (36870) was not as reliable. In my case, I feel this was due to a slight 60H, ripple in my CRT sweep circuitry causing the scan line pulse to change periodic rate.

By PEEKing into these locations, a user could recognize the scan line position and branch a program accordingly. Uses could be a menu, listings, multiple choice answers, and for chase or move-the-target type games. ©

This hybrid (a mixture of BASIC and machine language) utility program will quickly locate a target within a large Atari "superstring."

Substring Search Utility

Edward C Smith
Harrisburg, PA

The Atari can handle very long strings of data. However, searching for a substring within a long string is slow when the search is executed entirely in BASIC. A combination of BASIC and machine language results in greater speed. Both methods are presented below.

BASIC A+ [*sold by Optimized Systems Software*] employs an instruction called "FIND," which searches a long string to find a substring at a specified starting location. If you don't have BASIC A+, you can achieve similar results by utilizing a subroutine that combines BASIC and machine code.

Program 1 incorporates two subroutines for comparison purposes. A long string of data (5592 bytes) is created to permit searching for any of 100 records. If you RUN the program, the prompt "ENTER SUBSTRING" will appear. Respond by entering "RECORD #100." After the next prompt "ENTER START LOCATION," respond with "1." The next prompt "ENTER SELECTION 700 or 800." If you respond with 800, you have selected the BASIC subroutine to perform the search operation. If you start your stopwatch immediately after entering 800, it should take approximately 98 seconds until the answer "SEARCH RESULT = 5536" appears. This means that RECORD #100 starts at the 5536th byte of the long string (Y$). Confirmation of the result is indicated by the next line printed – "FOUND STRING = RECORD #100."

You can prove this to yourself by typing (in the direct mode): ?Y$(5536,5546). Now repeat the same procedure, except after the prompt "ENTER SELECTION 700 or 800," respond with "700." You should receive the same answer in less than one second. You may wish to try other substrings. All answers are referenced to the beginning of the main string.

Program Operation

Line 10 defines the size of all strings used. Y$ is the main string. X$ is the substring and should not exceed 255 bytes. DAT$ and R$ are used in construction of the main string.

Lines 20 and 25 create a long string (Y$) to simulate 100 records, numbered RECORD #1 to RECORD #100 (5592 bytes).

Line 40 loads the 92 machine code bytes.

Lines 50 to 65 are the input prompts. A starting location less than one is assumed to be one.

Line 70 directs execution of chosen subroutine.

Lines 80 to 85 are possible search results. If no substring is found, the search result is zero.

Lines 700 to 770 are the subroutine combining BASIC and machine language. Inputs required for the machine language portion defined by the USR function are: (1) Main string Y$, (2) Substring X$, (3) Length of Y$, (4) Length of X$-1, and (5) Starting location.

Lines 800 to 870 are the subroutine written entirely in BASIC. Inputs required for this routine are identical to those used for subroutine 700.

Lines 900 to 920 check for abnormal entries.

Lines 20000 to 21050 load the machine code bytes into page six of memory.

Program 2 is a liberally remarked listing of the assembled machine code.

Program 1.

```
5 REM SUBSTRING SEARCH PROGRAM
6 REM BY EDWARD C. SMITH
7 REM APRIL 6,1982
10 DIM Y$(6000),X$(255),DAT$(80),R$(9)
12 REM Y$ IS THE MAIN STRING - X$ IS THE SUB
   STRING - DAT$ AND R$ ARE USED TO DEVELOP
   THE MAIN STRING
15 REM LINES 20 TO 25 CREATE A LARGE STRING.
     PRINT Y$ TO SEE THIS STRING.
20 R$="RECORD #":DAT$=" NAME....ADDRESS....C
   ITY....STATE....PHONE...."
25 FOR I=1 TO 100:Y$(LEN(Y$)+1)=R$:Y$(LEN(Y$
   )+1)=STR$(I):Y$(LEN(Y$)+1)=DAT$:NEXT I
40 GOSUB 20000:REM LOAD MACHINE CODE BYTES
45 ? :REM SUBSTRING X$ MUST BE LESS THAN 255
    BYTES.
47 REM IF YOU ENTER 'RECORD #100',START LOCA
   TION 1, AND GO BASIC SUBROUTINE 800 SEARC
   H TIME WILL BE 98 SECONDS.
50 ? :? "ENTER SUBSTRING";:INPUT X$
52 ?
55 ? "ENTER SEARCH START LOCATION";:INPUT A1
57 ?
60 ? "700 BASIC PLUS MACHINE CODE"
62 ? "800 BASIC CODE ONLY"
63 ?
65 ? "ENTER SELECTION,700 or 800";:INPUT SR
67 ?
70 GOSUB SR:REM THE VALUE "A" RETURNED IS TH
   E # OF BYTES TO THE RIGHT OF (A1-1)
80 ? CHR$(253);"SEARCH RESULT=";A1+A-1
83 IF A=0 THEN ? "STRING NOT FOUND":GOTO 50
85 ? "FOUND STRING=";Y$(A1+A-1,A1+A+LX-2)
90 GOTO 50
700 REM STRING SEARCH USING BOTH BASIC AND M
    ACHINE CODE TOGETHER
705 LY=LEN(Y$):LX=LEN(X$):POKE 207,LX-1
```

```
710 GOSUB 900:IF A=0 THEN 770
720 B=LY-LX-A1+3
730 A=USR(1664,ADR(Y$(A1)),ADR(X$),B)
740 IF A=0 THEN A1=1
770 RETURN
800 REM STRING SEARCH USING BASIC ONLY
805 LY=LEN(Y$):LX=LEN(X$)
810 GOSUB 900:IF A=0 THEN 870
820 FOR I=1 TO LY-LX-A1+2
830 IF Y$(A1+I-1,A1+I+LX-2)=X$ THEN 850
840 NEXT I
845 A=0:A1=1:GOTO 870
850 A=I
870 RETURN
900 REM CORRECT START LOCATION A1 IF ENTERED
  VALUE IS OUT OF RANGE
905 A=1:IF A1<1 THEN A1=1
910 IF A1>LY-LX+1 OR LX>LY THEN A=0:A1=1
920 RETURN
20000 REM LOAD 92 MACHINE CODE BYTES
20005 FOR I=1664 TO 1755:READ A:POKE I,A:NEX
T I:RETURN
20008 DATA 104,104,133,204,104,133
20010 DATA 203,104,133,206,104,133
20020 DATA 205,104,141,222,6,104
20030 DATA 141,221,6,169,1,133
20040 DATA 212,169,0,133,213,160
20050 DATA 255,200,177,203,209,205
20060 DATA 240,40,24,165,203,105
20070 DATA 1,133,203,165,204,105
20080 DATA 0,133,204,24,165,212
20090 DATA 105,1,133,212,165,213
21000 DATA 105,0,133,213,205,222
21010 DATA 6,208,216,165,212,205
21020 DATA 221,6,208,209,240,7
21030 DATA 152,197,207,208,204,240
21040 DATA 6,169,0,133,212,133
21050 DATA 213,96
```

Program 2.

```
10 ; SUBSTRING SEARCH PROGRAM
20 ;   BY EDWARD C. SMITH
30 ;      APRIL 6,1982
40 ; CALLED FROM BASIC BY
50 ; A=USR(1664,ADR(Y$(A1)),
   ADR(X$),B)
60 ; WHERE
70 ;  Y$ IS THE MAIN STRING
     LOCATED AT ADDRESS YD
80 ;
90 ;  X$ IS THE SUBSTRING
     LOCATED AT ADDRESS XD
0100 ;
0110 ;LX=LENGTH OF X$-1
0120 ;
0130 ;A1 IS THE START OF
      SEARCH MEASURED FROM
      LEFT END OF Y$
0140 ;
0150 ;B=LY-LX-A1+3
0160 ;A2 IS THE RESULT
      0 IF NOT FOUND
0170 ;
0180 ;NOTE:LX-1 MUST BE
      POKED AT 207 ($CF)
0190  *=$680
0200 YD=$CB
0210 XD=$CD
0220 LX=$CF
0230 A2=$D4
0240  PLA
0250  PLA ;GET ADDRESS OF MAIN STRING
0260  STA YD+1
0270  PLA
0280  STA YD
0290  PLA ;GET ADDRESS OF SUBSTRING
0300  STA XD+1
0310  PLA
0320  STA XD
0330  PLA ;GET ADDR OF MAX POSSIBLE COMPARES
0340  STA B+1
0350  PLA
0360  STA B
0370 ;INITIALIZE TO 1ST     BYTE OF MAIN STR
    ING
0380  LDA #$01
0390  STA A2
0400  LDA #$00
0410  STA A2+1
0420 START LDY #$FF
0430 NEXT INY
0440 CMP1 LDA (YD),Y ;COMPARE YTH BYTE OF MA
    IN STRING VS SUBSTRING
0450  CMP (XD),Y
0460  BEQ CMP2
0470  CLC
0480 ; MOVE TO NEXT BYTE     IN MAINSTRING
0490  LDA YD
0500  ADC #$01
0510  STA YD
0520  LDA YD+1
0530  ADC #$00
0540  STA YD+1
0550  CLC
0560 ;UPDATE RESULT LOCATOR
0570  LDA A2
0580  ADC #$01
0590  STA A2
0600  LDA A2+1
0610  ADC #$00
0620  STA A2+1
0630  CMP B+1 ;IS MAX ALLOWABLE COMPARES REA
    CHED?
0640  BNE START
0650  LDA A2 ;                    YES ON HIGH BY
    TE
0660  CMP B ;IS MAX ALLOWABLE COMPARES REACH
    ED?
0670  BNE START
0680  BEQ NOMATCH ;                     YES ON L
    OW BYTE
0690 CMP2 TYA ;HAVE ALL BYTES OF SUBSTRING B
    EEN LOOKED AT?
0700  CMP LX
0710  BNE NEXT ;NO
0720  BEQ RETN ;YES
0730 NOMATCH LDA #$00
0740  STA A2
0750  STA A2+1
0760 RETN RTS
0770 B=RETN+$02
```

©

A Monthly Column

All computer users can benefit from this month's column – many of Bill's observations and hints are not specific to the Atari. If you're thinking of translating a BASIC game program into machine language to achieve greater speed, you'll find some valuable information below. For example, there's a discussion of the "ball/boarder" problem which can be the most difficult puzzle to solve when programming certain kinds of games.

Insight Atari

Bill Wilkinson
Optimized Systems Software
Cupertino, CA

This month we return to the world of program writing. As I noted in my last column, there has been a growing demand for me to explain how to write graphics programs in assembly language. So I will begin a two or three-part series this month on converting BASIC programs to assembly language. Although the programs will be specifically written for the Atari computers, it won't take too much imagination to convert them to Apple and Commodore machines.

The Bouncing BASIC Ball

Since we are going to try to build up this program in stages, we will start this month with the simplest possible form. Program 1 is an Atari BASIC program which bounces a "ball" around inside the rectangular screen. There is no scoring, no paddles, no sound, no players, no missiles, no intelligence.

In fact, perhaps the only thing which needs explaining is the frequent occurrence of the subexpression: INT(n * RND (0)). With Apple Integer BASIC, one could obtain the equivalent function by coding RND(n); and I have often wished that Atari had let us include that capability in the original specifications for Atari BASIC (oh, well, maybe in the next version of BASIC A+?). Anyway, the idea is to produce an integer random number in the range of 0 to n-1, inclusive.

So now let's examine the program as a whole. (First, a comment: I have used the convention that X means "horizontal" and Y means "vertical." This

Program 1. Simple Bouncing Ball Program

```
100 GRAPHICS 3
200 XMOVE=INT(5*RND(0))-2
300 YMOVE=INT(5*RND(0))-2
400 IF XMOVE+YMOVE=0 THEN 200
500 X=INT(40*RND(0))
600 Y=INT(20*RND(0))
700 XNEW=X:YNEW=Y
900 POKE 19,0:POKE 20,0:REM RESET TIMER
1000 REM LOOP STARTS HERE
1100 COLOR 0:PLOT X,Y
1200 COLOR 2:PLOT XNEW,YNEW
1300 X=XNEW:Y=YNEW
1400 XNEW=X+XMOVE:YNEW=Y+YMOVE
1500 IF XNEW<=0 OR XNEW>=39 THEN XMOVE=-XMOV
     E
1600 IF XNEW<0 OR XNEW>39 THEN XNEW=X
1700 IF YNEW<=0 OR YNEW>=19 THEN YMOVE=-YMOV
     E
1800 IF YNEW<0 OR YNEW>19 THEN YNEW=Y
1900 IF PEEK(19)=0 THEN 1000
2000 RUN
```

Program 2. Bouncing Ball Initialization

```
0000        20          .PAGE 'initialization'
            30 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            40 ;
            50 ; A SIMPLE BOUNCING BALL PROGRAM
            60 ;
            70 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            80 ;
0000        90          *=   $3000
            0100 LINE100
            0101 ;>>>
                        GRAPHICS 3

3000 A903   0110        LDA  #3
3002 20E830 0120        JSR  GRAPHICS
            0197 ;
            0198 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0199 ;
            0200 LINE200
            0201 ;>>>
                        XMOVE=INT(5*RND(0))-2

3005 A904   0210        LDA  #4
3007 205731 0220        JSR  RND         ; GET RANDOM NUMBER FROM 0 TO 4
300A 38     0230        SEC
300B E902   0240        SBC  #2          ; NOW IS RANDOM FROM -2 TO +2
300D 8DE230 0250        STA  XMOVE       ; AS IN BASIC PROGRAM
            0297 ;
            0298 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0299 ;
            0300 LINE300
            0301 ;>>>
                        YMOVE=INT(5*RND(0))-2

3010 A904   0310        LDA  #4
3012 205731 0320        JSR  RND         ; GET RANDOM NUMBER FROM 0 TO 4
3015 38     0330        SEC
3016 E902   0340        SBC  #2          ; NOW IS RANDOM FROM -2 TO +2
3018 8DE330 0350        STA  YMOVE       ; AS IN BASIC PROGRAM
            0397 ;
            0398 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0399 ;
            0400 LINE400
            0401 ;>>>
```

can have some strange implications. See below.) We start by establishing the least detailed graphics mode (which is, incidentally, roughly equivalent to Apple's LO-RES mode). Then we set both of the variables XMOVE and YMOVE to a random number in the range -2 to +2, inclusive. (Do you see how? 'INT(5*RND(0))' gives a number from zero to four, inclusive, and we then subtract two from it.) But we don't allow both values to be zero (line 400). (In a real "Pong" type game, you wouldn't want the X-motion to ever be zero. Here, allowing XMOVE to be zero is instructive.)

We then give the ball a starting position with X in the range of 0 to 39 and with Y from 0 to 19. Note that both the current position (X and Y) and the to-be-made-current position (XNEW and YNEW) are set equal. This is simply to get things started evenly. Line 900 resets the system timer. (You will have to do something differently here if you are using an Apple.)

The main loop is almost as simple. First, we erase (COLOR 0) the old "ball" (note that we are erasing nothing if this is the first time through the loop). Then we PLOT the new ball with a convenient, visible color (COLOR 2). We update our current ball position (line 1300) and also our to-be-made-current position (line 1400).

It Gets A Bit Difficult

Here is where it begins to get tricky. If the ball will be *at or beyond* the edge(s) of the screen, we must reverse its movement, as appropriate (lines 1500 and 1700). But suppose that the movement has *already* carried it beyond the screen bounds; we must then bring it back inbounds (lines 1600 and 1800). Finally, for this simple demo,

```
                                IF XMOVE+YMOVE=0 THEN 200

301B ADE230 0410        LDA  XMOVE
301E 18     0420        CLC
301F 6DE330 0430        ADC  YMOVE      ; XMOVE + YMOVE
3022 F0E1   0440        BEQ  LINE200    ; IF = 0  THEN 200
            0497 ;
            0498 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0499 ;
            0500 LINE500
            0501 ;>>>
                        X=INT(40*RND(0))

3024 A927   0510        LDA  #39
3026 205731 0520        JSR  RND        ; GET RANDOM NUMBER FROM 0 TO 39
3029 8DDE30 0530        STA  X          ; AND KEEP IT
            0597 ;
            0598 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0599 ;
            0600 LINE600
            0601 ;>>>
                        Y=INT(20*RND(0))

302C A913   0610        LDA  #19
302E 205731 0620        JSR  RND        ; GET RANDOM NUMBER FROM 0 TO 19
3031 8DDF30 0630        STA  Y          ; AND KEEP IT
            0697 ;
            0698 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0699 ;
            0700 LINE700
            0701 ;>>>
                        XNEW=X : YNEW=Y

3034 ADDE30 0710        LDA  X
3037 8DE030 0720        STA  XNEW       ; XNEW = X
303A ADDF30 0730        LDA  Y
303D 8DE130 0740        STA  YNEW       ; YNEW = Y
            0897 ;
            0898 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0899 ;
            0900 LINE900
            0901 ;>>>
                        POKE 19,0:POKE 20,0

3040 A900   0910        LDA  #0
3042 8513   0920        STA  19
3044 8514   0930        STA  20         ; DON'T NEED TO DO LDA #0 TWICE
            0997 ;
            0998 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0999 ;
            1000 LINE1000
            1001 ;>>>
                         REM LOOP STARTS HERE

            1097 ;
            1098 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1099 ;
            1100 LINE1100
            1101 ;>>>
                         COLOR 0 : PLOT X,Y

3046 A900   1110        LDA  #0
3048 201531 1120        JSR  COLOR
304B AEDE30 1130        LDX  X
304E ACDF30 1140        LDY  Y          ; LOAD VALUES FOR SUBROUTINE CALL
3051 202031 1150        JSR  PLOT
            1197 ;
            1198 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1199 ;
            1200 LINE1200
            1201 ;>>>
                         COLOR 2 : PLOT XNEW,YNEW

3054 A902   1210        LDA  #2
3056 201531 1220        JSR  COLOR
3059 A900   1230        LDA  #0         ; (NEEDED FOR PLOT)
305B AEE030 1240        LDX  XNEW
305E ACE130 1250        LDY  YNEW
3061 202031 1260        JSR  PLOT
            1297 ;
            1298 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1299 ;
            1300 LINE1300
```

ATARI® Software for Personal Computers

• BOWLER'S DATABASE (New)	Req. — 16K	RAM/Cassette	$14.95
	— 24K	RAM/Disk	14.95
• LEAP FROG (New)	Req. — 16K	RAM/Cassette	14.95
	— 16K	RAM/Disk	14.95
• PLAYER PIANO (New)	Req. — 32K	RAM/Cassette	17.95
	— 40K	RAM/Disk	17.95
• 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
• WARSHIPS	Req. — 16K	RAM/Cassette	14.95
	— 24K	RAM/Disk	19.95
• CCA Data Management System	Req. — 40K	RAM/Disk	99.95
• LETTER WRITER	Req. — 24K	RAM/Disk	19.95

® Trademark of ATARI, Inc.

DIVISION OF CUSTOM ELECTRONICS, INC.
CE 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

ZAP

THE BIG MATH ATTACK™

POW

Challenging new math program . . .
Sharpen your skills by entering the correct answer before the equation 'lands' on your city! Provides hours of educational entertainment.

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.

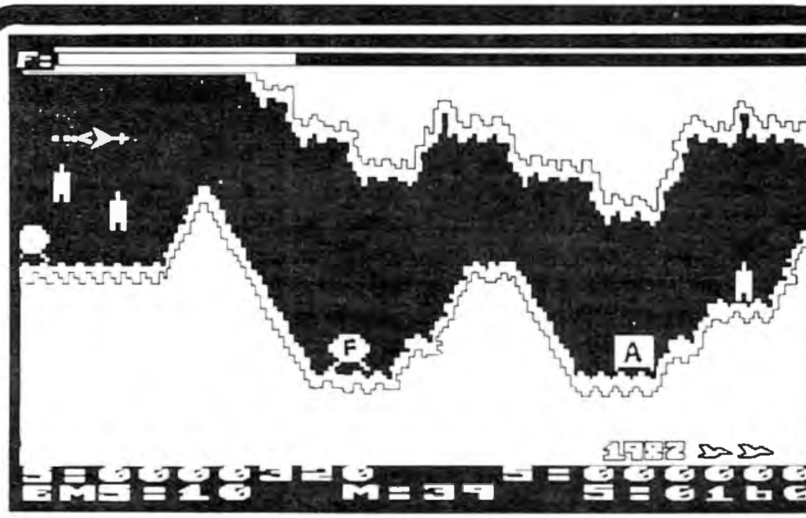
or order direct:
T.H.E.S.I.S.
P.O. Box 147
Garden City, MI 48135
(313) 595-4722

Please add:
$1.50 shipping, handling
$1.50 C.O.D.
WRITE FOR FREE CATALOG
DEALER INQUIRIES WELCOME

COMPUTE!

Is Looking For FORTH Screens: Applications, Utilities, and Programming Techniques

---



AIRSTRIKE

NEW! ©AIRSTRIKE FOR THE ATARI 400/800 16K

- INTRODUCING THE NEW ARCADE-STYLE GAME FOR ATARI COMPUTERS!
- SUPERB FINE SCROLLING GRAPHICS AND DAZZLING ARRAY OF COLOUR.
- FIGHT YOUR WAY THROUGH CAVERNS AND TUNNELS WITH ATTACKING MISSILES, FUEL AND AMMO DUMPS TO CONTEND WITH!
- 100% MACHINE CODE PROGRAM FOR ONE OR TWO PLAYERS.
- MULTIPLE SKILL LEVELS.

AVAILABLE NOW DIRECT ON 16K CASSETTE : $39.95 PLUS $5 AIRMAIL POSTAGE. SEND YOUR VISA/MASTERCARD NUMBER TODAY!

ATARI IS A T.M. OF ATARI INC.

DEALERS! DON'T DELAY CONTACT US TODAY TO BECOME A STOCKIST OF AIRSTRIKE! MORE INCREDIBLE PROGRAMS DUE SHORTLY

ENGLISH SOFTWARE COMPANY
DEPT.C. 50 NEWTON STREET, PICCADILLY, MANCHESTER M1 2EA, ENGLAND, U.K. Tel: 061-236 7259

we simply do this loop until the clock ticks (4.26 seconds, roughly) and then start all over.

Even ignoring the limited goals of this program, there are a few significant flaws: (1) There is no visible border around the screen to tell you when and where the ball will "hit." (2) There are no sound effects. (3) The ball isn't round (or even remotely so). (4) Sometimes, the ball rebounds without hitting the wall. I am going to leave (1) and (2) for next time, and (3) can't really be changed without using player-missile graphics. But flaw (4) is an interesting one, and worth some discussion.

The problem lies in the basic algorithm I chose for moving the ball: the X and Y movements can range from -2 to +2 units, independently, and I move the ball each time in both X and Y according to the current movement factors (XMOVE and YMOVE). Let's take an example: suppose that the XMOVEment is zero and the YMOVEment is -2. And further suppose that the ball is currently at Y position +1 (one square from the edge of the screen). If I allow the ball to move to the new Y position determined by Y and YMOVE (YNEW= Y+YMOVE in line 1400), then it will be off the screen (YNEW will be -1). What to do?

One solution might be to pretend we have absorbent walls (IF YNEW<0 THEN YNEW=0). This will work, but will give strange flight paths for the ball. The solution I chose was to imagine that the ball hit the wall smack in the middle the two times I chose to make it visible. (Imagine: the ball is displayed at Y position +1. One-half of a time-tick later, it hits the wall and rebounds. Another one-half of a time-tick later, it has rebounded back out

```
               1301 ;>>>
                               X=XNEW : Y=YNEW

3064 ADE030 1310          LDA  XNEW
3067 8DDE30 1320          STA  X
306A ADE130 1330          LDA  YNEW
306D 8DDF30 1340          STA  Y
            1397 ;
            1398 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1399 ;
            1400 LINE1400
            1401 ;>>>
                               XNEW=X+XMOVE : YNEW=Y+YMOVE

3070 ADDE30 1410          LDA  X
3073 18     1420          CLC
3074 6DE230 1430          ADC  XMOVE
3077 8DE030 1440          STA  XNEW
307A ADDF30 1450          LDA  Y
307D 18     1460          CLC
307E 6DE330 1470          ADC  YMOVE
3081 8DE130 1480          STA  YNEW
            1497 ;
            1498 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1499 ;
            1500 LINE1500
            1501 ;>>>
                 IF XNEW<=0 OR XNEW>=39 THEN XMOVE=-XMOVE

3084 ADE030 1510          LDA  XNEW
3087 3006   1515          BMI  THEN1500  ;XNEW < 0
3089 F004   1520          BEQ  THEN1500  ;XNEW = 0
308B C927   1525          CMP  #39
308D 9009   1530          BCC  LINE1600  ;XNEW NOT >= 39
            1550 THEN1500
308F A900   1555          LDA  #0
3091 38     1560          SEC
3092 EDE230 1565          SBC  XMOVE     ;GET 0 - XMOVE
3095 8DE230 1570          STA  XMOVE     ; TO XMOVE
            1597 ;
            1598 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1599 ;
            1600 LINE1600
            1601 ;>>>
                 IF XNEW<0 OR XNEW>39 THEN XNEW=X

3098 ADE030 1610          LDA  XNEW
309B 3006   1620          BMI  THEN1600  ;XNEW < 0
309D C927   1630          CMP  #39
309F F008   1640          BEQ  LINE1700  ;XNEW = 39
30A1 9006   1650          BCC  LINE1700
            1660 THEN1600
30A3 ADDE30 1670          LDA  X
30A6 8DE030 1680          STA  XNEW      ; XNEW = X
            1697 ;
            1698 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1699 ;
            1700 LINE1700
            1701 ;>>>
                 IF YNEW<=0 OR YNEW>=19 THEN YMOVE=-YMOVE

30A9 ADE130 1710          LDA  YNEW
30AC 3006   1715          BMI  THEN1700  ;YNEW < 0
30AE F004   1720          BEQ  THEN1700  ;YNEW = 0
30B0 C913   1725          CMP  #19
30B2 9009   1730          BCC  LINE1800  ;YNEW NOT >= 19
            1750 THEN1700
30B4 A900   1755          LDA  #0
30B6 38     1760          SEC
30B7 EDE330 1765          SBC  YMOVE     ;GET 0 - YMOVE
30BA 8DE330 1770          STA  YMOVE     ; TO YMOVE
            1797 ;
            1798 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1799 ;
            1800 LINE1800
            1801 ;>>>
                 IF YNEW<0 OR YNEW>39 THEN YNEW=Y

30BD ADE130 1810          LDA  YNEW
30C0 3006   1820          BMI  THEN1800  ;YNEW < 0
30C2 C913   1830          CMP  #19
30C4 F008   1840          BEQ  LINE1900  ;YNEW = 39
30C6 9006   1850          BCC  LINE1900
```

to Y position + 1. We thus display it again at position + 1, since we are displaying only at integral time-ticks.) This choice is reflected in the programming in lines 1600 and 1800.

Of course, all "motion" via a computer is no more true motion than is a motion picture or a television picture. In truth, you are simply seeing a series of still pictures flashed in front of your eyes so quickly that your brain perceives the result as motion. Thus, there is nothing inherently wrong with my solution. Except that, from BASIC, the time between pictures is so long that even my lazy brain can sometimes clearly see that the ball didn't touch the wall. (Notice that if XMOVE is zero, so that we have only vertical ball movement, the effect is even easier to see.)

Can we do better? From BASIC, probably not. From assembly language, probably yes. If we choose a different algorithm, a different graphics mode, or make the pictures change faster, maybe we can give better illusions of motion. But that will wait for next time. This month, we will simply recode our BASIC routine in assembly language.

Having A Ball With Assembly Language

First note that the BASIC line numbers have been preserved, with line 100 in the assembly code having the label LINE100 and being followed, on line 101, with a remark containing the BASIC source for that line. (If you want to make your listings neat and readable, you might try the trick I used here: I placed a control-J [an ASCII line-feed character] both before and after the BASIC source. It can make your listing much more readable.)

Also note the inclusion of my graphics subroutines from the February issue of **COMPUTE!**

```
               1860 THEN1800
30C8 ADDF30    1870         LDA   Y
30CB 8DE130    1880         STA   YNEW        ; YNEW = Y
               1897 ;
               1898 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               1899 ;
               1900 LINE1900
               1901 ;>>>
                              IF PEEK(19)=0 THEN 1000

30CE A513      1910         LDA   19
30D0 D009      1920         BNE   LINE2000
               1930 ;====   LEAVE THESE 6 LINES OUT FIRST TIME ====
30D2 A514      1940         LDA   20          ; LSB OF CLOCK
               1950 CLOCKWAIT
30D4 C514      1960         CMP   20          ; CHANGED YET?
30D6 F0FC      1970         BEQ   CLOCKWAIT ; NO...WAIT
               1980 ;====   BUT ALWAYS KEEP LINE 1990 ====
30D8 4C4630    1990         JMP   LINE1000
               1997 ;
               1998 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               1999 ;
               2000 LINE2000
               2001 ;>>>
                              RUN

               2010 ;
               2020 ; to truly simulate RUN, we should store
               2030 ; zero to all variables.  For this program
               2040 ; that is not necessary, since all variables
               2050 ; are reset in the beginning of the program
               2060 ; anyway.  In general, though, be careful
               2070 ; to check such things.
               2080 ;
30DB 4C0030    2090         JMP   LINE100
               3000 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               3010 ;
               3020 ; !!!   VARIABLES AND SUBROUTINES !!!
               3030 ;
               3040 ; There are no direct BASIC equivalents
               3050 ; for the following lines ... the BASIC
               3060 ; interpreter handles all this for you
               3070 ; automatically.
               3080 ;
               3090 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               3100 ;
               3110 ; first -- the variable declarations
               3120 ;
30DE 00        3130 X       .BYTE 0
30DF 00        3140 Y       .BYTE 0
30E0 00        3150 XNEW    .BYTE 0
30E1 00        3160 YNEW    .BYTE 0
30E2 00        3170 XMOVE   .BYTE 0
30E3 00        3180 YMOVE   .BYTE 0
               3190 ;
               3200 ; second -- the subroutines
               3210 ;
               3220 ; these are mostly the same subroutines that
               3230 ; we presented in the February, 1982, issue
               3240 ; though they have been modified slightly.
               3250 ; Note that SETCOLOR and LOCATE have been
               3260 ; dropped (not used in this program)
               3270 ; and a new function, RND, has been added.
               3280 ;
               3290 ;
30E4           3300         .INCLUDE #D:GRAPHICS.ASM
```

Equates, etc., for graphics subroutines

```
30E4           9000         .PAGE 'Equates, etc., for graphics subroutines'
               9005 ;
               9010 ; CIO EQUATES
               9015 ;
E456           9020 CIO     =     $E456      ; Call OS thru here
0342           9025 ICCOM   =     $342       ; COMmand to CIO in IoCb
0344           9030 ICBADR  =     $344       ; Buffer or filename ADdRess
0348           9035 ICBLEN  =     $348       ; Buffer LENgth
034A           9040 ICAUX1  =     $34A       ; AUXilliary byte # 1
034B           9045 ICAUX2  =     $34B       ; AUXilliary byte # 2
               9050 ;
0003           9055 COPN    =     3          ; Command OPeN
000C           9060 CCLOSE  =     12         ; Command CLOSE
```

(Issue #21). I have added a RaNDom function, to make the mainline code easier and more compatible with the BASIC original. Even if you choose not to type in the mainline assembly language this month, you should type in and preserve these routines. Or simply add RND to the listing you typed in from February (you *did* type all that in, of course). We will use these same routines in the later articles in this series, but the listing will *not* be repeated.

As much as possible, the assembly language is self-explanatory, especially when coupled with the BASIC source. For example, what could be clearer than the translation of "GRAPHICS 3" into "LDA #3" and "JSR GRAPHICS"? If you don't understand *why* this works, you really need to get a good introductory book and read up on 6502 assembly language. For those of you into such things, you might note that when we convert from BASIC to assembly language, we tend to convert expressions by using reverse Polish notation. Thus, for example, line 300's assembly language equivalent might be expressed in "pidgin-HP" (that is, in a parody of the keyboard language used by HP reverse Polish calculators) as something like this:

4 RND 2 – ENTER xmove STORE

And those of you into FORTH will presumably also see the obvious corollaries.

The assembly language coding here is not the best nor the most efficient. For example, lines 410 through 430 could be replaced by a simple "ORA XMOVE" (because the A-register already contains YMOVE and because we don't really need the sum to find out if the two values are both zero). Rather, the idea here was to do as straightforward a translation as possible, allowing more of

```
000B          CPBINR =     11          ; Command Put BINary Record
0011          CDRAW  =     17          ; Command DRAWto
              ;
0004          OPIN   =     4           ; OPen for INput
0008          OPOUT  =     8           ; OPen for OUTput
              ;
              ;
              ; EQUATES used by the S: driver and
              ;   . the VBLANK routines
              ;
0055          HORIZONTAL = $55
0054          VERTICAL =   $54
02FB          DRAWCOLOR = $2FB
02C4          COLOR0 =     $2C4
              ;
              ; miscellany
              ;
00FF          LOW    =     $FF
0100          HIGH   =     $100
D20A          RANDOM =     $D20A
```

The graphics subroutines

```
30E4                        .PAGE 'The graphics subroutines'
              ;
              ;
30E4 00       SAVECOLOR .BYTE 0        ; where COLOR is saved
              ;
30E5 53       SNAME   .BYTE "S:",0     ; the filename for open
30E6 3A
30E7 00
              ;
              ;
              ; GRAPHICS g
              ;
              ; ENTRY: A-reg contains graphics mode 'g'
              ;   EXIT: Y-reg has completion status
              ;
              GRAPHICS
30E8 48                     PHA                ; save 'g'
30E9 A260                   LDX   #6*$10       ; file 6
30EB A90C                   LDA   #CCLOSE
30ED 9D4203                 STA   ICCOM,X
30F0 2056E4                 JSR   CIO          ; First, we must close file #6
              ; (we ignore any errors from the close)
              ;
30F3 A260                   LDX   #6*$10       ; again, file 6
30F5 A903                   LDA   #COPN        ; we will open this 'file'
30F7 9D4203                 STA   ICCOM,X
30FA A9E5                   LDA   #SNAME&LOW
30FC 9D4403                 STA   ICBADR,X     ; we use the file name "S:"
30FF A930                   LDA   #SNAME/HIGH
3101 9D4503                 STA   ICBADR+1,X ; by pointing to it
              ;
              ; all is set up for OPEN, now
              ; we tell CIO (and S:) what kind of open
              ;
3104 68                     PLA                ; our saved 'g' graphics mode
3105 9D4B03                 STA   ICAUX2,X     ; is given to S:
              ; (note that S: ignores the upper bits of AUX2)
3108 29F0                   AND   #$F0         ; now we get just the upper bits
310A 4910                   EOR   #$10         ; and flip bit 4
              ; (Read the text.   S: expects this bit inverted
              ;     from what normal BASIC usage is.)
310C 090C                   ORA   #$0C         ; allow read and write access (f
                                               or CIO)
310E 9D4A03                 STA   ICAUX1,X     ; make CIO and S: happy
3111 2056E4                 JSR   CIO          ; and do the OPEN of S:
3114 60                     RTS
              ;
              ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              ;
              ; COLOR c
              ;
              ;   ENTER: Color 'c' in A-register
              ;     EXIT: Unchanged
              ;
              COLOR
3115 8DE430                 STA   SAVECOLOR
3118 60                     RTS                ; exciting, wasn't it?
              ;
              ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              ;
              ; POSITION h,v
```

you to understand how simple assembly language can be.

Are there any tricky spots in the code? Not really. Though, if you are like me, you will have to pause each time you use a CMP and figure out if you really want BCS or BCC (or whether you also need a BEQ or...). Again, some of the CMP's could have been made simpler (for example, by using 'CMP #40' on line 1630 and omitting line 1640). And, again, I opted for consistency with the BASIC program.

The program does work. Try it. It took me about three hours to type it in and debug it (including about an hour of debugging the debugger). This represents much less time than it would have taken if I had not had the BASIC program as a working model. You might omit lines 1930 to 1980 the first time you run it. I won't tell you what will happen, but I will tell you that the lines are used to synchronize ball movement with the clock.

On Assembling And Debugging

You may have noted that the master origin ('*=') for this program is at $3000. If you use that origin and don't do anything special, assembling the program will wipe out the source code and *kablooey!* What can you do? Personally, I prefer to direct the object code to disk when I assemble. (I usually use 'ASM ,#R:,#D:file.OBJ' where "file" is the same name as the source file and I use "R:" because I list to a DIABLO or DEC serial printer.) Then, with the source also safely LISTed to disk, I can use NEW and reLOAD the object and proceed to run and debug it. Using this method, it makes sense to place the origin somewhere fairly high in EASMD's (or the Assembler/Editor's) working

```
            9455 ;
            9460 ;  ENTER: h (horizontal) position in X,A
            9465 ;               registers (LSB,MSB)
            9470 ;         v (vertical) position in Y-register
            9475 ;
            9480 ;   EXIT: unchanged
            9485 ;
            9490 POSITION
3119 8655   9495         STX  HORIZONTAL
311B 8556   9500         STA  HORIZONTAL+1 ; read the text
311D 8454   9505         STY  VERTICAL   ; too simple, right?
311F 60     9510         RTS
            9515 ;
            9520 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            9525 ;
            9530 ; PLOT h,v
            9535 ;
            9540 ;  ENTER: must have done a previous COLOR call
            9545 ;         X,A,and Y registers set as in POSITION
            9550 ;
            9555 ;   EXIT: Y-register has completion status
            9560 ;
            9565 PLOT
3120 201931 9570         JSR  POSITION
3123 A260   9575         LDX  #6*$10     ; file 6, again
3125 A90B   9580         LDA  #CPBINR    ; Command Put BINary Record
3127 9D4203 9585         STA  ICCOM,X
312A A900   9590         LDA  #0
312C 9D4803 9595         STA  ICBLEN,X
312E 9D4903 9600         STA  ICBLEN+1,X ; if buffer length is zero...
3132 ADE430 9605         LDA  SAVECOLOR ; then CPBINR puts one char from
                              A-reg
3135 2056E4 9610         JSR  CIO        ; and this is how we PLOT
3138 60     9615         RTS
            9620 ;
            9625 ;
            9630 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            9635 ;
            9640 ; DRAWTO h,v
            9645 ;
            9650 ;  ENTER: must have done a previous PLOT
            9655 ;         X,A,and Y registers as in POSITION
            9660 ;
            9665 ;   EXIT: Y-register has completion code
            9670 ;
            9675 DRAWTO
3139 201931 9680         JSR  POSITION
313C ADE430 9685         LDA  SAVECOLOR
313F 8DFB02 9690         STA  DRAWCOLOR ; where DRAWTO expects its color
3142 A260   9695         LDX  #6*$10     ; file 6...once more
3144 A911   9700         LDA  #CDRAW     ; Just a command to 'S:'
3146 9D4203 9705         STA  ICCOM,X
3149 A90C   9710         LDA  #$0C
314B 9D4A03 9715         STA  ICAUX1,X   ; insurance
314E A900   9720         LDA  #0
3150 9D4B03 9725         STA  ICAUX2,X   ; ...guaranteed to work
3153 2056E4 9730         JSR  CIO        ; do the actual DRAWTO
3156 60     9735         RTS
            9740 ;
            9745 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            9750 ;
            9755 ;
            9760 ; RND r
            9765 ;
            9770 ;  ENTER: r in A-register
            9775 ;   EXIT: A-register contains a random
            9780 ;         number from 0 to r, inclusive
            9785 ;
            9790 ;
            9795 RND
3157 8D6631 9800         STA  RTEMP
315A EE6631 9805         INC  RTEMP      ; makes CMP easier
            9810 RNDWAIT
315D AD0AD2 9815         LDA  RANDOM     ; get a random number
3160 CD6631 9820         CMP  RTEMP      ; too big?
3163 B0F8   9825         BCS  RNDWAIT    ; yes...wait
3165 60     9830         RTS
            9835 ;
3166 00     9840 RTEMP   .BYTE 0         ; RESERVE SPACE
            9845 ;
            9850 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            9855 ;
3167        9999         .END
```

memory.

An alternative method is to keep the object code in memory *below* all my source listing. With EASMD this is easy to do. For example, with this program, I simply used a 'LOMEM 3800' command to tell EASMD not to use any memory below \$3800. With the Assembler/Editor cartridge, it is almost as easy: simply use BUG to issue "C2E5 < 00,38" and then "G A000". (\$02E5 is system LOMEM, which the Assembler picks up and uses for its own when it is coldstarted at \$A000.) In both instances, make sure you have LISTed off any program in memory before changing the LOMEM bound, since it is the occurrence of NEW which forces the change.

Actually, I often use *both* of the above measures. And even then I can run into problems. When I was working on this month's program, for example, I could assemble and then load the program fine. But when I went to use "G3000" from BUG, the system looped madly. I'm still trying to figure out why, but I solved it by loading the OBJect file from the operating system and then reentering the Assembler via a cold start. BUG then worked fine. I hope that by next month I will have figured out the reason for this strange behavior and will report a fix to you. (To be fair, I am using a *very* early pre-release version of the cartridge...perhaps you won't have this problem.)

Breakpoint Setting

Possibly the biggest fault of BUG (both versions) is the lack of easy breakpoint capabilities. Changing instructions to BRKs (\$00) and back often gets so tiresome that I tend to say the heck with it and try out an otherwise unchecked portion of code. When I'm lucky, it all works. When I'm not, I turn off the power and start again. Thank goodness I'm not trying to do this with just a cassette. The corollary? If you are using a cassette-only system, proceed with utmost caution and take the trouble to set lots of breakpoints.

That's about it for this month. Next month we will add several complications to the bouncing ball program. We will also explore some news, trivia, and gossip. And, whatever you do, don't believe everything that people say about the Atari and Atari BASIC: we may have some surprising benchmarks for you. ©

For PET/CBM and VIC, this handy utility should solve some memory space problems, especially when instructions can be safely deleted from a program after they're no longer needed. It cannot work, however, on Original ROM PETs.

Electric Eraser

Louis F Sander
Pittsburgh

If you program in BASIC, you'd sometimes like to delete certain program lines after they've been executed, either to protect your program from piracy, or to free up memory for the rest of the program to use. The lines that print your on-screen instructions, for example, are good candidates for deletion as soon as they've been run. Having served their purpose, they do nothing but take up space, which can really be at a premium in small-memory machines like the VIC and the 8K PET.

It would be a real help if there were an easy way to delete such lines under program control. Well, now there *is* one: Electric Eraser is a two-line routine that deletes itself and all subsequent lines as soon as it's called.

Lines 210 and 220 in the accompanying program are the Electric Eraser for Upgrade and 4.0 ROM PET/CBM machines. If you have a VIC, your eraser appears in the REMarks following line 300. Move it up to lines 210-220 before you run the forthcoming demo. In all cases, line 300 activates the Eraser. There is nothing special about this choice of line numbers, and the three lines can be renumbered at will when you use them in other programs. They consume just over 100 bytes of memory.

To use the Eraser, you must set up the lines to be erased as the last lines in your program. There can be as many of them as you wish, and they should preferably include the activator line, since you'll have no need for it once the other lines have been erased. Put the Eraser immediately before the first line you want to erase. Then your program can execute any of its lines, except for the activator, to its heart's content.

There's no need to bypass the Eraser, since it has no meaningful effect until it's activated. When it's time for the Electric Eraser to do its work, execute the activator line. This will clear all variables and make the Eraser and everything after it disappear from the program. You can, if you like, replace the END in the Eraser with another statement, and it will be executed after it is deleted (!). If you leave out the END altogether, the subsequent lines may be executed, depending on what's in them, or your program may crash.

Watch It Work

Right now, let's see the Electric Eraser at work. Type in the demo program and SAVE it. Don't RUN it first to check your work, or you'll have to type it in again! LIST the program and carefully check lines 210, 220 and 300 for errors. Now RUN the program, and see for yourself that all its lines are actually executed, which should be obvious from the text that prints on the screen. RUN the program again, and you'll see that lines 210 and up do not execute this time, and that you now have several hundred more bytes of free memory. LIST the program to verify that lines 210-350 are no longer there. They have been electrically erased. You *could* say that these lines were executed, then they were executed. Or maybe they were just RUN to death. Anyway, they are gone without a trace, replaced by usable memory.

Eraser's Secret

Here is where they went: the first two PEEKs in line 210 are the keys to Electric Eraser's success. These locations contain a pointer to the start of the line currently being executed. When activated, the Eraser POKEs zeros into the link for that line and, using the USR vector as a temporary storage area, sets the Start of Variables pointer to the location just above that. As a result, BASIC thinks the program ends with the last line before the Eraser, which of course it now does. If all this is over your head, the System Information chapter of Osborne's *PET/CBM Personal Computer Guide* holds the keys to understanding. If you don't care about such matters, don't worry – you can use the Electric Eraser without understanding how and why it works.

Now you've seen the Electric Eraser in all its simple splendor, and maybe you've been impressed. If so, your next step is to add it to your bag of programming tricks, and to make equally impressive use of its powerful erasatorial punch. You could exercise your talents on the demo program, by replacing the END in line 220 with a RUN.

```
100 PRINT"{DOWN}FRE(0)=";FRE(0)
110 PRINT"{DOWN}WHERE IS THE REST OF THIS"
120 PRINT"LITTLE PROGRAM?"
200 REM ** 210 - 220 ARE THE ERASER
210 A=PEEK(58)+256*PEEK(59)+3:POKE2,INT(A/2
    56):POKE1,A-256*PEEK(2)
220 IFERTHENPOKEA-2,0:POKEA-1,0:POKE42,PEEK
    (1):POKE43,PEEK(2):CLR:END
230 PRINT"{DOWN}IF YOU LIST IT, YOU WON'T"
240 PRINT"FIND IT! IF YOU RUN IT ONCE"
250 PRINT"MORE, YOU'LL SEE THAT YOU"
```

```
260 PRINT"HAVE GAINED SOME MEMORY."
270 PRINT"{DOWN}THE ELECTRIC ERASER IS"
280 PRINT"POWERFUL MEDICINE!!"
300 ER=1:GOTO210:REM ** ACTIVATOR
310 REM
320 REM ** ERASER FOR THE VIC:
330 REM
340 A=PEEK(61)+256*PEEK(62)+3:POKE2,INT(A/2
    56):POKE1,A-256*PEEK(2)
350 IFERTHENPOKEA-2,0:POKEA-1,0:POKE45,PEEK
    (1):POKE46,PEEK(2):CLR:END
```

©

Put a digital clock on the Atari screen which ticks away, regardless of what's going on in BASIC. Type in this BASIC program and, after you run it, the clock will operate until you hit RESET. This clock has several worthwhile uses, not to mention the general applicability of this technique to other independent operations you might want to perform.

System Clock For The Atari

Bill Zimmerman
Littleton, CO

Run this BASIC program. Enter the correct time. A small digital clock will appear in the upper right-hand corner of the screen. The program ends. You are back to BASIC, but the little clock is still there counting the seconds. It stays there until you press RESET or disable the display. In the meantime, both you and your programs enjoy instant access to the correct time of day.

The key to this is the vertical blank routine of the Atari operating system. Sixty times every second, while the electron beam leaps to the top of the TV screen, your Atari steals a little time for itself. During this short interval the Atari restores numerous system values, maintains its real time clock and looks at the keyboard and game controllers.

A Polite Operating System

The Atari operating system even has manners! If a time-critical operation (like a disk read) is interrupted, the OS will add one to its clock and immediately return to the waiting operation. If a normal operation is interrupted, the entire vertical blank routine is executed. For a detailed description see page 99 of the *Atari Operating System User's Guide* (C016555).

The Atari documentation claims that page six (addresses 1536 to 1791) is used by the system only when the power is turned on. During testing, I discovered that page six is used by the BASIC LIST function. For this reason the clock routines are stored in high memory behind RAMTOP where they are safe. Safe? Did you know that shift CLEAR spreads 64 zeroes past RAMTOP? So the routines end up a full page (256 bytes) above RAMTOP. Safe? Not quite. If you list more than a few lines of your program at one time through a text window, the system may go into an endless internal loop. Do a GR.0 before scanning through your program.

At line 110, the program finds the current value of RAMTOP and sets PAGE to one less. At line 120, RAMTOP is reset to one page lower than PAGE and the address PAGEADDR is calculated. Lines 150-180 POKE the clock routines into memory, beginning at PAGEADDR. Since the assembler routines are compiled relative to page six, all sixes are changed to the new base – PAGE.

Lines 160 and 170 POKE six back into locations which really were sixes. Lines 200 through 280 accept the current time and perform some elementary editing checks. Lines 300 to 330 POKE the time values one digit at a time into the clock, and line 340 sets the clock in motion.

The OS vertical blank routine is reached by the computer through a special address called a vector. The system clock program changes the vertical blank vectors to point to its own code.

The clock and its control byte may be accessed by any BASIC program. The following routine will recalculate PAGEADDR:

```
10 RAMTOP=6*16+10
20 PAGE=PEEK(RAMTOP)+1
30 PAGEADDR=PAGE*256
```

The control byte is at PAGEADDR. To temporarily disable the display, POKE PAGEADDR,0. You might want to do this for games or when the clock would interfere with your screen. To redisplay the clock, POKE PAGEADDR,1.

The clock is stored in the six bytes following the control byte. Hours are stored in PAGEADDR+1 and PAGEADDR+2, minutes are stored in PAGEADDR+3 and PAGEADDR+4, and seconds are stored in PAGEADDR+5 and PAGEADDR+6. A program needing the current time could execute the following routine:

```
40 SAVETIME=0
50 CURRTIME=1000*PEEK(PAGEADDR+1)+100
   *PEEK(PAGEADDR+2)+10*PEEK(PAGEADDR
   +3)+PEEK(PAGEADDR+4)
60 IF CURRTIME<>SAVETIME THEN SAVETIME
   =CURRTIME:GOTO50
```

Type GR.2 for the big-screen effect, then GR.0 when you are ready to use your computer again.

Be careful when typing the DATA statements. A mistake will probably have dire consequences. In fact, it would be wise to save your work before RUNning the first time.

```
10 REM SYSTEM CLOCK FOR ATARI
20 REM WRITTEN BY BILL ZIMMERMAN
```

```
30 REM CLOCK CONTROL AT (RAMTOP)+1
40 REM 1 = DISPLAY
50 REM 0 = NO DISPLAY
60 REM CLOCK VALUE AT ((RAMTOP)+1)+1
70 REM SIX BYTES - HHMMSS
80 REM
100 DIM A$(3)
110 RAMTOP=6*16+10:PAGE=PEEK(RAMTOP)-1
120 POKE RAMTOP,PAGE-1:GRAPHICS 0:PAGEADDR=PAGE*256
130 ? :? :? ,"WINDING THE CLOCK":? :? :?
140 REM ARK POKE CLOCK INTO RESERVED MEMORY
150 FOR I=0 TO 237:READ X:IF X=6 THEN X=PAGE
160 POKE PAGEADDR+I,X:NEXT I
170 POKE PAGEADDR+9,6:REM REAL SIXES
180 POKE PAGEADDR+45,6
200 ? "WHAT TIME (HHMM)";:INPUT TIME
210 THH=INT(TIME/100):TMM=TIME-THH*100
220 IF THH>23 THEN 200
230 IF TMM>59 THEN 200
240 IF THH<>12 THEN 270
250 ? "MIDDAY";:INPUT A$:IF A$(1,1)<>"Y" THEN THH=0
260 GOTO 300
270 IF THH>12 THEN 300
280 ? "AM OR PM";:INPUT A$:IF A$(1,1)="P" THEN THH=THH+12
290 REM POKE IN TIME AND START CLOCK
300 X=INT(THH/10):POKE PAGEADDR+1,X
310 Y=INT(THH-X*10):POKE PAGEADDR+2,Y
320 X=INT(TMM/10):POKE PAGEADDR+3,X
330 Y=INT(TMM-X*10):POKE PAGEADDR+4,Y
340 X=USR(PAGEADDR+8)
350 END
1000 DATA 1, 0, 0, 0, 0, 0, 0, 196, 169, 6
1001 DATA 160, 28, 162, 6, 32, 92, 228, 169, 7, 160
1002 DATA 54, 162, 6, 32, 92, 228, 104, 96, 238, 7
1003 DATA 6, 240, 3, 76, 95, 228, 173, 0, 6, 73
1004 DATA 128, 141, 0, 6, 238, 6, 6, 169, 196, 141
1005 DATA 7, 6, 208, 235, 162, 4, 138, 208, 24, 173
1006 DATA 1, 6, 41, 2, 240, 17, 173, 2, 6, 41
1007 DATA 4, 240, 10, 169, 0, 141, 1, 6, 141, 2
1008 DATA 6, 240, 50, 169, 9, 221, 2, 6, 176, 45
1009 DATA 56, 189, 2, 6, 233, 10, 157, 2, 6, 254
1010 DATA 1, 6, 169, 197, 141, 7, 6, 169, 5, 221
1011 DATA 1, 6, 176, 105, 169, 0, 157, 1, 6, 202
1012 DATA 48, 97, 202, 169, 196, 141, 7, 6, 254, 2
1013 DATA 6, 16, 179, 240, 84, 173, 0, 6, 74, 144
1014 DATA 78, 10, 141, 0, 6, 165, 204, 72, 165, 205
1015 DATA 72, 24, 173, 48, 2, 105, 4, 133, 204, 173
1016 DATA 49, 2, 105, 0, 133, 205, 160, 1, 177, 204
1017 DATA 72, 136, 177, 204, 105, 30, 133, 204, 104, 105
1018 DATA 0, 133, 205, 162, 4, 160, 8, 189, 2, 6
1019 DATA 32, 222, 6, 189, 1, 6, 32, 222, 6, 202
1020 DATA 48, 8, 169, 10, 32, 222, 6, 202, 144, 233
1021 DATA 104, 133, 205, 104, 133, 204, 238, 0, 6, 76
1022 DATA 98, 228, 9, 16, 13, 0, 6, 145, 204, 136
1023 DATA 96, 13, 0, 6, 145, 204, 136, 96
```

©

This is an explanation for PET and VIC owners who wonder what BASIC looks like to the computer (it's not exactly what's on the screen). Also, have you ever needed to send text to your printer from machine language? This article explores both of these topics.

Inner BASIC

Jim Butterfield
Toronto

Question: When I type in a line of BASIC, how is it stored in memory? I've looked at the contents of hexadecimal addresses 400 and up in my PET and can't recognize anything.

Question: How do I print on my printer from machine language?

The two questions are partly related.

When a BASIC line is typed with a line number (so that it goes into memory), it will be stored almost as typed. In the PET, it will go into the area from hex 0400 and up. In the VIC it depends on the system: a minimum 5K VIC uses the area from hex 1000 and up. Without explaining in detail, here are the parts of a BASIC line stored in memory:

First two bytes: address link to next line ... or, if zero, end of program.

Next two bytes: line number in binary

Remainder: BASIC text with tokens

End-of-line: zero byte

If you don't know about tokens, you might read Herman's "Tokens Aren't Just For Subways" in *COMPUTE!'s First Book of PET/CBM*. So: 10 PRINT"XXX" will become: 0C 04 (link to next line at hex 040C); 0A 00 (line number 10); 99 (PRINT token); 22 58 58 58 22 ("XXX"); 00 (end of line).

That's not machine language; it's just tokenized BASIC. If you'd like to see where the interpreter does its machine language work, look up PRINT in a memory map; you can then disassemble and try to make sense out of it.

To PRINT in machine language, LOAD the A register with the ASCII character and call (JSR) hex FFD2. The character will print to the "standard" output – the screen.

To PRINT to a device other than the screen, the file must be OPENed first; this is most easily done from BASIC. When the machine language program is ready to PRINT, select the device with LDX (logical file number)/JSR $FFC9 – this sequence is equivalent to CMD (logical file number). Now PRINT as above. When you have finished for the moment, disconnect the device with JSR $FFCC. Eventually, you should CLOSE the file. Again, this is most easily done in BASIC.

If you have a PET, try entering the following information in hex:

```
0400 00 13 04 0A 00 9F 31 2C
0408 34 3A 9E 31 30 34 35 3A
0410 A0 31 00 00 00 A2 01 20
0418 C9 FF A9 41 20 D2 FF 18
0420 69 01 C9 5B D0 F6 A9 0D
0428 20 D2 FF 20 CC FF 60 00
```

This will also work on a VIC with a 3K expansion module. However, if you have the minimum 5K VIC, try entering:

```
1000 00 13 10 0A 00 9F 31 2C
1008 34 3A 9E 34 31 31 37 3A
1010 A0 31 00 00 00 A2 01 20
1018 C9 FF A9 41 20 D2 FF 18
1020 69 01 C9 5B D0 F6 A9 0D
1028 20 D2 FF 20 CC FF 60 00
```

You'll need a monitor for the VIC to do this, of course.

We have entered a program that is both BASIC and machine language. BASIC is contained in the first two and a half lines; the rest is machine language. Check it carefully. You can go back to BASIC and LIST the BASIC part. To see the ML part you'll need a disassembler.

The program as given should RUN, but to wrap things up neatly we should do one more thing: set the Start-of-Variable pointer. It's good practice and will make our program SAVE-able. On the PET, we should put address 042F into this pointer (located at hex 7C and 7D on Original ROM PETs; hex 2A and 2B on newer machines). On the VIC, we should put address 102F into the pointer at hex 2D and 2E. Don't forget that addresses go in backwards, or low order first, so that in the case of newer PETs, value 2F would go into address 2A and value 04 into address 2B.

Whether the pointers are fixed up neatly or not, you may go back to BASIC and say RUN. The program, all 47 bytes of it, causes the alphabet to be output to the printer.

Sometimes a working example is worth many pages of explanation. Try this one. Dissect it. See if you can see how it works.

We've written a BASIC program in hexadecimal, manufacturing line number, tokens, and all. Then we wrote a linked machine language program, and made it all work together.

Now see if you can output the numeric digits as well as the alphabet. ©

Making backup copies of a disk can present problems when you are using a single-drive device. Owners of the Commodore 2031 disk drive should find this file copying program indispensable.

Programs 1 and 2 are BASIC loaders. You type in the version for your PET (either Upgrade or 4.0) and it will create the machine language for you. Then, to start the program, type SYS 634.

Copy 2031 Files

G H Watson
University of Delaware
Newark, DE

Mass data storage and retrieval has been made convenient, fast, and reliable for the microcomputer user with access to a floppy disk drive. With the introduction of the CBM 2031 Single Disk Drive, Commodore has allowed the benefits of disk storage to be available to PET/CBM VIC owners with even the smallest computing budgets. However, while a single drive is more affordable than a dual drive, certain handicaps soon become apparent.

A major problem is the inability to quickly produce backup copies of disk files on a different diskette (handled easily on a dual drive with a single command). With the program here, Copy 2031 Files, the contents of a disk file are transferred to PET's programmable memory and then transferred back to a different diskette, all at machine language speed. The user simply enters the filename and switches diskettes at the appropriate time.

Operation of the program may be understood through comparison with its BASIC counterpart. In *OPEN ERROR CHANNEL* a channel is prepared for input of disk error messages.

```
100 OPEN 1,8,15
```

The name of the file to be copied is entered in * OPEN FILE FOR READ*. The filename is then appended with ",P,R" (or ",S,R") and the file is opened for reading.

```
110 PRINT:INPUT "FILENAME";FL$
120 OPEN 2,8,2,FL$+",P,R"
130 GOSUB 500
```

READ FILE loads the file into the memory of the PET (normally occupied by a BASIC program). The end of the file is detected via a change in the status word ST.

```
140 XFR=TP
150 GET#2,C$
160 IF C$="" THEN C=0:GOTO 180
170 C=ASC(C$)
180 POKE XFR,C
190 IF ST THEN 210
200 XFR=XFR+1:GOTO 150
210 EOF=XFR
220 CLOSE 2
```

At this point the diskettes are switched and a file is opened for writing in *OPEN FILE FOR WRITE*

```
230 PRINT:PRINT"SWITCH DISKETTES,"
240 PRINT"THEN HIT RETURN."
250 GETC$:IF C$="" THEN 250
260 OPEN 2,8,2,FL$+",P,W"
270 GOSUB 500
```

The reverse process is carried out in *WRITE FILE*. The file contents are transferred byte by byte until the end of the file is indicated.

```
280 XFR=TP
290 C=PEEK(XFR)
300 C$=CHR$(C)
310 PRINT#2,C$;
320 IF XFR<EOF THEN XFR=XFR+1:GOTO 290
```

When the file is completely transferred, all files are closed in *EXIT*.

```
330 CLOSE 2:CLOSE 1:END
```

The subroutine *DERROR* allows disk errors to be detected and displayed.

```
500 INPUT#1,EN$
510 IF EN$="00" THEN RETURN
520 PRINT:PRINT"DISK ERROR ";EN$;"!"
```

For the BASIC equivalent to work correctly a safe storage space must be allocated in memory for the file.

```
10 POKE 53,8:CLR
20 TP=PEEK(53)*256+PEEK(52)
```

Copy 2031 Files has been assembled to reside in the first and second cassette buffers of a BASIC 4.0 PET. [*The BASIC loaders provided (Programs 1 and 2) are for 4.0 and Upgrade BASIC.*] The program might run on a VIC-20 if the system variables and subroutine calls can be supplied by a knowledgeable VIC owner. Incidentally, the program will also work with the CBM 4040 Dual Disk Drive.

ECLECTIC SYSTEMS CORPORATION
Order TOLL FREE 1-800-527-3135

P.O. Box 1166 • 16260 Midway Road
Addison, TX 75001 • (214) 661-1370

ANNOUNCES

EM© for the SuperPET

commodore

EM© is ANSI standard MUMPS

Make your SuperPET Super-Load in EM© and Dramatically Improve your Programming Productivity

EM© is more than just a programming language. It is a well integrated data management system combining with one syntax what other operating systems would call 1) an application programming language; 2) a job control language; 3) a linkeage editor; 4) a database management system; and 5) a communications monitor.

PROGRAM MANAGEMENT:
EM© provides all programming management facilities needed to manage programs and program files. Programs can be created, edited, cataloged and debugged from within EM©. Programs can be as large as disk capacity. A resident algorithm rids memory of least frequently used variables and program modules so that what you need off-disk normally resides in memory.

STRING POWER:
EM© makes string handling easy with its extensive set of string operations and functions. Variable length strings can be used routinely without the obstacles presented by most other programming languages.

PATTERN MATCHING:
EM© can "filter" user input with a useful pattern matching that will result in fewer user or device errors. For example: dates, zip codes and names can be tested for validity with a single statement.

GLOBALS:
EM© obviates the need for traditional read and write operations on secondary storage devices by allowing data elements to be directly referenced as a set of subscripts; all the details of file organization and retrieval are handled by the system.

TIMING:
EM© enables a programmer to associate timing constraints with several operations. This feature allows testing for terminal malfunctions as well as prompting users in time-critical dialogue.

DATA BASE MANAGEMENT:
Sorts and merges are not necessary as EM© automatically stores data in a dynamically allocated balanced tree structure. Random access to any data item requires at most three disk reads.

EM© UNMATCHED IN PROGRAMMING PRODUCTIVITY:
System houses that program in EM© (MUMPS) find that their costs are lower than those of their competitors using other languages. Fewer lines of code are necessary per application. Dimension statements are not required. Subscripts may be alpha, numeric or any legal string. Data types need not be defined and can change freely throughout as EM© can recognize when it is dealing with alpha, numeric, integer or floating-point data types. EM© gives the professional programmer a full set of software tools designed for real-life tasks and problems he consistently encounters in the production and maintenance of application software. EM© adheres rigidly to ANSI MUMPS standards, which make it transportable to larger processors manufactured by DEC, TANDOM, DATA GENERAL, HARRIS and others. Additionally EM© gives the less-experienced programmer the tools to do a professional job on formidable programming applications.

You may order EM© or SuperPET by calling ECLECTIC SYSTEMS toll-free at 1-800-527-3135 from 10 AM to 4 PM CDT Monday through Friday, or you can order by mail using the form below. Texas residents call 1-214-661-1370.

ECLECTIC SYSTEMS CORPORATION
P.O. Box 1166, 16260 Midway Road, Addison, Texas 75001

Here's my order for EM©@ $299 plus $3.75 for shipping and handling (UPS surface unless specified otherwise). Residents of Texas, Louisiana, Oklahoma City and Tulsa, Oklahoma must add applicable taxes.

☐ My certified check or money order is enclosed.
☐ Please charge my VISA #____________________ or
MasterCard #____________________ Expiration date ________

Name ____________________ Signature ____________________

Address ____________________

City __________ State __________ Zip __________

The program resides happily in the cassette buffers unless 1) the cassette drive is accessed, or 2) the advanced DISK BASIC commands in BASIC 4.0 are used. For example, entering DIRECTORY D0 would cause part of the program in the second cassette buffer to be overwritten. If this creates a problem, assemble the program elsewhere. Using DOS Wedge commands will not harm the program though.

As shown, Copy 2031 Files will copy program files (BASIC programs, *WordPro* files, MAE files, ...). This is controlled by the appendix ",P,R" stored in *STRING TABLE* in reverse order. In order to copy sequential files (Data files, *PaperMate* files, ASM/TED files, ...) the *P* in the appendix should be replaced with an *S*. This change may be accomplished before running the program (SYS 634) by changing the byte with a POKE (POKE 952,80 for program files and POKE 952,83 for sequential files). For copying a large number of files, you may consider changing JMP READY to JMP BEGIN.

Program 1. 4.0 Version

```
500 FOR ADRES=634TO954:READ DATTA:POKE
    ADRES,DATTA:NEXT ADRES
634 DATA 169, 1, 133, 210, 32, 226
640 DATA 242, 169, 8, 133, 212, 169
646 DATA 15, 133, 211, 169, 0, 133
652 DATA 209, 32, 99, 245, 160, 3
658 DATA 169, 115, 32, 29, 187, 32
664 DATA 226, 180, 169, 0, 133, 218
670 DATA 169, 2, 133, 219, 160, 255
676 DATA 200, 177, 218, 208, 251, 162
682 DATA 4, 189, 181, 3, 145, 218
688 DATA 200, 202, 208, 247, 132, 209
694 DATA 169, 2, 133, 210, 32, 226
700 DATA 242, 169, 8, 133, 212, 169
706 DATA 2, 133, 211, 32, 99, 245
712 DATA 32, 77, 3, 162, 2, 32
718 DATA 198, 255, 169, 4, 133, 1
724 DATA 169, 3, 133, 0, 160, 0
730 DATA 32, 21, 242, 145, 0, 166
736 DATA 150, 208, 7, 200, 208, 244
742 DATA 230, 1, 208, 240, 132, 5
748 DATA 165, 1, 133, 6, 169, 2
754 DATA 32, 226, 242, 32, 204, 255
760 DATA 160, 3, 169, 127, 32, 29
766 DATA 187, 32, 228, 255, 240, 251
772 DATA 164, 209, 136, 169, 87, 145
778 DATA 218, 32, 99, 245, 32, 77
784 DATA 3, 162, 2, 32, 201, 255
790 DATA 169, 4, 133, 1, 169, 3
796 DATA 133, 0, 160, 0, 177, 0
802 DATA 32, 102, 242, 165, 1, 197
808 DATA 6, 208, 4, 196, 5, 240
814 DATA 14, 200, 208, 238, 230, 1
820 DATA 208, 234, 160, 3, 169, 164
826 DATA 32, 29, 187, 169, 2, 32
832 DATA 226, 242, 169, 1, 32, 226
838 DATA 242, 32, 204, 255, 76, 255
844 DATA 179, 162, 1, 32, 198, 255
850 DATA 32, 21, 242, 141, 176, 3
856 DATA 32, 21, 242, 141, 177, 3
862 DATA 32, 204, 255, 173, 176, 3
868 DATA 201, 48, 208, 206, 173, 177
874 DATA 3, 201, 48, 208, 199, 32
880 DATA 204, 255, 96, 13, 70, 73
886 DATA 76, 69, 78, 65, 77, 69
892 DATA 63, 32, 0, 13, 83, 87
898 DATA 73, 84, 67, 72, 32, 68
904 DATA 73, 83, 75, 69, 84, 84
910 DATA 69, 83, 44, 13, 84, 72
916 DATA 69, 78, 32, 72, 73, 84
922 DATA 32, 82, 69, 84, 85, 82
928 DATA 78, 46, 13, 0, 13, 68
934 DATA 73, 83, 75, 32, 69, 82
940 DATA 82, 79, 82, 32, 0, 0
946 DATA 33, 13, 0, 0, 82, 44
952 DATA 80, 44, 246, 230, 1, 76
```

Program 2. Upgrade ROM Version

Change these lines in Program 2.

```
634 DATA 169, 1, 133, 210, 32, 174
652 DATA 209, 32, 36, 245, 160, 3
658 DATA 169, 115, 32, 28, 202, 32
664 DATA 111, 196, 169, 0, 133, 218
694 DATA 169, 2, 133, 210, 32, 174
706 DATA 2, 133, 211, 32, 36, 245
730 DATA 32, 225, 241, 145, 0, 166
754 DATA 32, 174, 242, 32, 204, 255
760 DATA 160, 3, 169, 127, 32, 28
766 DATA 202, 32, 228, 255, 240, 251
778 DATA 218, 32, 36, 245, 32, 77
802 DATA 32, 50, 242, 165, 1, 197
826 DATA 32, 28, 202, 169, 2, 32
832 DATA 174, 242, 169, 1, 32, 174
838 DATA 242, 32, 204, 255, 76, 137
844 DATA 195, 162, 1, 32, 198, 255
850 DATA 32, 225, 241, 141, 176, 3
856 DATA 32, 225, 241, 141, 177, 3
```

©

BATTERIES INCLUDED

village by the grange, 71 mccaul st. (f6) toronto m5t 2x1 telephone 596-1405

ARBITER 1.4 MULTI-USER DISK SYSTEM FOR COMMODORE 4.0 COMPUTERS

OVER THREE HUNDRED IN USE ACROSS ONTARIO

Since September 1981 **BATTERIES INCLUDED** has been installing the ARBITER system in classrooms of Commodore BASIC 4.0 computers. The computers are connected to CBM Disk Drives and printers. All users have access to all disk drives and printers plus a host of commands to make this system configuration really usable!

THE ARBITER 1.4 SYSTEM IS READY TO GO!

FEATURES

1) Easy installation.
2) Uses no RAM or Utility Sockets.
3) Up to 32 computers in one system.
4) System self initializes on power up.
5) Operation is completely transparent to the user.
6) Extended commands allow a friendly multi-user environment.
7) System design virtually eliminates interleaved printer output.

$150.00

SPECIAL COMMANDS

@S – Allows students to protect files with a five character password. A three character user ID is forced into the file name.

@L – Allows the students to load protected files if the password code is known.

LISTC – Used to produce program listings with a Commodore printer. Clumsy OPEN, CMD, LIST, PRINT#, CLOSE sequence not needed. It overcomes the listing problems found on other multi-user hardware systems.

LISTP – Used to get program listings on systems which have an ASCII printer. The cursor control characters are expanded and displayed in brackets. e.g. <home>

ALL FILE TYPES ARE SUPPORTED – During relative or sequential file access a delay has been built in so the computer will retain control of the system until the file is closed.

TEACHER UTILITY – A utility is supplied on disk to allow the teacher to produce a hardcopy listing and output from any of the protected or unprotected files selected. Once the files are chosen from the disk directory the teacher may do other tasks while the job is completed.

IF YOUR CLASSROOM WAS DESIGNED TO TEACH COMPUTER LITERACY OR STRUCTURED BASIC THEN THIS SYSTEM WAS DESIGNED FOR YOU.

Arbiter and Arbiter 1.4 are copyrights of Batteries Included.

SWARM-100

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.

For 24 Pin ROM Machines Only.

BATTERIES INCLUDED

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

Hit one key and a whole BASIC word is printed for you on the screen. This programming shorthand is adapted to the VIC from a PET program, "Keyword." There is also an example of how to go about converting PET machine language to work on the VIC.

VIC-Key

Thomas Henry
Mankato, MN

VIC-Key is a utility for the Commodore VIC-20 computer written in machine language. Like Charles Brannon's Keyword (**COMPUTE!**, August, 1981, #15), it lets one keystroke do a lot of work. For example, hit SHIFT-A and the statement "ASC" shoots out. SHIFT-B gives "STEP", SHIFT-C gives "CHR$" and so on. In short, 26 of the most common BASIC statements now have one-keystroke equivalents. And, unlike the standard two keystroke abbreviations that Commodore provides (for example, "I,SHIFT-N" is equivalent to "INPUT"), this version spells out the entire phrase instantly. Now when you hit SHIFT-I the entire word "INPUT" dashes out on the screen. As you can tell, this is a real time saver.

An Important Addition

Although VIC-Key is based on the article mentioned above, one important addition has been made to the program. Using capital letters for the various keywords is a great idea since the VIC-20 doesn't like to see shifted letters in a BASIC statement anyway. However, there is one time when you want a capital letter to really be a capital letter (not a keyword), and that's when you're inside quotes. For example, line 10 of a program may read:

```
10 PRINT "I AM YOUR QUIZ-MASTER. HIT RETURN."
```

You clearly want the capital "I" and the capital "H" to be just that, ordinary capitals. Well, VIC-Key has been written in such a way that it keeps track of whether you're inside quotes or outside quotes and adjusts accordingly.

VIC-Key is able to decide if you're in quotes or not by inspecting location $D4 in the zero page. If this location contains a zero then the quotes are OFF, and it's safe to perform the key-statement transformation. If this location contains a one, then the quotes are ON and the transformation must be skipped. (If you are a PET user, you may want to modify the original Keyword program mentioned above to also keep track of quotes. The quotes flag location for the PET is $CD).

Modifying PET Machine Language To Run On VIC

Changing Brannon's Keyword program into VIC-Key was mostly a matter of disassembling the original, finding all the zero page locations called out, finding their equivalents in the VIC-20 memory map and changing them accordingly. However, there was one tricky point that almost made me give it up as hopeless. Since the VIC-20 is a relatively new computer, very little has been published on its BASIC in ROM. In short, I couldn't figure out where the needed Table of BASIC Keywords was located. All I knew was that it was somewhere between $C000 and $FFFF!

After just about giving up, I hit upon the idea of inspecting the VIC's ROMs with my CBM 8032. First I transferred the VIC's ROMs to DATA statements 500 bytes at a time using H. Linder's Automatic Data Statement program (**COMPUTE!**, October, 1981, #17) (modified for use with the VIC-20). After doing this I loaded the tape just made into my CBM 8032. I did this with the help of L. Jordan's "Train Your PET to Run VIC Programs" (**COMPUTE!**, October, 1981, #17). In effect, I recreated the VIC ROMs in my CBM 8032's RAM. I then disassembled this "pseudo-ROM" using Cochrane's Micromon (**COMPUTE!**, January, 1982, #20), an extended monitor, and eventually found the table I needed. To save yourself this work, you may want to make a note that the start of the Table of BASIC Keywords is $C09E.

To use the program, follow these steps:

1) Enter the program.

2) After inspecting it for accuracy, SAVE it to tape.

3) RUN it, then SYS7501. VIC-Key is now activated.

4) Give it a try. The table shows the keyword equivalents. Confirm that VIC-Key knows whether you're in quote mode or not.

5) If you want to deactivate the program, simply SYS7501 again. VIC-Key is now dormant, but not wiped out from memory. You can reactivate it again at any time by doing another SYS7501.

6) Since the top of memory pointers has been lowered, VIC-Key is safe from BASIC program interference. In addition, typing NEW will not affect it. However, hitting the STOP/RESTORE key combination will wipe it out completely.

The keyword equivalents in the table are very easy to memorize if you note the following:

1) Most commands are simply alphabetical.

For example, SHIFT-A equals "ASC", SHIFT-C equals "CHR$", etc.

2) SHIFT-W, X and Y are DATA type commands, i.e., "DATA", "READ", "RESTORE".

3) For SHIFT-H think "halt" (equals STOP).

4) SHIFT-P is POKE and, one letter later, (SHIFT-Q) is PEEK.

VIC-Key consumes 174 bytes of memory, which leaves plenty left over for BASIC programming even with the limited memory of a stock VIC-20. With the new quote mode detector, VIC-Key is so easy to use that I think you'll agree that it will more than "pay" for the little memory that it uses. So rest those tired hands; let VIC-Key do the typing.

Table of BASIC Keywords

A	ASC	O	OPEN
B	STEP	P	POKE
C	CHR$	Q	PEEK
D	DIM	R	RIGHT$
E	END	S	STR$
F	GET	T	TAB(
H	STOP	U	USR
I	INPUT	V	VAL
J	GOTO	W	DATA
K	GOSUB	X	READ
L	LEFT$	Y	RESTORE
M	MID$	Z	SYS
N	NEXT		

Program.

```
100 POKE55,77:POKE56,29
110 PRINT"WAIT..."
120 FORI=7501TO7679
130 READA:POKEI,A:X=X+A
140 NEXT
150 PRINT"SYS7501 TO ACTIVATE.";
160 IF X <> 22351 THEN PRINT" THERE IS AN E
    RROR IN YOUR TYPING OF THE DATA LI
    NES"
170 NEW
180 DATA120,173,20,3,72,173,21,3,72,173,116
    ,29,208,2,169,118
190 DATA141,20,3,173,117,29,208,2,169,29,14
    1,21,3,104,141,117
200 DATA29,104,141,116,29,88,96,0,0,72,138,
    72,152,72,165,215
210 DATA72,165,212,240,4,104,76,221,29,104,
    201,193,144,82,201,219
220 DATA176,78,56,233,193,170,189,229,29,16
    2,0,134,198,170,160,158
230 DATA132,34,160,192,132,35,160,0,10,240,
    16,202,16,12,230,34
240 DATA208,2,230,35,177,34,16,246,48,241,2
    00,177,34,48,17,8
250 DATA142,255,29,230,198,166,198,157,119,
    2,174,255,29,40,208,234
260 DATA230,198,166,198,41,127,157,119,2,16
    9,20,141,119,2,230,198
270 DATA104,168,104,170,104,76,191,234,198,
    169,199,134,128,129,161,144
280 DATA133,137,141,200,202,130,159,151,194
    ,201,196,163,183,197,131,135
290 DATA140,158,127
```

References

1) C. Brannon, "Keyword," **COMPUTE!** *#15, August 1981, pp. 120, 122.*

2) H. Linder, "Automatic DATA Statements for CBM and Atari," **COMPUTE!** *#17, October 1981, p. 22.*

3) L. Jordan, "Train Your PET to Run VIC Programs," **COMPUTE!** *#17, October 1981, p. 138.*

4) R. A. Cochrane, "MICROMON: An Enhanced Machine Language Monitor," **COMPUTE!** *#20, January 1982, pp. 160-173.*

©

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

In order to make special characters, inverse video, and cursor characters easy to type in, **COMPUTE!** magazine's Atari listing conventions are used in all the program listings in this magazine.

Please refer to the following tables and explanations if you come across an unusual symbol in a program listing.

Atari Conventions

Characters in inverse video will appear like: INVERSE VIDEO
Enter these characters with the Atari logo key, {⅄}.

When you see	Type	See	
{CLEAR}	ESC SHIFT <	↖	Clear Screen
{UP}	ESC CTRL -	↑	Cursor Up
{DOWN}	ESC CTRL =	↓	Cursor Down
{LEFT}	ESC CTRL +	←	Cursor Left
{RIGHT}	ESC CTRL *	→	Cursor Right
{BACK S}	ESC DELETE	◀	Backspace
{DELETE}	ESC CTRL DELETE		Delete character
{INSERT}	ESC CTRL INSERT		Insert character
{DEL LINE}	ESC SHIFT DELETE		Delete line
{INS LINE}	ESC SHIFT INSERT		Insert line
{TAB}	ESC TAB	▶	TAB key
{CLR TAB}	ESC CTRL TAB		Clear tab
{SET TAB}	ESC SHIFT TAB		Set tab stop
{BELL}	ESC CTRL 2		Ring buzzer
{ESC}	ESC ESC		ESCape key

Graphics characters, such as CTRL-T, the ball character ● will appear as the "normal" letter enclosed in braces, e.g. {T}.

A series of identical control characters, such as 10 spaces, three cursor-lefts, or 20 CTRL-R's, will appear as {10 SPACES}, {3 LEFT}, {20 R}, etc. If the character in braces is in inverse video, that character or characters should be entered with the Atari logo key. For example, {♥} means to enter a reverse-field heart with CTRL-comma, {5 U} means to enter five inverse-video CTRL-U's.

For PET/CBM/VIC

Generally, any PET/CBM/VIC program listings will contain bracketed words which spell out any special characters: {DOWN} would mean to press the cursor-down key; {3DOWN} would mean to press the cursor-down key three times.

To indicate that a key should be *shifted* (hold down the SHIFT key while pressing the other key), the key would be underlined in our listing. For example, S would mean to type the S key while holding the shift key. This would result in the "heart" graphics symbol appearing on your screen.

Sometimes in a program listing, especially within quoted text when a line runs over into the next line, it is difficult to tell where the first line ends. How many times should you type the SPACE bar? In our convention, when a line breaks in this way, the ~ symbol shows exactly where it broke. For example:

```
100 PRINT "TO START THE GAME  ~
    YOU MAY HIT ANY OF THE KEYS
    ON YOUR KEYBOARD."
```

shows that the program's author intended for you to type two spaces after the word *GAME*.

For The Apple

Programs listed as "Microsoft" are written for the PET/CBM, Apple, OSI, etc. Although the programs are general in nature, you may need to make a few changes for them to run correctly on your Apple. Microsoft BASIC programs written for the PET/CBM sometimes contain special cursor control characters. The following table shows equivalent Apple words. Notice that these Apple commands are *outside* quotations (and even separate from a PRINT statement). PRINT"[RVS]YOU WON" becomes INVERSE: PRINT"YOU WON":NORMAL

[CLEAR[(Clear Screen) HOME

[DOWN] (Cursor down)
Apple II +: Call -922
POKE 37,PEEK(37)+(PEEK(37)<23)

[UP] (Cursor up)
POKE 37,PEEK(37)-(PEEK(37)>0))

[LEFT] (Cursor left) PRINT CHR$(8);

[RIGHT] (Cursor right)
PRINT CHR$(21)

[RVS] (Inverse video on. Turns off automatically after a carriage return. To be safe, turn off inverse video after the print statement with NORMAL unless the PRINT statement ends with a semicolon.)
INVERSE

[OFF] (Inverse video off) NORMAL

Shifted characters can represent either graphics characters or uppercase letters. If within text, just use the non-shifted character, otherwise substitute a space. Some "generalized" programs contain a POKE such as POKE 59468,14. Omit these from the program when typing it in. One final note: you will probably want to insert a question mark or colon within an INPUT prompt. PET/CBM and many other BASICs automatically print a question mark:

INPUT "WHAT IS YOUR NAME";N$
becomes
INPUT "WHAT IS YOUR NAME?";N$

All Commodore Machines

Clear Screen	{CLEAR}	**Cursor Left**	{LEFT}
Home Cursor	{HOME}	**Insert Character**	{INST}
Cursor Up	{UP}	**Delete Character**	{DEL}
Cursor Down	{DOWN}	**Reverse Field On**	{RVS}
Cursor Right	{RIGHT}	**Reverse Field Off**	{OFF}

VIC Conventions

Set Color To Black	{BLK}	**Function Two**	{F2}
Set Color To White	{WHT}	**Function Three**	{F3}
Set Color To Red	{RED}	**Function Four**	{F4}
Set Color To Cyan	{CYN}	**Function Five**	{F5}
Set Color To Purple	{PUR}	**Function Six**	{F6}
Set Color To Green	{GRN}	**Function Seven**	{F7}
Set Color To Blue	{BLU}	**Function Eight**	{F8}
Set Color To Yellow	{YEL}	**Any Non-implemented Function**	{NIM}
Function One	{F1}		

8032/Fat 40 Conventions

Set Window Top	{SET TOP}	**Erase To Beginning**	{ERASE BEG}
Set Window Bottom	{SET BOT}	**Erase To End**	{ERASE END}
Scroll Up	{SCR UP}	**Toggle Tab**	{TGL TAB}
Scroll Down	{SCR DOWN}	**Tab**	{TAB}
Insert Line	{INST LINE}	**Escape Key**	{ESC}
Delete Line	{DEL LINE}		

A Monthly Feature

You often need to know on which screen you defined a particular word. If your system supports, say, 300 screens, it's tiresome to index through them, looking for something. This search routine combines machine language with FORTH and is a fast, efficient way to find "lost" definitions.

If you have come up with some interesting FORTH applications or techniques, send them in to The FORTH Page, **COMPUTE!** *Magazine, P.O. Box 5406, Greensboro, NC, 27403 and share them with the rest of us.*

The FORTH Page

Speed Search

Richard Mansfield, Assistant Editor

These three screens compile the word HUNT, which will locate anything on disk. Assume that you are writing a game and you remember that somewhere on your disk you defined RND to provide a random number. Unfortunately, you cannot now recall exactly where RND is located, but you think it might be between screens 50 and 70. All too often, you must laboriously list each screen and read through it, looking for that "missing" definition.

This fast search routine will fly effortlessly through your disk, reporting the screen and line number where it finds matches. To find RND, you first introduce the target by typing " RND" and then type:

50 70 HUNT

and each screen number is printed as it is checked. Any line containing a match is printed out beneath the screen number. To hunt only for the actual definition of the word, use the colon as well:

" : RND"

FORTH Compatibility

Ideally, FORTH would be system independent: it wouldn't matter what computer you are using, you could type in a screen from **COMPUTE!** and it would work on your machine as printed. In practice, however, there always seem to be a few minor adjustments to make to a FORTH program of any significant length before it will work for your particular setup.

This search routine was developed on "FORTH For PET" which includes a word, ?TERMINAL, which checks to see if the PET STOP key is pressed. The user then can exit a loop from the keyboard as illustrated in line 13 of Screen 112. HUNT contains the modifications necessary to make it work on the APX figFORTH for the Atari. ?TERMINAL is not available on the APX version of FORTH.

Line three, Screen 110 is an Atari specific definition for ?TERMINAL. It reads the console switches and returns a three-bit result between one and seven. Each bit (1, 2, 4) represents either the START, SELECT, or OPTION keys. Any combination of these keys could be tested by using AND, but here we are merely seeing if any are pressed and, if so, we LEAVE the HUNT.

A second, minor, variation between these FORTHs requires the substitution of IFEQ for 0= on line ten, Screen 111, within the machine language character comparison. There is a major difference, on the other hand, in the way that Atari handles BLOCK.

BLOCK Modifications For The Atari

On the PET, the word BLOCK (n1 — addr) returns the memory address of the start of a 1024-byte block. On the Atari, the word BLOCK returns the address of a 132-byte block and the value of n1 is a disk sector number (not a screen number). The Atari block is 128 bytes plus four additional bytes which are perhaps for sector management.

To simulate the PET method of handling BLOCK, line one of screen 110 defines the word BLOK. It multiplies the screen number by eight to get the correct sector and then reads in eight sectors. The address of the first sector is then left on the stack. The following sectors are in memory as required along with the four-byte tags. If you want to try to eliminate the four tag bytes, beware of damage to disk management caused by any subsequent FLUSHes.

The translation between PET and Atari FORTH is not perfect. Because of those tag bytes, a false match will be reported now and then in the Atari version. What's more, the original PET (80 column) version included a superior alternative to .LINE. When a match was found, HUNT listed the screen and flashed the target word on and off while ringing the bell. Calculating the exact video screen position of the target word is, of course, especially machine-specific, but it is impressive to watch. It requires the following modifications to MARKSTRING and the addition of the word WHITEIT:

```
: WHITEIT OVER OVER 0 DO DUP I + 80 TOGGLE LOOP
    7 EMIT DROP ;
: MARKSTRING ( SCR# ADDR --- SCR# )
```

```
     OVER DUP SCR @ = 0=
      IF DUP LIST CR ENDIF
     BLOCK - 40 /MOD 1+ 50 * 4 + + 8050 + PAD C@
       BEGIN WHITEIT WHITEIT GET UNTIL
     DROP DROP ;
```

HUNT

```
SCR # 110
 FORTH DEFINITIONS HEX      0 VARIABLE 1STCHAR
 : BLOK 8 * DUP BLOCK SWAP DUP 7 + SWAP DO I BLOCK DROP LOOP ;

 : ?TERMINAL -2FE1 C@ 7 XOR ;   ( READS ATARI CONSOLE SWITCHES)
 ( BLOK AND ?TERMINAL ARE FOR ATARI USERS ONLY )
 : MATCH ( ADDR1 ADDR2 N --- F )
   -DUP IF OVER + SWAP
           DO DUP C@ I C@ -
                IF 0= LEAVE ELSE 1+ THEN
           LOOP
       ELSE DROP 0= THEN ;

  : CHECKIT PAD 1+ PAD C@ MATCH ;    ( ADDR --- F )

  : HEADER CR ." SEARCHING FOR " 22 EMIT SPACE PAD
     1+  PAD C@ TYPE 22 EMIT  SPACE  ." ON SCR # .. ." ; -->


SCR # 111

  : MARKSTRING  ( SCR# ADDR --- SCR# )
     OVER BLOCK - C/L / CR DUP . SPACE  ( ATARI, USE BLOK)
    OVER .LINE CR ;




  CODE ?CHAR  ( ADDR --- ADDR F )
     1 # LDA,  SETUP JSR,
     N )Y LDA,  1STCHAR CMP,  0=  ( ATARI, USE IFEQ, NOT 0=)
       IF,  1 # LDA, PUSH0A JMP,  THEN,
       0 # LDA,  PUSH0A JMP,             FORTH
  -->


SCR # 112
  : ONEBLK          ( SCR# ADDR --- )
     DUP 400 + SWAP  ( ATARI, USE 410, NOT 400)
       DO I ?CHAR
          IF I CHECKIT
              IF I MARKSTRING ENDIF
          ENDIF
       LOOP DROP ;

 : "  22 WORD HERE DUP C@ 1+ PAD SWAP CMOVE ;

 : HUNT    (  SCR#1  SCR#2  ---   ;WITH STRING AT PAD )
    0 SCR !  PAD 1+ C@ 1STCHAR !  HEADER  1+  SWAP
     DO I DUP DUP CR 2 SPACES . BLOCK ONEBLK ( ATARI, USE BLOK)
       ?TERMINAL IF LEAVE ENDIF
     LOOP CR CR ." END SEARCH" CR ;
  DECIMAL  ;S
```

©

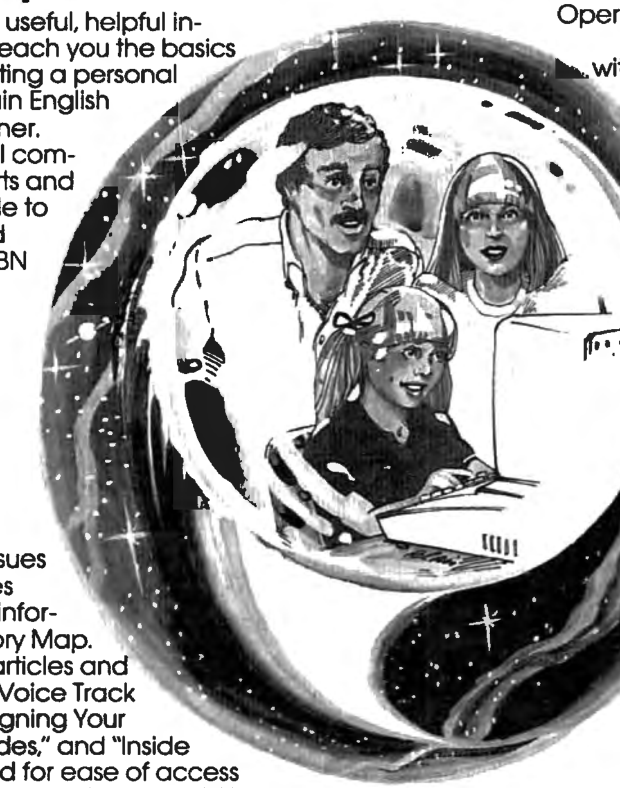
Turn To The Future With COMPUTE! Publications

The Beginner's Guide To Buying A Personal Computer

A Novice's handbook of useful, helpful information designed to teach you the basics of evaluating and selecting a personal computer. Written in plain English for the interested beginner. Complete with personal computer specification charts and buyer's guide. Applicable to home, educational, and small business buyers. ISBN 0-942386-03-5. Paperback. $3.95.

COMPUTE!'s First Book Of Atari

192 pages of useful, informative applications and programs from **COMPUTE!** magazine issues now out of print. Includes previously unpublished information including Memory Map. Contents include such articles and programs as "Adding A Voice Track to Atari Programs," "Designing Your Own Atari Graphics Modes," and "Inside Atari BASIC." Spiral bound for ease of access to listings. For Beginner level to Advanced Atari users. ISBN 0-942386-00-0. Paperback. $12.95.



Inside Atari DOS

From the authors of the Atari Disk Operating System, an exciting step-by-step guide to the DOS software. Complete with listings of commented source code and detailed explanations of each module of code. Author: Bill Wilkinson, Optimized Systems Software, Inc. Spiral bound for ease of access to listings. For Intermediate to Advanced Atari Users. ISBN 0-942386-02-7. Paperback. $19.95

COMPUTE!'s First Book Of PET/CBM

256 pages of Commodore PET and CBM articles from **COMPUTE!** magazine issues now in print. Includes such classic articles and programs as "Feed Your PET Some Applesoft," "Disk Lister: A Disk Cataloging Program," and "Cross Reference For The PET." Spiral bound. ISBN 0-942386-01-9. Paperback. $12.95

COMPUTE! Books invites dealer inquiries. Call the Toll Free Number below for Dealer Information.

COMPUTE! Magazine

A Monthly encyclopedia of informative applications articles and programs. **COMPUTE!** features articles, programs, and columns covering the spectrum of home and educational computing. Monthly reviews, complete BASIC and machine language listings of games, utilities, applications such as "Programming Your Home Insurance Inventory," "Real Estate Investment Analysis," "Telecommunications: How To Use A Modem," and much more. Written for children and parents, educators, novices to advanced programmers. Principal editorial coverage is Atari, Apple, Commodore PET/CBM, and VIC-20. Editorial coverage is expanding to include TI-99/4A, Sinclair ZX-81, and Radio Shack Color Computer. Latest issue: 224 pages.

Mail to: **COMPUTE!** Publications, P.O. Box 5406, Greensboro, NC 27403 USA

COMPUTE! MAGAZINE

My Computer Is:

☐ Commodore PET/CBM
☐ VIC-20
☐ Apple
☐ Atari
☐ AIM
☐ OSI
☐ Radio Shack Color Computer
☐ TI 99/4A
☐ Other ______
☐ Don't yet have one

For Fastest Service Call Our **Toll-Free** US Order Line **800-345-8112**
In Pennsylvania Call 800-662-2444

☐ $20.00 One Year US Subscription
☐ $36.00 Two Year US Subscription
☐ $54.00 Three Year US Subscription
☐ $25.00 Canada and Int'l Surface Mail
☐ $38.00 Europe Air Delivery (Foreign orders must be pre-paid in US Funds)

☐ Payment Enclosed ☐ VISA ☐ MasterCard ☐ American Express

Acct. No. ______ Expires __/__

Name ______

Address ______

City ______ State ______ Zip ______

Country ______

COMPUTE! Books

For fastest service, in the US call **Toll Free 800-334-0868.**
In NC call 919-275-9809.

Quantity	Price	Shipping/Handling	
____ **Beginner's Guide**	$ 3.95 ea.	+ $1.00 ea.	____
____ **First Book of Atari**	12.95 ea.	+ 2.00 ea.	____
____ **Inside Atari DOS**	19.95 ea.	+ 2.00 ea.	____
____ **First Book of PET/CBM**	12.95 ea.	+ 2.00 ea.	____
		Total	____

All orders must be prepaid (money order, check or charge). All payments must be in US funds. (Outside the US add $4.00 shipping and handling for air mail, $2.00 for surface mail.) NC residents add 4% sales tax.

☐ Payment Enclosed ☐ VISA ☐ MasterCard ☐ American Express

Acct. No. ______ Expires __/__

Name ______

Address ______

City ______ State ______ Zip ______

Country ______

Allow 4-6 weeks for delivery. Foreign surface delivery 2-3 months.

COMPUTE! Back Issues

Here are some of the applications, tutorials, and games from available back issues of **COMPUTE!**. Each issue contains much, much more than there's space here to list, but here are some highlights:

January 1981: Load PET Programs Into The Apple II, Player-Missile Graphics for Atari, The Atari DOS, The Kernel of the OSI Operating System, Fixing LOADing Problems on the PET, Spooling with the PET Disk, Expanding KIM.

February 1981: Simulating PRINT USING, Using the Atari as a Terminal for Telecommunications, Attach a Printer to the Atari, Double Density Graphing on C1P, Commodore Disk Systems, PET Crash Prevention, A 25¢ Apple II Clock.

May 1981: Named GOSUB/GOTO in Applesoft, Generating Lower Case Text on Apple II, Copy Atari Screens to the Printer, Disk Directory Printer for Atari, Realtime Clock on Atari, PET BASIC Delete Utility, PET Calculated Bar Graphs, Running 40 Column Programs on a CBM 8032.

June 1981: Computer Using Educators (CUE) on Software Pricing, Apple II Hires Character Generator, Ever- expanding Apple Power, Color Burst for Atari, Mixing Atari Graphics Modes 0 and 8, Relocating PET BASIC Programs, An Assembler In BASIC for PET, QuadraPET: Multitasking?

July 1981: Home Heating and Cooling, Animating Integer BASIC Lores Graphics, The Apple Hires Shape Writer, Adding a Voice Track to Atari Programs, Machine Language Atari Joystick Driver, Four Screen Utilities for the PET, Saving Machine Language Programs on PET Tape Headers, Commodore ROM Systems, The Voracious Butterfly on OSI.

August 1981: Minimize Code and Maximize Speed, Apple Disk Motor Control, A Cassette Tape Monitor for the Apple, Easy Reading of the Atari Joystick, Blockade Game for the Atari, Atari Sound Utility, The CBM "Fat 40," Keyword for PET, CBM/PET Loading, Chaining, and Overlaying.

September 1981: The Column Calculator, What is a Modem and Why Do I Need One?, PET, Apple, Atari: On Speaking Terms, A Tape "EXEC" for Applesoft, A Self-altering Program for Apple II, Positioning P/M Graphics and Regular Graphics in Memory, An Atari BASIC Sort, Shoot, an Arcade Game for Atari, Exploring OSI's Video Routine, PET Tape Append and Renumber, All About LOADing PET Cassettes.

October 1981: Automatic DATA Statements for CBM and Atari, VIC News, Undeletable Lines on Apple, PET, VIC, Budgeting on the Apple, Switching Cleanly from Text to Graphics on Apple, Atari Cassette Boot-tapes, Atari Variable Name Utility, Atari Program Library, Train your PET to Run VIC Programs, Interface a BSR Remote Control System to PET, A General Purpose BCD to Binary Routine, Converting to Fat-40 PET.

November 1981: SuperPet: A Preview, Japanese Micros: A First Look, Introduction to Binary Numbers, An Apple Primer, Page Flipper for Apple, An Atari Database System, A Program for Writing Programs on the Atari, Atari Textplot, OSI Relocation, The PET Speaks, Inversion Partitioning, A Personal News Service on PET, Bits, Bytes, and Basic Boole.

December 1981: Saving Fuel $$ (Multiple Computers: versions for Apple, PET, and Atari), Unscramble Game (multiple computers), Maze Generator (multiple computers), Animating Applesoft Graphics, A Simple Printer Interface for the Apple II, A Simple Atari Wordprocessor, Adding High Speed Vertical Positioning to Atari P/M Graphics, OSI Supercursor, A Look At SuperPET, Supermon for PET/CBM, PET Mine Maze Game.

January 1982: Invest (multiple computers), Developing a Business Algorithm (multiple computers), Apple Addresses, Lowercase with Unmodified Apple, Cryptogram Game for Atari, Superfont: Design Special Character Sets on Atari, PET Repairs for the Amateur, Micromon for PET, Self-modifying Programs in PET BASIC, Tinymon: a VIC Monitor, Vic Color Tips, VIC Memory Map, ZAP: A VIC Game.

February 1982: Insurance Inventory (multiple computers), Musical Transposition (multiple computers), Multitasking Emulator (multiple computers), Disassemble Apple Programs from BASIC, Plotting Polar Graphs on Apple, Atari P/M Graphics Made Easy, Atari PILOT, Put A Rainbow in your Atari, Marquee for PET, PET Disk Disassembler, VIC Paddles and Keyboard, VIC Timekeeping.

March 1982: Word Hunt Game (multiple computers), Infinite Precision Multiply (multiple computers), Atari Concentration Game, VIC Starfight Game, CBM BASIC 4.0 To Upgrade Conversion Kit, Apple Addresses, VIC Maps, EPROM Reliability, Atari Ghost Programming, Atari Machine Language Sort, Random Music Composition on PET, Comment Your Apple II Catalog.

April 1982: Track Down Those Memory Bugs (multiple computers), Shooting Stars Game (multiple computers), Intelligent Input Subroutines (multiple computers), Ultracube for Atari, Customizing Apple's Copy Program, Using PET/CBM In The High School Physics Lab, Grading Exams on a Microcomputer (multiple computers), Atari Mailing List, Renumber VIC Programs The Easy Way, Browsing the VIC Chip, Disk Checkout for PET/CBM.

May 1982: VIC Meteor Maze Game, Atari Disk Drive Speed Check, Modifying Apple's Floating Point BASIC, Fast Sort For PET/CBM, Extra Atari Colors Through Artifacting, Life Insurance Estimator (multiple computers), PET Screen Input, Getting The Most Out Of VIC's 5000 Bytes.

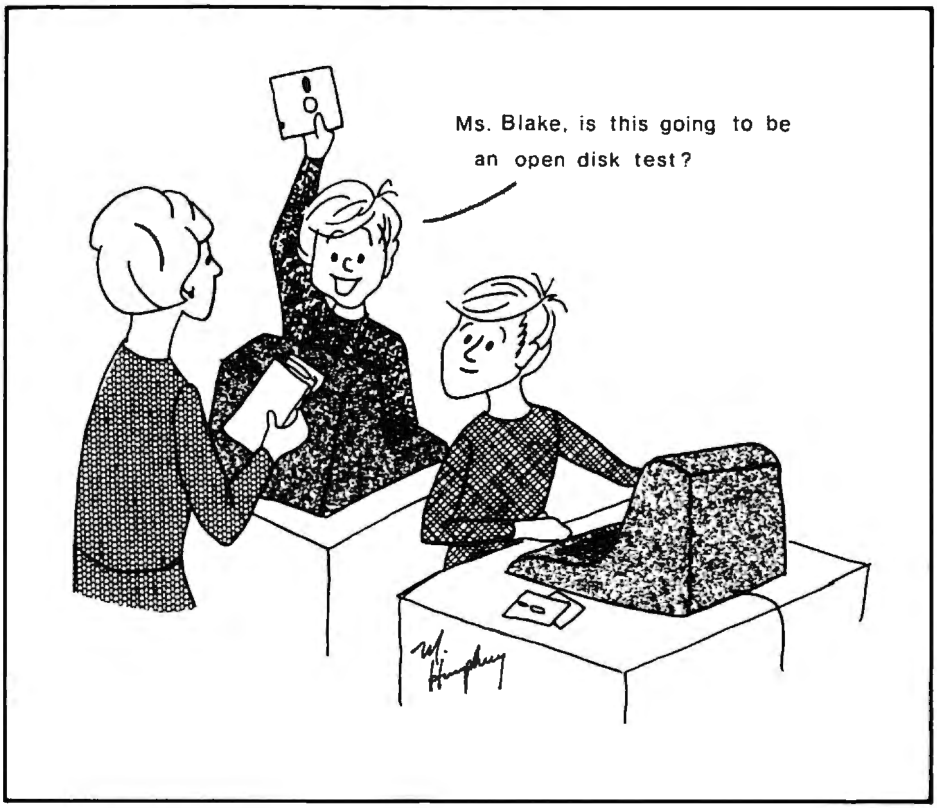
Home and Educational COMPUTING! (Fall 1981 and Summer 1981 – count as one back issue): Exploring The Rainbow Machine, VIC As Super Calculator, Custom Characters, Alternate Screens, Automatic Line Numbers, Using The Joystick (Spacewar Game), Fast Tape Locater, Window, VIC Memory Map.

Back issues are $3.00 each or six for $15.00. Price includes freight in the US. Outside the US add $1.00 per magazine ordered for surface postage. $3.00 per magazine for air mail postage. All back issues subject to availability.

In the Continental US call
TOLL FREE 800-334-0868
(In NC Call 919-275-9809)

Or write to **COMPUTE!** Back Issues, P.O. Box 5406, Greensboro, NC 27403 USA. Prepayment required in US funds. MasterCard, Visa and American Express accepted. North Carolina Residents add 4% sales tax.

COMPUTER CLASS M. HUMPHREY



CAPUTE!:

Modifications Or Corrections To Previous Articles

Improved Search For Apple II

Our thanks to Jim Gordon for the following improvement to the Apple version of "Search For PET And Apple II Plus," June 1982, #25, pg. 43. Change line 700 to:

```
700 FOR ADRES=768 TO 902: READ DTA: POKE
    ADRES,DTA: NEXT
```

and revise the following lines:

```
852 DATA 4, 200, 76, 76, 3, 162
870 DATA 76, 76, 3, 76, 119, 3
888 DATA 163, 32, 237, 253, 32, 32
894 DATA 237, 169, 160, 32, 237, 253
900 DATA 76, 108, 3
```

Self-Modifying P/M Graphics Utility Updated

Ken Grace, the author of "A Self-modifying P/M Graphics Utility," June 1982, #25, pg. 120, sent in the following update to his article.

Line 420 of Program 2 should be changed to:

```
420 IF A=15 THEN 310
```

Further testing of the program revealed that certain combinations of inputs lead to the famous "keyboard lockup" problem. The problem results from having all the deletions bunched together in lines 57-68. By splitting them up and sprinkling them among the earlier lines, the problem does not show up. Some renumbering of lines 3-55 will be needed to make room for these deletion steps. For example, lines 3-12 could be deleted by inserting a new line 13:

```
13 GOSUB 90:FOR I=3 TO 12:? I:NEXT I:GOSUB 91
```

Similarly, the number of players is obtained in line 20; therefore, the deletions in lines 64 and 65 could be done after line 20. Avoid putting the deletions inside the loops from 21 to 40 and from 42 to 51. Line 56 will no longer be needed. The final cleanup, as in line 68, would have to remain at the end, with appropriate changes in the line numbers in the PRINT (?) statements.

Shooting Stars

The following changes should be made to the PET/CBM version of "Shooting Stars" from **COMPUTE!**, April 1982, #23.

```
440 GET K$:IF K$<>" " THEN 480
550 PRINT  " SHOTS FIRED:";SH;"{LEFT} SCORE: {
    REV}";INT(H*100/SH);"{OFF} HITS:";H;
```

©

New Products

SuperPILOT Added To Instructional Development Software

SuperPILOT, an extension of the Apple PILOT software language, has been announced by Apple Computer, Inc. It joins several new products in Apple's PILOT series that help educators and industrial trainers create lessons and illustrations for computer-aided instruction.

SuperPILOT offers all the capabilities of Apple PILOT plus added features for graphic enhancement, easy debugging, and external video control.

The SuperPILOT program:

– controls external videodisc and videotape through user and computer command and response

– presents "turtle" graphics for graphics programming and discovery learning

– allows for immediate debugging of a program-in-progress, which reduces programming frustration

– displays color text on color background

– displays double-sized characters for emphasis

Also announced are two support products in Apple's PILOT family, Co-PILOT and SuperPILOT Log. Co-PILOT is a completely self-contained, self-paced interactive tutorial on two diskettes which teaches how to program in Apple PILOT. SuperPILOT Log works with SuperPILOT as an administrative record keeping program that automatically tracks test scores by item, student, or class, and can also analyze non-computer test scores entered manually.

Price And Distribution

SuperPILOT (product #A2D0051) will be available mid-July from authorized Apple dealers. Included in the SuperPILOT package is the diskette tutorial Co-SuperPILOT. The program requires an Apple II or Apple II Plus personal computer with 64K of RAM (such as a 48K Apple II Plus with a language card). The suggested retail price is $200.

A price reduction has been announced for Apple PILOT (product #A2D0028). It is now $100, a 33% reduction.

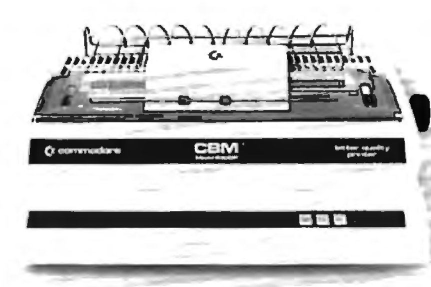
Co-PILOT (product #A2 D0050) is priced at $35, and SuperPILOT Log (product #A2D0052) has a suggested retail price of $50.

Apple Computer, Inc.
20525 Mariani Ave.
Cupertino, CA 95014
(408)973-3019

Commodore Introduces New Letter Quality Printer

The new Commodore 8300P Letter Quality Printer, designed especially for use with PET and CBM Computers, has been announced.

A version of the Diablo Model 630 Receive-Only Terminal, the 8300P includes the following standard features: immunity to electrostatic discharge, end-of-ribbon sensor, paper-out detection, cover-open interlock, internal self-test diagnostics, 320-byte printer buffer and automatic bi-directional printing.



Standard control panel features include: form-feed, pause and reset switches, as well as two lights indicating ready/error and power-on. Switch selectable features available by raising the access cover are printwheel select, pitch, parity, protocol, baud-rate and self-test.

Optional support of languages other than English is available. Optional accessories include an adjustable-width continuous forms tractor mechanism.

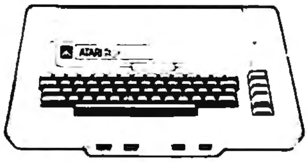
The standard ribbon supplied with the CBM 8300P is the Diablo multi-strike film ribbon. The CBM 8300P directly supports use of most Diablo metal or plastic printwheels.

An IEEE to RS-232C printer adaptor will be supplied with the printer. All CBM printers are equipped with a standard PET-IEEE interface connector.

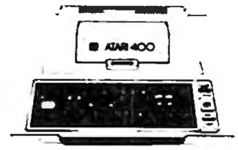
Retail price is $3400.

Commodore Business Machines Ltd.
3370 Pharmacy Avenue
Agincourt, Ontario
M1W 2K4
(416)499-4292

ATARI HOME COMPUTERS



ATARI



ATARI 800		ATARI 400	
16K ...	$649	16K ...	$269
32K ...	$729	32K ...	$389
48K ...	$769	48K ...	$489

410 Recorder	$76.00
810 Disc Drive	$449.00
822 Printer	$269.00
825 Printer	$589.00
830 Modem	$159.00
820 Printer	$259.00
850 Interface	$169.00
New DOS 2 System	$29.00
CX30 Paddle	$18.00
CX40 Joy Stick	$18.00
CX853 16K RAM	$77.95
Microtek 16K RAM	$74.95
Microtek 32K RAM	$119.95
Ramdisk (128K)	$429.95
Intec 48K Board	$219.95
Intec 32K	$119.95
One year extended warranty	$70.00
481 Entertainer	$69.00
482 Educator	$130.00
483 Programmer	$49.00
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
CX4103 Statistics I	$20.00
CX8107 Stock Analysis	$20.00
CXL4015 TeleLink I	$23.00
HOME STUDY	
CX4101 An Invitation to Programming I	$20.00
CX4106 An Invitation to Programming 2	$23.00
CX4117 An Invitation to Programming 3	$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
CXL 4007 Music Composer	$34.00
CX4123 Scram, uses joystick	$20.00
CX4112 States & Capitals	$13.00
CX4110 Touch Typing	$20.00
HOME ENTERTAINMENT	
PAC MAN	$35.00
CENTIPEDE	$35.00
CAVERNS OF MARS	$32.00
CXL4013 Asteroids	$29.00
CXL4004 Basketball	$27.00
CX4105 Blackjack	$13.00
CXL4009 Computer Chess	$29.00
CXL4012 Missile Command	$29.00
CXL4008 Space Invaders	$29.00
CXL4011 Star Raiders	$39.00
CXL4006 Super Breakout	$29.00
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
CXL 4018 PILOT	$72.00
CX405 PILOT (Educational)	$105.00

Visicalc	$189.00
Letterperfect (Word Processor)	$109.00
Data Soft Textwizzard	$89.00
Canyon Climber	$24.00
Tumble Bug	$24.00
Shoot Arcade	$24.00
Pacific Coast	$24.00
Bishops Square	$24.00
Micro Painter	$27.00
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
Crystal Software	
Bermuda Fantasy	$26.00
Beneath Pyramids	$20.00
Galactic Quest	$26.00
House of Usher	$20.00
Forgotten Island	$26.00
Haunted Palace	$33.00
Compumax (Acct. Rec., Gen. Ledger, Inventory, Payroll, ea.)	$110.00
Synapse	
File Manager 800	$79.95
Dodge Racer	$19.00
Chicken	$24.00
Slime	$24.00
Nautilus	$24.00
Disk Manager	$24.00
Fort Apocalypse	$24.00
Assembler	$39.00
Protector	$24.00
EXPY (Automated Simulation)	
Ricochet	$14.50
Crush, Crumble & Chomp	$24.00
Star Warrior	$29.00
Rescue at Rigel	$24.00
Datestones	$16.00
OnLine	
Jaw Breaker	$27.00
MouskATTACK	$31.00
Invasion Orion	$18.50
Mission Asteroid	$22.00
The Next Step	$34.00
Softporn	$27.00
Wizzard & Princess	$29.00
Arcade Plus	
Ghost Hunter (cassette)	$24.00
Ghost Hunter (disk)	$30.00
K-Byte	
K-BYTE Krazy Shoot Out (ROM)	$39.00
K-DOS	$69.00
K-Razy Kritters	$39.00
K-Star Patrol	$39.00
K-Razy Antiks	$39.00
Stick Stand	$6.99

Texas Instruments

TI-99/4A $299

PHA2100 R F Modulator	$29.00
PHP1600 Telephone Coupler	$179.00
PHP1200 Peripheral Expansion Box	$199.00
PHP1220 RS 232 Card	$143.00
PHP1240 Disk Controller	$199.00
PHP1250 Disk Drive	$319.00
PHP1100 Wired Remote Controllers (pair)	$31.00
PHP1260 32K RAM	$229.00
PHP Printer Solid State	$319.00
PHM3006 Home Financial Decisions	$26.00
PHM3013 Personal Record Keeping	$43.00
PHD5001 Mailing List	$60.00
PHD5021 Checkbook Manager	$18.00
PHM3008 Video Chess	$60.00
PHM3010 Physical Fitness	$26.00
PHM3009 Football	$26.00
PHM3018 Video Games I	$26.00
PHM3024 Indoor Soccer	$26.00
PHM3025 Mind Challengers	$22.00
PHM3031 The Attack	$35.00
PHM3032 Blasto	$22.00
PHM3033 Blackjack and Poker	$22.00
PHM3034 Hustle	$22.00
PHM3036 Zero Zap	$18.00
PHM3037 Hangman	$18.00
PHM3038 Connect Four	$18.00
PHM3039 Yahtzee	$22.00
Tombstone City 21st Century	$34.00
Munch Man	$34.00
T I INVADERS	$34.00
CAR WARS	$34.00

Printers

Smith Corona TP1	$699.00
Centronics 739-1	$519.00
Centronics 739-3	$619.00
Diablo 630 Special	$1799 00
Epson	
MX80 w/Graftrax	$449 00
MX80FT III	$Call
MX100	$Call
NEC	
8023	$549 00
7710/7730	$2399.00
3510/3530	$1789.00
Okidata	
82A	$499 00
83A	$749 00
84	$1129.00
Citoh Starwriter	
F10-40 CPS	$1469.00
F10-55 CPS	$Call
Prowriter	$499.00
Talley	
8024-L	$1629 00
IDS	
Prism	$Call
MPC Apple Parallel Board & Cable	$69 00
2 Meter RS232-RS232	$29 95

Cables Available For Most Interfacing Purposes

Computer Covers

ATTRACTIVE DUST COVERS FOR YOUR COMPUTER AND DISK DRIVE

Atari 400	$6.99
Atari 800	$6.99
Atari 810	$6.99
Commodore VIC-20	$6.99
Commodore 8032	$14.99
Commodore 8050/4040	$10 99

All Atari Covers are Beige.

All Commodore covers are Royal Blue.

Most software for Atari 400/800 available on cassette or disk.

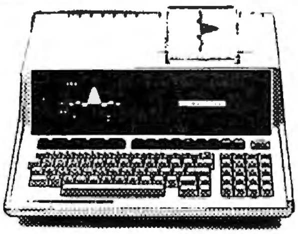
computer mail order west

CALL TOLL FREE **800-648-3311**

IN NEVADA, CALL (702) 588-5654

P.O. BOX 6689, STATE LINE, NEVADA 89449

HEWLETT PACKARD



HP•85 $1899

HP•86	$Call
HP•87	$1769.00
HP•125	$1999.00
HP•85 16K Memory Module	$169.00
5¼" Dual Master Disc Drive	$1769.00
Hard Disk w/Floppy	$4349.00
Hard Disk	$3440.00
"Sweet Lips" Plotter	$1149.00
80 Column Printer	$799.00
87 CP/M	$399.00
87 128K Memory	$610.00
87 Visicalc	$205.00

HP•41CV Calculator $239.00

41C	$189.00
11C	New Low Price $79.00
12C	$114.00
34C	$114.00
38C	$114.00
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

NEC

8001-A	$749.00
8031	$749.00
8012	$549.00
Accounts Receivable	$295.00
Word Processing	$395.00
General Accounting	$295.00
Inventory Control	$295.00
Job Cost	$295.00

Maxell Disks

MD I (box of 10)	$36.00
MD II (box of 10)	$46.00
MFD I (8")	$44.00
MFD II (8" Double Density)	$54.00
Syncom (box of 10)	$29.00

commodore

BUSINESS MACHINES

SOFTWARE

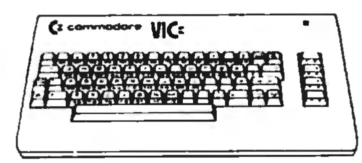
Word Pro 5 Plus	$319.00
Word Pro 4 Plus	$299.00
Word Pro 3 Plus	$199.00
Commodore Tax Package	$589.00
Visicalc	$189.00
Medical Billing	$449.00
The Source	$89.00
OZZ Information System	$289.00
Dow Jones Portfolio	$129.00
Pascal	$239.00
Legal Time Accounting	$449.00
Word Craft 80	$289.00
Power	$79.00
Socket-2 Me	$20.00
Jinsam	$Call
MAGIS	$Call
CPA	$Call
Real Estate Package	$Call
The Manager	$209.00
Softrom	$129.00
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

CBM 8032 $1039

4032	$969.00
4016	$769.00
8096 Upgrade Kit	$369.00
Super Pet	$1599.00
2031	$529.00
8250 Doubled Sided Disk Drive	$1699.00
D9060 5 Megabyte Hard Disk	$2399.00
D9090 7.5 Megabyte Hard Disk	$2699.00
8050	$1299.00
4040	$969.00
8300 (Letter Quality)	$1799.00
8023	$769.00
4022	$499.00
Pet to IEEE Cable	$37.00
IEEE to IEEE Cable	$46.00
Tractor Feed for 8300	$240.00

VIC 20 $239
VIC 64 Call

VIC 6 Pack Program	$44.00
VIC 1530 Commodore Datassette	$69.00
VIC 1540 Disk Drive	$499.00
VIC 1515 VIC Graphic Printer	$339.00
VIC 1210 3K Memory Expander	$32.00
VIC 110 8K Memory Expander	$53.00
16K VIC Expansion	$99.00
VIC 1011 RS232C Terminal Interface	$43.00
VIC 112 VIC IEEE-488 Interface	$86.00
VIC 1211 VIC 20 Super Expander	$53.00
VT 232 VICTerm I Terminal Emulator	$9.00
VIC 1212 Programmers Aid Cartridge	$45.00
VIC 1213 VICMON Machine Language Monitor	$45.00
VIC 1901 VIC AVENGERS	$23.00
VIC 1904 SUPERSLOT	$23.00
VIC 1906 SUPER ALIEN	$23.00
VIC 1907 SUPER LANDER	$23.00
VIC 1908 DRAW POKER	$23.00
VIC 1909 MIDNIGHT DRIVE	$23.00
VIC 1910 RADAR RAT RACE	$23.00



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
VT 106A Recreation Pack A	$44.00
VT 107A Home Calculation Pack A	$44.00
VT 164 Programmable Character/Gamegraphics	$12.00
Household Finance	$27.00
VIC Games	$19.00
VIC Home Inventory	$13.00
VIC Rec/Ed II	$13.00
VL 101 Introduction to Computing	$19.00
VL 102 Introduction to BASIC Programming	$19.00
VM110 VIC 20 Programmers Reference Guide	$15.00

Apple

Call for availability and prices
on all Apple computers and peripherals.

In-stock items shipped same day you call. No risk, no deposit on C.O.D. orders. Pre-paid orders receive free shipping within the continental United States with no waiting period for certified checks or money orders. All prices shown are cash prices. Add 3% for Mastercard and Visa. NV and PA residents add sales tax. All items subject to availability and price change.

Televideo

910	$579.00
912C	$699.00
920C	$749.00
925C	$749.00
950	$939.00
Call for computers	
802	$Call
802H	$Call
816	$Call
806	$Call

Monitors

BMC 12" Green	$85.00
Amdek	
300G	$169.00
Color I	$339.00
Color II	$699.00
Color III	$429.00
TI 10" Color	$349.00
Zenith 9" (Green)	$119.00

Modems

Hayes	
Smart	$239.00
Chronograph	$199.00
Micromodem II	$279.00
Micromodem 100	$309.00
Novation Auto	$239.00
D Cat	$169.00
Cat	$159.00
Anchor Modem	$79.00

Nev. & Pa. residents add sales tax.

computer mail order east

CALL TOLL FREE **800-233-8950**

INTERNATIONAL CALLS AND IN PA. CALL (717) 327-9575

477 E. THIRD ST., WILLIAMSPORT, PA 17701

Starship Duel: A Two-Player Computer Game From Program Design

Program Design has released Starship Duel, a two-player computer game written by John Kanopa.

The object of the game is to destroy the opponent's fleet of starships, while losing as few of your own starships as possible. The greater the number of ships remaining in your fleet after the opposing fleet has been destroyed, the higher the score.

A starship's laser fires only in the direction that the ship moves. Thus, quick handling of the joystick is needed to chase the enemy, or to get out of its way.

Each of the ten ships in a fleet has a limited amount of ammunition. If it is used up, the ship is expended. It is possible to replenish a ship's ammunition supply by hitting a white "X" that occasionally pops up on the screen. But this requires quick action, for the "X" only remains on the screen for a brief moment.

Starship Duel consists of four games. Game 1 is the simplest: one-on-one starship combat until one fleet is destroyed. In Game 2 the starships become partially or totally invisible as they move toward the left and right edges of the screen. They can still be destroyed – if the opponent knows where they are hiding. Game 3 has a blinking phantom ship that moves independently across the field of battle. If the phantom ship collides with another ship, the second ship is destroyed. However, if a player hits a phantom ship with his or her laser fire, the phantom ship becomes that player's ally, and will only destroy the opponent's ships. Game 4 is a combination of Game 2 and 3.

Starship Duel is available for use on Atari 400/800 computers with a memory of at least 16K. Available on cassette, it retails for $19.95.

Program Design, Inc.
11 Idar Court
Greenwich, CT 06830
(203)661-8799

Computer Furniture

H.S.P. (Health Science Products, Inc.) introduces its ergonomically designed Computer Furniture, the DataLeggett.

This split-level CRT Workstation is comfort engineered to meet individual needs. A major feature of the Workstation is the frontal placement of the copy holder, or leggett. The

ECLECTIC SYSTEMS CORPORATION
Order TOLL FREE 1-800-527-3135

ECLECTIC SYSTEMS

P.O. Box 1166 16260 Midway Road
Addison, TX 75001 (214) 661-1370

Authorized Commodore service center

Sales and Service of the complete line of Commodore products

In a hurry? Check our modular exchange program

commodore

HARDWARE:	
CBM 8032 Computer, 80 Column	$1065
CBM 8050 Disk Drive	1299
CBM 4032 Computer, 40 Column	965
CBM 4040 Disk Drive	965
CBM 4022 Printer	595
CBM VIC 20 Computer	259
CBM VIC 1515 Printer	349
CBM 8300P (DIABLO)	1799
CBM VS100 Cassette	65
PET to IEEE Cable	33
IEEE to IEEE Cable	39

SOFTWARE:	
OZZ	289
Wordcraft 80	289
The Manager (Data Base)	240
Wordpro 5+	319
Wordpro 4+	299
VISIC ALC	169
MUMPS for Super PET	299
CMS Accounting System	call
Assembler Development Package	77
BASF Diskette, Box of 10	30

ANS MUMPS Programmer s Reference Manual	$17.50
MUMPS Pocket Guide	2.00
Computer Programming in ANS MUMPS	17.50
RS232 Interface for Commodore VIC 20	39 95
EIO has two serial asynchronous RS232 ports, two parallel ports with handshaking, one shift register, two 16 bit timers, and room for two optional buffer IC's.	188.00
EIO-C Alternate character generator ROM board for screen display allows you to display characters or graphics of your choice Alternate characters are soft selectable EIO board required.	Call for price
EIO-TX Terminal ROM for EIO firmware to turn your CBM into a communications terminal. Store and transmit from disk to remote host and terminal EIO board required	49.95
EIO-RS232 Cable for EIO board	28 00

BOOKS:

Commodore Software Encyclopedia 2nd edition	9.95
The PET Revealed	19.95
Library of PET Subroutines	19 95
PET Interfacing	16 95
PET Basic	12 95
PET and IEEE 488 Buss	15 00

OTHER:

EPROM Burner for CBM burns 2716, 2732 & 2532	89 95
Software for EPROM Burner on CBM	15 95
D.C Hayes Smart Modem	239 00

ESC-100 RS232 interface manual selector —switch select between device A, B, or C; switching 25 conductors. Requires no input power and uses receptacle type DB25S. 140.00

ESC-120 M43 TTY-EIA/CURRENT LOOP interface unit—converts TTL level signals from the M43 to EIA or 20ma outputs Convert EIA or 20ma inputs back to TTL level signals for the M43. Single printed circuit mounts inside the M43 Comes with interface cable. 65 00

ESC-140 RS232 standard interface cable kit. Build your own cable
7 conductor 16 00
12 conductor 19.95

ESC-150 The Electronic Switch/Poller allows a single computer serial port to selectively communicate with an assortment of peripheral devices As many as seven serial asynchronous and six parallel devices may be selected This concept effectively creates a simplified multi-drop, polled environment. The computer maintains control of the network and may seize any of the peripherals by transmitting a simple escape sequence to the "ESC-150" 488.00

ESC-170 RICKETYMETEr-Indispensable hand-held tool for trouble-shooting, checkout and installation of RS232 type serial line communications systems 69 95

ESC-180 RICKETYTRAp is a small hand-held device which when interposed in an EIA/RS232 line can monitor and trap on any specific ASCII, binary, or control character The RICKETYTRAp can also monitor and trap on any specific range of characters bit selectable This device can also check for type of parity being sent and proper framing The RICKETYTRAp is switch selectable for 1 or 2 stop bits and can operate at speeds of 150 to 9600 baud Call for price

ESC-200 PREDITOR is a small microprocessor with ROM firmware, RAM, and 8 RS232 asynchronous serial input/output ports. The PREDITOR is available in either stand-alone, self-contained, or rack-mounted models. The PREDITOR can be programmed by firmware to function as a prompter and editor in a distributed processing network, transforming 4 dumb asynchronous terminals into 4 intelligent terminals communicating with 4 computer asynchronous input/output ports. Call for price

ESC-861M The DATA CONCENTRATOR accepts data from a multitude of inputs—printers, CRTs, parallel ports, status lines, etc —then transmits the composite (multiplexed) data stream down a high speed synchronous serial line to a remote DATA CONCENTRATOR The remote DATA CONCENTRATOR separates (demuxes) the composite data into its original form to be transmitted to the computer ports. Alternately, multiple computer port data is multiplexed, transmitted down the high speed synchronous serial line, demuxed, and finally dispatched to the individual terminals 850 00

Call toll-free for shipping charges.

Order TOLL FREE 1+800-527-3135

10 AM to 4 PM CDT Monday through Friday

Texas residents call 1+214-661-1370

VISA, MASTER CHARGE, MONEY ORDERS, AND C.O.D. "Certified Check" accepted.

PERSONAL CHECKS REQUIRE 2-WEEK WAITING PERIOD.

Units in stock shipped within 24 hours, F.O.B. Dallas, Texas.

All equipment shipped with manufacturer's warranty.

Residents of Texas, Louisiana, Oklahoma City and Tulsa, Oklahoma must add applicable taxes.

Be sure to write to the address at the top of this ad for more information. Dealer inquiries welcome.

DataLeggett holds the source document in front of the operator between keyboard and screen, so the operator doesn't need to constantly look off-side at reference material. It can be adjusted in degree of tilt and lifts up for access to a storage area.

The height adjustable video display platform slants for optimum glare control from overhead lighting and windows. This design provides proper viewing position and distance from video display to reduce eye strain.

H.S.P. Computer Furniture
P.O. Box 5545
Birmingham, AL 35207

File Management Software For The Apple III

Apple Computer, Inc. announces Quick File III, a filing system for managing small to medium size collections of information on the Apple III personal computer.

With Quick File III, a doctor, small business owner, homemaker, or scientist can quickly turn receipts, notes, lists, and schedules into coherent files and reports. Quick File III allows for simple arrangement of records in alphabetic, numeric, date, or time order and saves time and effort in producing repetitive reports, calculations, and corrections. Two types of report formats – tables (rows and columns) and labels or index cards – can be easily created and printed.

Quick File III...

– allows the user to design forms to meet special needs

– allows categories to be added and deleted without retyping previously-held information

– can selectively search, display, and summarize records

– can view many records simultaneously

– can "talk" to Apple Writer III and other ASCII character files

The program provides these additional convenient reporting features:

– calculates totals and subtotals of numeric information

– contains a calculated column (for percentages, the sum of two other columns, etc.)

– allows for the choice of which rows and columns are printed and in what order.

The program (product #A3D0020) requires an Apple III system with at least 128K bytes RAM. It has a suggested retail price of $100, and will be available in late August from authorized Apple dealers.

Apple Computer, Inc.
20525 Mariani Ave.
Cupertino, CA 95014
(408)973-3019

Colorport Cartridge For TRS-80 Color Computer

The Colorport plug-in cartridge adds I/O capability to the TRS-80 Color Computer, resulting in a cost-effective 6809-based control system. This unit adds two fully programmable 8-bit bidirectional parallel ports with full handshaking, which can be configured by the user for versatile interfacing to peripherals. Interrupts are supported, and important computer voltage and logic lines are brought out to the standard 44-pin edge connector. The Colorport has its own power supply, ensuring no system power degradation.

A socket in the cartridge allows insertion of either 2K bytes of RAM or 2K bytes of EPROM. This allows software for the control of I/O operations to be stored separately from the main user memory space. Provision is also made for selection of autostart of the memory in the cartridge and of synchronous reset of the Colorport and the computer.

The Colorport cartridge comes complete with power supply and full instructions, and sells without any memory for $129.95. 2K RAM chips are available for $19.95 each, 2K EPROMS are available for $12.95 each.

Maple Leaf Systems,
P.O. Box 2190
Station "C", Downsview
Ontario, Canada M2N-2S9

Educational Shows Scheduled

ECCO, The Educational Computer Consortium of Ohio, presents the Second Annual Educational Computer Fair on October 16, 1982, at Cleveland State University.

Forty workshops for beginning and experienced computer users, small discussion groups, audio-visual displays, vendor exhibits, and student demonstrations will be held. This is a fair for educators K through College, *by* educators, *for* educators.

For further information contact:

Ellen Richman
ECCO Coordinator
4777 Farnhurst Rd.
Cleveland, OH 44124

* * * * * * *

Commodore is planning a series

DISCOUNT COMPUTER SOFTWARE ACCESSORIES

apple® commodore® IBM® ATARI®

APPLE

APPLE	Retail	Discount
Crossfire	$ 29.95	$ 21.00
Cannonball Blitz	34.95	25 00
Mouskattack	34.95	25.00
Bandits	34.95	25.00
Lemmings	29.95	21.00
A2-FS 1	29.95	21.00
AP-PBI (Pinball)	29.95	21.00
Goldrush	34.95	25.00
Deadline	49.95	36.00
Eliminator	24.95	18.00
Raster Blaster	29.95	21.00
PFS	125.00	90.00
D.B. Master	229.00	165.00
T.G. Game Paddles	39.00	29.00
T.G. Joystick	59.00	44.00
Visicalc 3.3	250.00	190.00
Frogger	34.95	25.00
The Joyport	74.95	54.00
Snack Attack	29.95	21.00
Gorgon	39.95	29.00
Hi-Res Adv #2 Wiz and Princess	32.95	24.00
Hi-Res Adv #1 Mission Asteroid	19.95	14.00
Hi-Res Adv #5 Time Zone	99.95	72.00
David's Midnight Magic	34.95	25.00
The Home Accountant	74.00	54.00
Apple Panic	29.95	21.00
Bug Attack	29.95	21.00
Magic Window	99.95	72.00
Super Text II	150.00	100.00
Visitrend/Visiplot	300.00	240.00
Castle Wolfenstein	29.00	21.00

CPM

CPM	Retail	Discount
Adventures 1-12	$129.00	$ 97.00
WordStar	495 00	350.00
DataStar	350 00	275 00
Mailmerge	150.00	100.00
Supersort	250.00	175 00
SpellStar	250 00	175 00
WordMaster	150 00	100 00
CalcStar	295 00	190.00
Basic Compiler	395.00	295.00
Basic - 80	350.00	260.00
dBase II	700.00	520.00
SuperCalc	295.00	225.00
Graham Dorian - Accounts Payable	1000 00	720.00
Graham Dorian - Accounts Receivable	1000 00	720 00

IBM

IBM	Retail	Discount
Temple of Apshai	$ 39.95	$ 29.00
The Home Accountant Plus	150.00	110.00
Mathemagic	89.95	70.00
IBM Joysticks	64 95	48.00
Visicalc	200.00	160.00
Visicalc/256 K	250.00	200.00
Deadline	49.95	36.00
SuperCalc	295.00	220.00

TRS-80

TRS-80	Retail	Discount
Attack Force (d)	$ 19.95	$ 16.00
Galaxy Invasion (t)	15.95	12.00
Invasion Orion (d) (t)	24.95	18.00
Sorcerer of Siva (d) (t)	29.95	22.00
Rescue at Rigel (d) (t)	29.95	22.00
Crush, Crumble & Chomp (d) (t)	29.95	22.00
Hellfire Warrior (d) (t)	39.95	29.00
Galactic Trader (t)	14.95	11.00
Galactic Trilogy (d)	39.95	29.00
Le Stick	39.95	29.00
Sargon II (t)	29.95	21.00
Battle of Shiloh (t)	24.95	18.00
Tigers in the Snow (t)	24.95	18.00
Flight Simulator (t)	25.00	19.00
Alien Armada (d)	18.95	14.00
Adventures' 1-12 (Gold Edition) (t)	100.00	75.00

SUPER SPECIALS
Zenith 12" Green Monitor $120.00
Intec 32K Board (Atari) $85.00

T=Cassette
D=Disk
C=Cartridge

ATARI

ATARI	Retail	Discount
Bug Attack (d) (t)	$29 95	$21.00
Crossfire (d) (t)	29 95	21.00
Deadline (d)	49.95	36.00
Megalegs (t)	34.95	25.00
Ghost Hunter (d)	34.95	25.00
PacMan (c)	44.95	33.00
Centipede (c)	44.95	33.00
Tumble Bugs (d)	29.95	21.00
Action Quest (d) (t)	29 95	21.00
Battle Trek (d)	39.95	29.00
Star Warrior (t)	39 95	29.00
S C R A M (t)	24 95	18.00
Invasion Orion (d) (t)	24 95	18 00
Survival/Adventure (t)	24.95	18.00
Personal Finance Management	74.00	54.00
Jawbreaker (d) (t)	29 95	21.00
Frogger (d)	34.95	25 00
Raster Blaster (d)	29 95	21.00
Apple Panic (d)	29.95	21.00
Text Wizard (d)	99 95	75.00
Match Racers (d)	29.95	21.00
Visicalc (d)	250.00	195.00
Hi-Res Adv Wiz and Princess (d) (t)	32.95	24.00
Star Raiders (c)	49.95	36.00
Asteroids (c)	44.95	33.00
K-Razy Shootout (c)	49.95	36.00
Midway Campaign (t)	16.00	12.00
Crush, Crumble and Chomp (t)	29.95	21.00
Canyon Climber (d)	29.95	21.00

MANY MORE PROGRAMS AVAILABLE — **VISA AND MASTERCARD ACCEPTED**

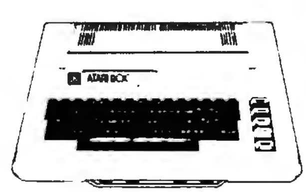
TERMS: Send check or money order for total purchase price, plus $2.00 for shipping. MI residents add 4% tax. C.O.D. accepted.

® MFGS TRADEMARK

STRÖM SYSTEMS INC.
P.O. Box 197
Plymouth, Mi. 48170
(313) 455-8022

WRITE OR CALL FOR FREE CATALOG
PHONE ORDER HOURS
4PM - 7PM MON. - FRI.
INCLUDE CARD NUMBER AND EXPIRATION DATE WITH CREDIT CARD ORDERS.
INCLUDE TYPE OF COMPUTER.

810 Disk Drive $429.00



800 Computer 16K $639.00

We also feature tremendous savings from:

- Atari
- Atari Program Exchange
- Adventure International
- Dynacomp
- Quality Software
- Avalon Hill
- Epyx
- Computer Consultants
- P.D.I.
- L.J.K.
- Synapse
- Datasoft
- United Software
- On-Line
- O.S.S.

Call or write for your FREE catalog.

Shipping costs:
Software - minimum $2.50
Hardware - prices will vary (please call)

Software Street
3392 Clipper Dr.
Chino, CA 91710
(714) 591-3061

ATARI IS A REGISTERED TRADEMARK

of educational shows for the fall of 1982. At each of the locations listed below a one-day conference on Computers in Education will be offered to teachers and administrators.

These conferences will offer a number of components including workshop sessions on various aspects of computers in education, "hands-on" experience, a keynote speaker, numerous handouts and training material, a drawing for a free computer and the debut of many new Commodore products.

Denver	–September 15
San Francisco	–September 22
Los Angeles	–September 29
San Diego	–October 13
Seattle	–October 20
Orange County	–October 27
Salt Lake City	–November 10
Phoenix	–November 17
Portland	–November 30

To register for any of these shows, write to:

Commodore Business Machines
Att'n: Jim Bussey
3330 Scott Boulevard
Santa Clara, CA 95050

or call one of the following numbers:

In Calif. (toll-free) 800-422-2122
Outside Calif. (toll-free) 800-854-8055
or 408-727-1130, ext. 213

Compumax Announces Micropers

Compumax Associates, Inc. of Menlo Park, California, announces the availability of Micropers for the Atari 800. Micropers contains both a complete payroll system and a personnel management system.

As in the previous Micropers versions, the payroll system calculates the payroll for both hourly and salaried employees and figures federal and California withholding*, social security tax, disability insurance, miscellaneous deductions, and gross and net pay. Using these figures, it prints the actual paychecks. Micropers also fills out W-2 forms and provides the values for the quarterly 941 Report. The Job Cost Report/Labor tells you how much has been spent on labor for each job, and may be used in conjunction with the Job Cost Report/Materials in Microinv to provide total job costing.

One feature that has been added is the Recap Summary Report, which gives company totals for such categories as wages, job costs, and taxes. Another feature unique to this version of Micropers is menu selections for copying your data files, making it even easier to safeguard your data.

In its personnel management capacity, Micropers provides a complete employment history for each employee, including vital statistics, status, position, and earnings, both current and previous. The master file also keeps track of accumulated deductions for each employee.

Micropers retails for $200 and comes complete with program, sample data, and thorough user documentation. BASIC source code is also included, enabling you to modify the program to suit your own particular needs. Hardware requirements include: Atari 800, 48K, 2 disk drives, and printer (optional).

Compumax Associates, Inc.
P.O. Box 7239
Menlo Park, CA 94025

*Micropers is a California payroll package. It must be customized for other states or foreign countries.

80-Column Text Editor From Metaresearch

The Metatext package by Metaresearch, Inc. comes on a single master disk, giving the user many Apple II system options.

Features of the package include: full ASCII 80-column software-packed alphanumerics, 40-column option for enhanced readability, creation routines allowing user to make custom fonts, a text formatter, and various line-oriented text editors. The package includes a serial output program which will drive most RS232 printers from the existing Game I/O connector.

The Metatext user can mix alphanumerics with graphics in arbitrary ways. This is because the font display routines, which use Apple II high-resolution graphics, have a memory-forcible blind cursor option for positioning characters.

The 80-column option is useful for editing and formatting, because the Apple display appears like the true printed page. Because CRTs vary in their resolution (a composite video monitor is best for Metatext), the package comes with 40-column font which is highly readable. As an example of arbitrary font, a Cyrillic (Russian language) text editor is supplied on the standard disk master. Editors which handle such foreign fonts, or even symbol tables for process control, are, in principle, capable of driving dot-matrix printers so that arbitrary font hard-copy can be obtained. All that is required is a dot-matrix printer which allows random-access dot printing. Then the user can create custom subroutines with which to drive the printer from editing mode.

Metatext is written in Applesoft, except for numerous instances in which machine-code speed is required. The essential machine routines can be called from within BASIC programs, as spelled out in the user manual. Thus, the user can print out in upper or lower case from BASIC, switch scrolling on and off, and so on. Graphs created in HGR (high-resolution graphics) mode can be labelled due to the blind-cursor forcing.option.

Metatext also allows for data processing of *mixed* structures. Specifically, the user can first use a MEDIT program to create columns of data, where each column is either all strings or all numbers. But different columns can be of different type. Then a BASIC germ program called PRO.DS, which processes one Data / one String in a two-column format, can be modified to handle the edited data.

With Metatext, there are no hardware modifications to the Apple II. A printer is normally driven out of pins 8 and 15 of the Apple Game I/O. The signals involved are unipolar, so a few rare printers cannot be so driven. In such a case, the user adds the circuit suggested in the Metatext manual to generate bipolar drive. The parts cost for such a unipolar-to-bipolar circuit is a few dollars. Metatext programs require the full 48K memory option for the Apple II.

The Metatext package, purchased as a single disk master along with the forty page user's manual, sells for $79.00.

For further information contact:

Metaresearch, Inc.
1100 SE Woodward St
Portland, OR 97202
(503)232-1712

Estate Tax Plan For Apple II

Aardvark Software, Inc. announces the release of its Estate Tax Plan program. Designed specifically for accountants, attorneys, insurance agents, trust officers, and financial planners, the program allows complex estate tax planning problems to be solved in a short time.

Estate Tax Plan allows the estate planner to enter a variety of factors affecting the gross estate, allowable deductions, and disposition of the client's assets via trust arrangements or bequests. It will then calculate the related effects attributable to changes in one or more of these items.

The program can construct a comparative analysis among up to four alternatives simultaneously. Estate tax planning considerations which may be examined are listed below.

– various dates of death for the client and spouse

– various valuations of the client's asset inventory

– selected marital deduction formula clauses in the client's will (e.g., maximum, "zero-tax," and equalization clause formulas)

– analysis of possible charitable bequests

– available estate tax deferral under IRC Section 6166

– available special use valuation under IRC Section 2032A

– availability and magnitude of redemptions of closely-held stock at capital gains rates under IRC Section 303

– growth rate assumptions concerning property passed to the surviving spouse

– present value analysis relative to impending estate tax liabilities

– cash needs and liquid assets available at death

Calculations performed by Estate Tax Plan result in the following seven reports: Gross Estate, Estate Tax Liability, Present Value Analysis of Estate Taxes, Deferred Payment of Estate Taxes, Deferred Payment Schedule, Liquidity Analysis, and IRC Section 303 Capital Gain.

The program was developed under the supervision of William A. Raabe, Ph.D., CPA, and is currently available for the Apple II (48K) or Western Digital Microengine. It is also expected to be available for a variety of CP/M systems in the near future.

Aardvark Software, Inc.
783 North Water Street
Milwaukee, Wisconsin 53202
(414)289-9988

Fabric Covers For The Atari

A new line of custom-tailored fabric dust covers for Atari home computers is being marketed by Empulse, a Massachusetts-based computer accessory firm.

Called "Cover-Ups," the dust covers are sewn of water-resistant rainwear poplin and are tailored to fit specific Atari

models precisely, while allowing ready access to I/O ports.

Cover-Ups are designed to provide a high-quality alternative to loose-fitting vinyl covers with no I/O access.

The dust covers are available by direct mail from Empulse in three colors: beige and chocolate brown – to match Atari computer colors – and navy blue.

Prices are $9.95 for the Atari 800 model and $8.95 for the Model 400.

Additional information is available from:

Empulse
22 Elm St.
P.O. Box 593
Great Barrington, MA 01230

Queue Computer Learning Centers

Queue, Inc. is offering a turnkey Computer Learning Center to schools and private investors, preferably educators. The Computer Learning Center will combine popular, inexpensive microcomputers and off-the-shelf software into an organized curriculum in computer programming, computer literacy, the traditional academic areas and specialized test preparation, such as SAT's. Queue's Computer Learning Centers are available for $15,000, and include all necessary hardware, software, course outlines and brochures, literature and training to run a complete profit or non-profit Learning Center. Lease plans are also available, starting as low as $500.00 per month. For information contact:

Jonathan D. Kantrowitz
c/o Queue, Inc.
5 Chapel Hill Drive
Fairfield, CT 06432
(203)335-0908.

New Journal Calls For Papers

A new quarterly, *The Journal of Computers Reading & Language Arts* (CRLA), is ready to receive papers. The journal's purpose is to support the rapidly growing interest in computers and their relationship to reading/language arts and related issues. The theme of the journal will be pragmatic in perspective. It will emphasize presenting papers which

have clear classroom/teaching implications.

The journal will be interdisciplinary and directed toward an audience of reading-language arts teachers, educational specialists, classroom teachers, educators of teachers and educational researchers.

Gerald H. Block, ALP Reading Clinics in Berkeley and Oakland, serves as editor of CRLA. Danny B. Daniel, University of Minnesota; Peter Joyce, York Board of Education, Toronto; Dorothy Judd, Northern Illinois University; George E. Mason, University of Georgia; Barbara Micetich, Diocese of Washington, DC; and Richard L. Shell, University of Cincinnati, serve as the editorial board.

Papers or requests for information should be sent to:

Gerald H. Block
CRLA
P.O. Box 13039
Oakland, CA 94661.

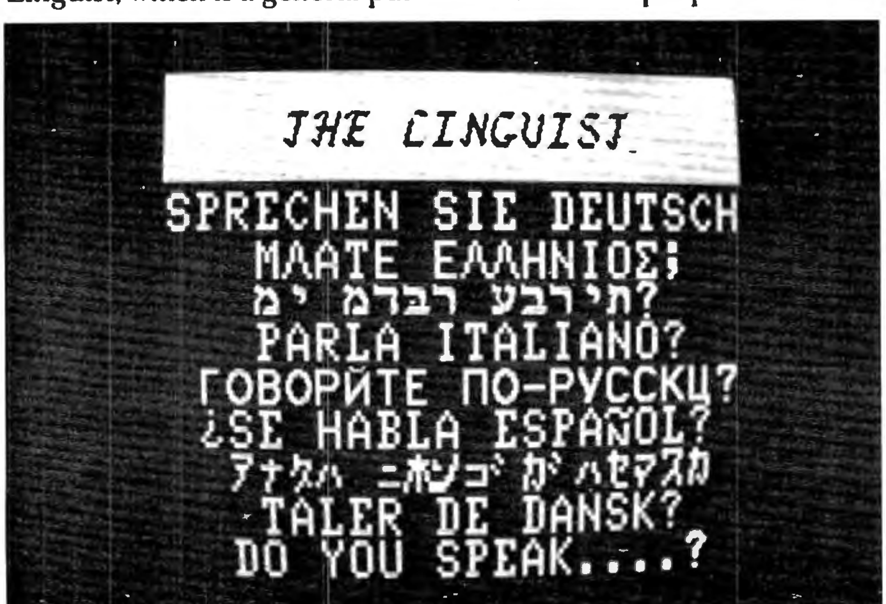
Foreign Language Program For Apple

Synergistic Software announces the release of a new educational software program called The Linguist, which is a general purpose foreign language translation and tutorial program for the Apple II computer. It allows the Apple to correctly print the foreign alphabets used by such languages as Hebrew, Russian, Japanese, Greek, German, plus the Romance languages and English. This unique program can work with words, phrases, definitions, technical terms, or phonetic pronunciations. Thus, it can be used by those learning a foreign language or those who wish to learn English.

The user of The Linguist types in the words, phrases, or definitions he would like to learn. Then the program will test the user on these words using hints. The Linguist will keep score and correct mistakes. If a phonetic pronunciation is desired, the user can input which pronunciation guide he would like to use (from American Heritage Dictionary, the International Phonetic Alphabet, or the Trager-Smith Phonemes). Then a word's correct pronunciation is easy to look up. The Linguist can operate with one or two stored languages with a maximum storage capacity of 4400 words, 2600 definitions, or 2000 foreign phrases.

The Linguist can be used for a number of purposes. If a foreign language teacher would like to drill a class on new vocabulary, The Linguist can be used to teach without supervision. If business people or tourists want to brush up on a foreign language before traveling, The Linguist can store the terms and quiz the person on their definitions and use. The Linguist requires an Apple II Computer, Applesoft, 48K, DOS 3.3. The price is $40.

Synergistic Software
830 North Riverside Drive
Suite 201
Renton, WA 98055
For ordering call (800)426-6505



File II For PET And VIC

File is a general purpose cassette-based file system for the PET/CBM, or VIC-20 computers. It will allow you to construct, sort, maintain, and print out a wide range of data types, such as mailing lists, accounts, book lists, etc. File allows the user to define the record format and is limited only by available memory. Commands include: LOAD, DUMP, PRINT (screen or printer options), ADD, CHANGE, REMOVE, SORT, and more.

Requires minimum of 8K PET/CBM or VIC-20 with 3K expansion cartridge. Expands automatically to available memory. Package includes software on cassette and complete documentation. Price is $9.95.

Complete catalog of products is available. Please specify type of computer.

Kinetic Designs
401 Monument Rd. #171
Jacksonville, FL 32211

New Book Aims At Consumer Protection

Questions concerning how to resolve computer consumer problems led to the publishing of

Consumer Protection for the Microcomputer Owner. The author is Attorney L. J. Kutten.

The 35-page booklet covers the steps to consider before making a purchase. Information about the general Law of Sales is given to advise the Buyer of the legal issues that can arise if he is not careful, such as: When is a sale valid? Are advertised prices binding? What can be done about partial delivery? What and when must the Seller deliver? A general comparison of the local computer store and mail order is given along with a brief introduction to Federal Mail Order Law. There is also a discussion of the different ways to pay for a purchase and some of the problems that each one entails. An introduction into warranties is given along with a discussion of warranty disclaimers. Hints on how to legally reject and revoke a prior acceptance of a product are included. Finally, there are suggestions on how to complain effectively, and various miscellaneous hints every computer purchaser will find useful. The booklet aids not only the consumer, but also the seller in becoming more effective.

The booklet is available through the mail for $15.00. There is an additional $3.00 charge for C.O.D. orders. Missouri residents should add 4.625% sales tax. Send check or money order to:

L. J. Kutten, Attorney at Law
201 S. Central Ave., POB 16185
St. Louis, MO 63105

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

AB Computers 64,65,127
ASAP Computer Products, Inc. 121
Aardvark Technical Services Ltd. 81
Abacus Software 22
The Alien Group 77
Amplify, Inc. 174
Apogee Software 124
Arcade Plus, Inc. 13
Artworx 55
The Arma Design Group 53
BBI 170
Batteries Included 163
Byte-A-Bit Computing Co. 97
BYTE Books 47
Byte Microsystems Corp. 67
C-Mart 152
CE Software 147
CFI 107
Canadian Micro Distributors 19
Comm*Data Systems, Inc. 59
Commodore Business Machines .... BC
Computer Age 191
The Computer Bus 17
Computer House 186
Computer Mail Order 176,177
Computer Outlet 189
Computer Seen 157
CompuServe 77
ComputerMat 154
Computertime, Inc. 170
Connecticut MicroComputer Inc. 15
Cosmic Computers Unlimited 190
Creative Software 25
Data Equipment Supply Corp. 155
Dataview Ltd. 49
Don't Ask Computer Software 43
Dunham Software and Consulting Co. 38,97
Dynacomp, Inc. 35,36,37
Dynamic Technologies 118
ECX Company 115
Eastern House Software 16,183
Eclectic Systems Corporation 161,179
Educational Software Inc. 79
English Software Company 147
H W Electronics 109
Harli Software 166
High Country Microsystems 143
Human Engineered Software 101
Hypertech 167
IDSI 23
InHome Software 57
Interlink, Inc. 16
Krell Software 71
Leading Edge Products .... IFC, IBC
The Library of Computer and Information Sciences 33
Lightning Software 17
Little Wizard Distributing 167
London Software 113
Lyco Computer Marketing and Consulting 185
MIS 141
MMG Micro Software 53
MTG Technical Sales 182
MW Software 57
Magic Carpet Software 154
Micro Computer Service Center 59
Micro Printer Marketing 21
Micro World Electronix Inc. 63
Micro-Ed, Inc. 89
Micromail 47
Microsoft 4
MicroSpec Ltd. 98
Mideastern Software 157
Midwest Micro Associates 140
Midwest Software 87
Mind Science Foundation 124
Mosaic Electronics, Inc. 11
New England Electronics Company 2,3
Nüfekop 92
On Line Software 165
Optimized Data Systems 162
PR Software 166
P.R.I.C.E. 38
Pacific Exchanges 63,83,144,170
Parsec Research 92
Percom Data Co., Inc. 7
Peripherals Unlimited 31
Philadelphia Computer Discount 139
Precision Technology, Inc. 102
Pretzelland Software 118
Pribusin, Inc. 53
Professional Software 1,9
Program Design, Inc. 22
The Program Store 40,41
The Programmer's Institute 27
Protecto Enterprizes 187
Quality Software 111
Questar International 45
RAR-TECH 57
RC Electronics 174
Random Access Microware 97
William Robbins 102
Royal Software 125
Skyles Electric Works 102,159
Small Systems Engineering, Inc. 61
The Software Connection 81
Software Galore 174
Software Street 181
Sport 'N Sound Electronics 178
Star Software 154
Strom Systems Inc. 181
subLogic Communications Corp. 74
Sunrise Electronics 57
Sunrise Software 27
Swifty Software, Inc. 73
Syncom 15
Synergistic Software 111
T.H.E.S.I.S. 147
Tara Computer Products 143
Tele-games 47
Tiny Tek, Inc. 94
Totl Software 165
University Microfilms International 174
Vervan Software 94
Victory Software 165
Voicetek 68
John Wiley & Sons, Inc. 25
Wunderware 165

COMPUTE! Publications

COMPUTE! Magazine 29
COMPUTE! Customer Service 173
COMPUTE! Back Issues 172
COMPUTE! Books 171
Every Kid's First Book Of Robots and Computers 71
Programming The PET/CBM 128,129
First Book Of VIC 167

COMPUTE!

For Fastest Service,
Call Our **Toll-Free**
US Order Line
800-345-8112
In Pennsylvania call 800-662-2444

My Computer Is:

☐ PET ☐ Apple ☐ Atari ☐ OSI ☐ VIC-20 ☐ TI 99/4A ☐ Sinclair ZX-81
☐ Radio Shack Color Computer ☐ Other ________ ☐ Don't yet have one...

☐ $20.00 One Year US Subscription
☐ $36.00 Two Year US Subscription
☐ $54.00 Three Year US Subscription

(Readers outside of the US, please see our foreign readers subscription card or inquire for rates).

Name ________

Address ________

City ________ State ________ Zip ________

☐ Payment Enclosed ☐ VISA
☐ MasterCard ☐ American Express

Account No. ________ Expires ____/____

8 9 10 11 12

COMPUTE!

Foreign Readers

Subscription rates outside the US:

☐ $25.00 Canada FI=2
☐ $38.00 Europe/Air Delivery FI=3
☐ $48.00 Middle East, North Africa, Central America/Air Mail FI=5
☐ $88.00 South America, South Africa, Australasia/Air Mail FI=7
☐ $25.00 International Surface Mail (lengthy, unreliable delivery) FI=4,6,8

Name ________

Address ________

City ________ Postal Code ________

Country ________

Payment must accompany this card.
Payment in US Funds drawn on a US Bank; International Money Order; or charge card: ☐ VISA ☐ MasterCard ☐ American Express

Account No. ________ Expires ____/____

8 9 10 11 12

COMPUTE! Books

Quan.	Title	Price		S/H	Total
____	The Beginner's Guide to Buying A Personal Computer	$ 3.95	+	$1.00*	____
____	COMPUTE!'s First Book of Atari	12.95	+	2.00*	____
____	Inside Atari DOS	19.95*	+	2.00*	____
____	COMPUTE!'s First Book of PET/CBM	12.95	+	2.00*	____
____	Programming the PET/CBM	24.95	+	3.00**	____
____	Every Kid's First Book of Robots and Computers	4.95	+	1.00*	____
____	COMPUTE!'s First Book of VIC	12.95	+	2.00*	____

*$4.00 / **$9.00 For air mail outside US.

For Fastest Service
Call Our **TOLL FREE**
US Order Line
800-334-0868
In NC call 919-275-9809

All orders must be prepaid (money order, check, or charge). All payments must be in US funds. NC residents add 4% sales tax.

☐ Payment enclosed

Please charge my:
☐ VISA ☐ MC ☐ Am. Express

Acc't No. ________

Expires ____/____

Name ________

Address ________

City ________ State ________ Zip ________

Country ________

Allow 4-5 weeks for delivery.

8 9 10 11 12

The Editor's Feedback:

Computer: ☐ PET ☐ ATARI ☐ APPLE ☐ VIC-20 ☐ OSI ☐ OTHER ________

Are you a **COMPUTE!** Subscriber? ☐ Yes ☐ No I would like to see:

☐ More ☐ Fewer Specific applications programs
☐ More ☐ Fewer BASIC programs.
☐ More ☐ Fewer Machine language programs.
☐ More ☐ Fewer Tutorials.
☐ More ☐ Fewer Educational articles.
☐ More ☐ Fewer Detailed explanations of programs.
☐ More ☐ Fewer Games.
☐ More ☐ Fewer Reviews of game software.
☐ More ☐ Fewer Reviews of business software.
☐ More ☐ Fewer Reviews of educational software.
☐ More ☐ Fewer Reviews of hardware.

What do you like best about **COMPUTE!**?

What do you like least?

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 HOLMES, PA

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE! Magazine
P.O. Box 636
Holmes, PA 19043

Place
Stamp
Here

COMPUTE! Magazine
Post Office Box 5406
Greensboro, NC 27403

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 2312 GREENSBORO, NC

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE! Books
Post Office Box 5406
Greensboro, NC 27403

The Library of Computer and Information Sciences

Please accept my application for trial membership and send me the ENCYCLOPEDIA OF COMPUTER SCIENCE (44900-3) billing me only $2.95. I agree to purchase at least three additional Selections or Alternates over the next 12 months. Savings range up to 30% and occasionally even more. My membership is cancelable any time after I buy these three books. A shipping and handling charge is added to all shipments.

No-Risk Guarantee: If you are not satisfied—for any reason—you may return the Encyclopedia of Computer Science within 10 days and your membership will be canceled and you will owe nothing.

Name ______________________________

Name of firm ______________________________
(if you want subscription sent to your office)

Address ______________________ Apt. ____

City ______________________________

State ______________ Zip ______________
(Offer good in Continental U.S. and Canada only. Prices slightly higher in Canada.)

Compute 8/82 7-BL4

The Library of Computer and Information Sciences

Please accept my application for trial membership and send me the ENCYCLOPEDIA OF COMPUTER SCIENCE (44900-3) billing me only $2.95. I agree to purchase at least three additional Selections or Alternates over the next 12 months. Savings range up to 30% and occasionally even more. My membership is cancelable any time after I buy these three books. A shipping and handling charge is added to all shipments.

No-Risk Guarantee: If you are not satisfied—for any reason—you may return the Encyclopedia of Computer Science within 10 days and your membership will be canceled and you will owe nothing.

Name ______________________________

Name of firm ______________________________
(if you want subscription sent to your office)

Address ______________________ Apt. ____

City ______________________________

State ______________ Zip ______________
(Offer good in Continental U.S. and Canada only. Prices slightly higher in Canada.)

Compute 8/82 7-BK7

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD
FIRST CLASS PERMIT NO. 230 RIVERSIDE, N J

POSTAGE WILL BE PAID BY ADDRESSEE

The Library of Computer and Information Sciences
Riverside, New Jersey 08075