NO. 64 SEPTEMBER 1983 U.S. Edition: $2.50 International Edition: $3.00

Special Issue For Educators!

MICRO

Advancing Computer Knowledge

Education

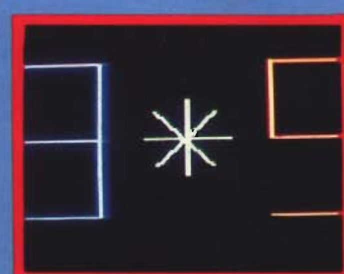
- Design your own educational software
- Elementary students use Logo
- Establish an effective computer curriculum in your school system
- Turtle Graphics for the VIC-20 and C64

Making More Than Money in the Silicon Valley

A Personal Look at a Personal Computer

See page 28

See page 60



See page 37

Atari Painting Program Wraps Up

A Product Catalog for the Atari and Apple

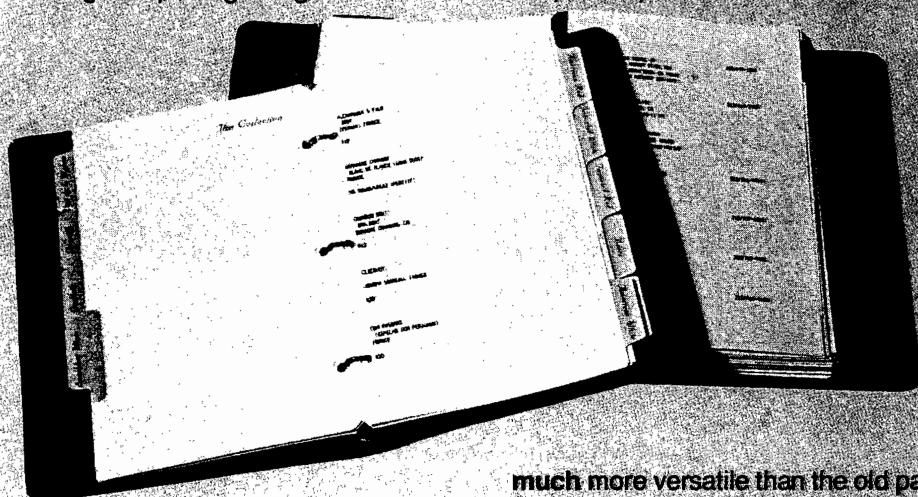
Text Compression and Encryption

MAGIC MEMORY

Will Remember Everything For You

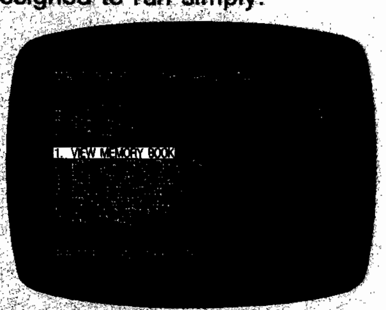
(If you want it to)

Imagine a system that would record all the wonderous, valuable information you have assimilated onto a single tiny disk. (No more scattered bits of paper, business cards, etc.) Imagine the same system giving you a typed sheet you could put into a notebook or print out for a party and instantly change, or add to, at a moments notice. Imagine cross-referencing to suit both your business needs and personal desires so that all your data was organized into one little black book! On top of all this — imagine having **fun** putting it together.

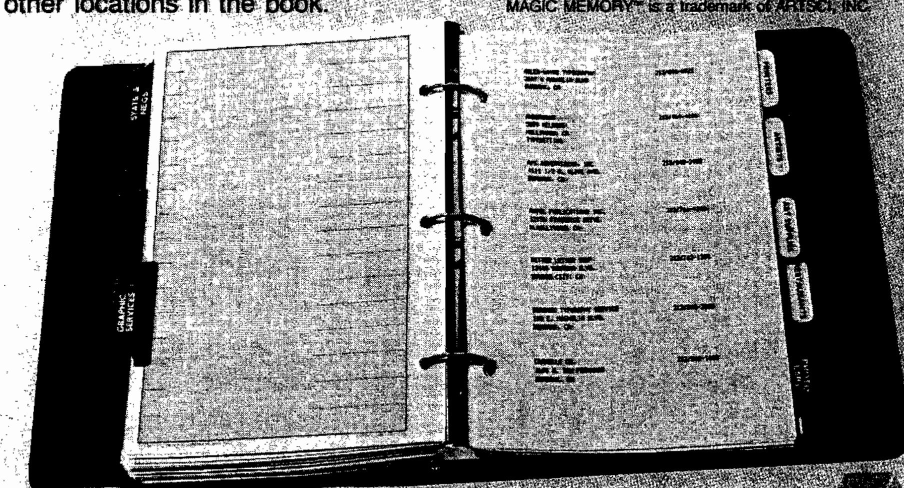


"Imagine" no more. ARTSCI, INC. proudly announces **MAGIC MEMORY™**, the complete advanced database system. **MAGIC MEMORY** is made to simplify the information storage process.

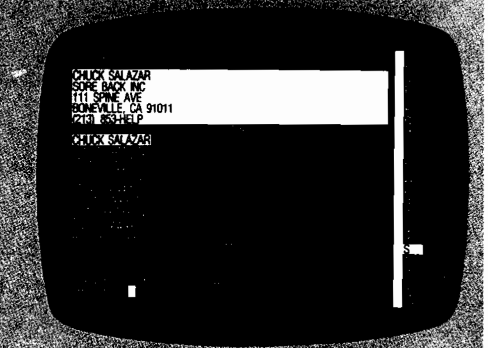
The typical first-time computer user has no *human* experience that will help him to relate to a *computer's* methods of handling information. Therefore, his learning ability is hampered and remains so for many hours of use, UNLESS the computer has been designed to run simply.



MAGIC MEMORY™ is built for the computer rookie. Everyone can relate to **MAGIC MEMORY™** because its form is familiar. It looks like an address book but its not. Its *more*. Like the address book, **MAGIC MEMORY™** presents an A thru Z index tabulation on the right edge of the video display. The user simply selects a tab and the *book* is opened to the proper page(s). A second set of tabs are available that can be labeled by the user (i.e.: companies one deals with, birthdays, lists, wines, shops, etc.) Yet **MAGIC MEMORY™** is **much** more versatile than the old paper address book. This system can instantly add or delete information, sort alphabetically, and transfer data to other locations in the book.



One of the features in the print section of **MAGIC MEMORY™** is the ability to print out any or all of the *address* book, or any of its special sections.



MAGIC MEMORY'S data presentation screen will instantly adapt to any type of information you may want to store. It is only limited to your imagination and your needs.

MAGIC MEMORY™ is designed to operate on an APPLE //e and still remain totally compatible with APPLE II. The system will operate in 40 columns or 80 columns. You may also use the 70-column display that requires no additional hardware.

MAGIC MEMORY™ retails for $99.95. Call 213/985-5763. Use your imagination to outwit your organizational woes.

MAGIC MEMORY™ is a trademark of ARTSCI, INC.

213/985-5763
5547 Satsuma Avenue
North Hollywood, Ca 91601

artsci™

MICRO™ Highlights

Education is the basis of all that is learned. It is the vehicle by which we gain the skills and knowledge that we use to exist. The quality of that education determines, to a great extent, the quality of our lives. Today we face sweeping changes in the methods of instruction. The reason for these changes? The microcomputer. The September issue of MICRO takes a look at what is happening in the field of computer education.

Dave Malmberg's "Turtle Graphics for the VIC-20 and C64" (pg. 28) was originally designed to teach his own youngsters the basics of computer programming. The task becomes fun and easy using the "turtle," which can be moved like a paint brush, leaving behind colorful pictures on the screen.

"Making More than Money in the Silicon Valley" (pg. 32) is a report on a business that is not "just another software company." Marjorie Morse discusses her interview with Nathan Schulhof, president and founder of Silicon Valley Systems.

Many school systems are using LOGO programs in the elementary grades. Phil Daley provides a brief synopsis of the Hillsboro-Deering, NH, school program and presents some samples. See "Logo in the Schools" (pg. 34).

"The Silicon Blackboard" by Emmalyn H. Bentley (pg. 39) is a comprehensive discussion of the use of computers in education. Examples of how different kinds of schools are instituting computer cirriculums in their systems are presented.

Marian Lorenze and Allan Moose describe the scope of applications for educational software and the various factors involved in designing such programs. A typical program is presented in "Writing Instructional Software" (pg. 44).

And finally, Dan Weston has a program that places text on the hi-res screen without using turtle graphics. See "Hi-Res Characters for Logo" (pg. 50).

MICRO completes the education feature with an "Educational Resource List" (pg. 54). Here you will find a compilation of educational software manufacturers.

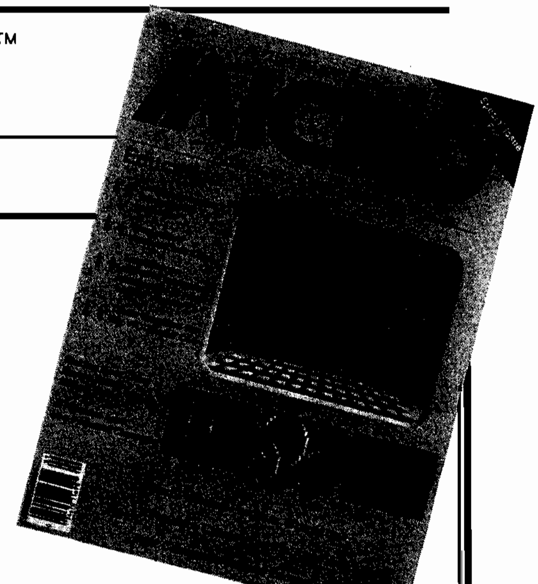
To round out your learning experience this month, MICRO includes "A Personal Look at a Personal Computer" by Richard Vile (pg. 66), "Using Signed Arithmetic on the 6502" by Randall Hyde (pg. 72), "Machine-Language Input Routines for Commodore Computers" by Thomas Henry (pg. 88), and "Text Compression and Encryption" by Walter Luke (pg. 92).

Also included are "Using VIC and C64 ROM Routines from Basic" by Terry Peterson (pg. 96), "Swap RAM or EPROM for Your ROM" by Ralph Tenny (pg. 100), "Displaying PET's Keyboard Matrix" by Werner Kolbe (pg. 104), and "Signed Binary Multiplication with the MC 6809" by T. J. Wagner and G. J. Liponski (pg. 111).

Don't miss our on-going columns: PET Vet (now known as Commodore Compass), From Here to Atari, CoCo Bits, and Interface Clinic; and part 3 of Paul Swanson's "Mode 10 Atari Painting Program" (pg. 58). Paul rounds out his program by adding convenient line, circle, and rectangle commands.

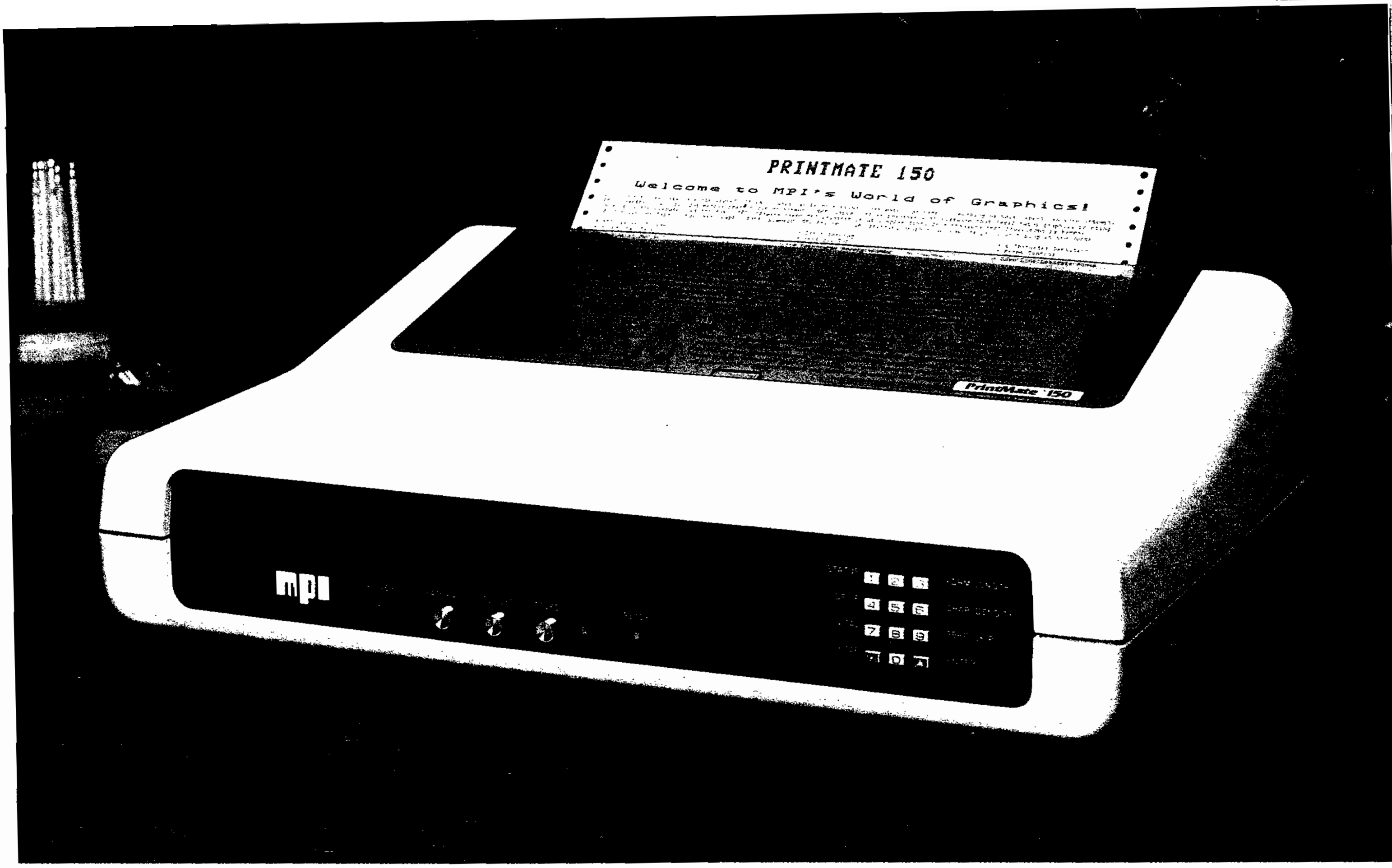
To ensure that your on-going education is a rewarding experience, read MICRO!

MICRO



ABOUT THE COVER

The colorful graphic on MICRO's cover is an interpretive representation of this month's feature — Education — as conceived by artist Curt Witt.



PrintMate™ 150

THE MOST ADVANCED PRINTER IN ITS CLASS.

Easy-to-use. Fast. Excellent print quality.

These are the features every pc and microsystem user wants. These are the same features we build into every PrintMate 150. And there's one more thing... value. No other printer offers more versatility for the price. At $995, the wide-carriage PrintMate 150 is an exceptional value.

HIGH SYSTEM THRUPUT
Print at 150 cps. Advanced logic-seeking impact printing. Accelerated print head slew rate. High-speed paper advance.

PRINT CAPABILITIES
10, 12, 15 or 17 characters per inch. High-density, high-speed correspondence printing, plus double-wide for emphasis.

SoftSwitch™ KEYPAD
Establish, change or display the operating mode from the front of the printer.

EXPANDABLE PRINT BUFFER
Increase performance with buffers that take you from 2K to 68K to provide high-speed interleaved printing with computing.

EASY WITH PAPER
Three paper paths—front, back or bottom—make the PrintMate 150 one very easy-to-use machine.

DOWNLINE LOADABLE FONTS
Custom character sets may be downloaded to a PrintMate 150 with a 4K or larger buffer.

PRINT WITH STYLE
No other printer offers more graphics support for the price. It's true. Our exclusive AP-PAK™, available for most popular computers, lets you print in dozens of stylized fonts, in characters up to ⅝" high. Got a graph on the screen? Need a custom font or logo? Do it with an AP-PAK.

OUR BOTTOM LINE
Superior performance starts at $995 on a PrintMate 150. We wouldn't print that if we didn't mean it. Call or write us today.
1-800-821-8848

OEM's: Ask us about custom AP-PAKs and low-cost private labeling.

The Printer People
mpi

Micro Peripherals, Inc.
4426 South Century Drive
Salt Lake City, UT 84107

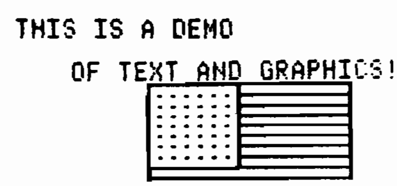
MICRO™

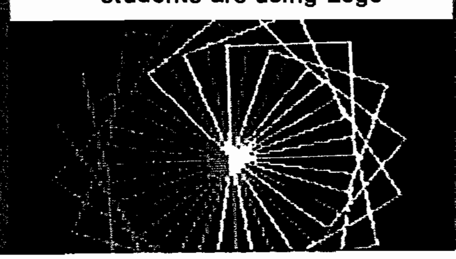
Advancing Computer Knowledge

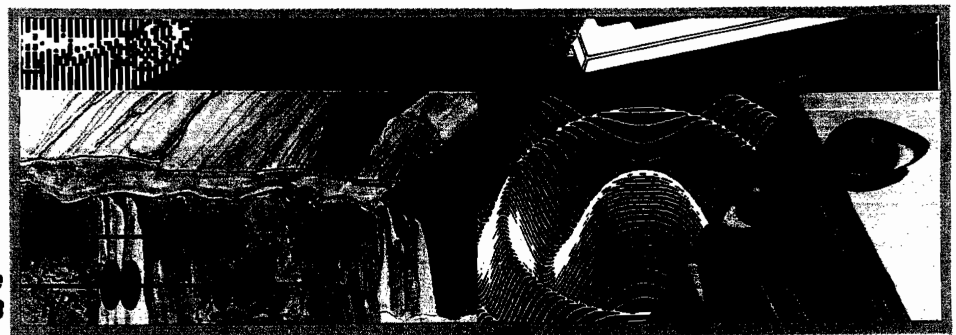
MICRO
P.O. Box 6502,
Amherst, NH 03031
(603) 889-4330

MICRO is published monthly by: MICRO, Amherst, NH 03031. Second Class postage paid at: Amherst, NH 03031 and additional mailing offices. USPS Publication Number: 483470. ISSN: 0271-9002. Send subscriptions, change of address, USPS Form 3579, requests for back issues and all other fulfillment questions to MICRO, 10 Northern Blvd., P.O. Box 6502, Amherst, NH 03031, or call (603) 889-4330, Telex: 955329 TLX SRVC; 800-227-1617. *Subscription rates (per year):* U.S. $24.00, $42.00 / 2 yr. Foreign surface mail $27.00. Air mail: Europe $42.00; Mexico, Central America, Middle East, North Africa, Central Africa $48.00; South America, South Africa, Far East, Australasia, New Zealand $72.00.

Education

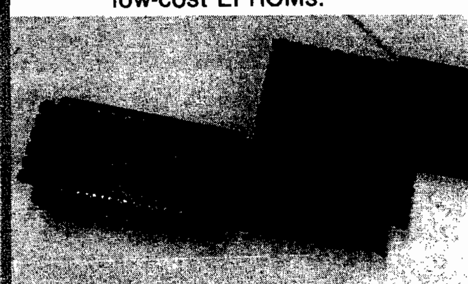






Education Feature begins page 28

Articles



$M + a_7*2^8 = a_7*2^7 + a_6*2^6 + ... + a_0*2^0$

Columns

DEPARTMENTS

MICRO™

Editorial

Who's Minding Computer Education?

Our education feature this month particularly emphasizes the methods used to teach grade-school students how to use and understand computers. We'd like to pose a question to our readers regarding computer education in classrooms. But first a little discussion.

Elementary students today *must* learn how to use computers. Of course not all students will work with computers as a profession, but it is a sure bet that they will need to know how to operate systems for word processing, database management, inventory control, etc. The point is, if children do not learn how to use computers in grade school, they will be at a disadvantage later in life.

Many educators do see the handwriting on the wall and have implemented extensive computer curriculums in their schools. Other educators know they should offer computer training but don't know how to go about it. So they buy a few microcomputers and put them in the classroom with teachers who have little or no computer training; the systems sit unused unless someone knows how to load PACMAN. And there are still schools with no evidence of computers at all.

Who is responsible for making sure effective computer curriculums are established in the schools?...Teachers? Parents? Students? School Administrators? Or is any one group the answer? These problems are addressed in depth in Emmalyn H. Bentley's article on page 39 of this issue. Emmalyn contends that, although the group actually developing the computer curriculum should be familiar with both hardware and software available (the teachers or school adminstration?), the parents can provide the major inspiring force to get a program started.

After talking with several teachers working in schools where computer programs have been established, it seems that the origins of these programs follow a definite trend. First, a teacher with some computer knowledge decides that it would be a good idea to bring computers into the classroom. Then this teacher trys to convince the principal and/or school board that a computer program would be worth the expense. Finally a budget is allocated and equipment is ordered. Generally the teacher responsible for the idea is responsible for setting up and maintaining the curriculum. On rarer occasions a budget is established and a teacher with programming expertise is hired specifically for the job.

But what about the schools lacking teachers or board members with computer knowledge? This is the point at which parents can begin participating. Any parent concerned about the education of his/her children should be concerned with establishing a computer program in the school. Probably many of the schools not yet hopping on the computer bandwagon would take note and jump aboard if enough parents showed interest.

If you are tentative about approaching your child's school regarding a computer program, it would probably be best if you did a little research first. Read education features in magazines like MICRO, and contact schools that you know already have a computer program. Present facts and suggestions (and perhaps financial estimates) and you probably will have greater success. Once again, Emmalyn's article on page 39 provides more details and suggestions for you on this topic.

Today's children need to know how to operate computers. Simple programming skills should be second nature to them by high-school graduation. Parents can provide the inspiration and motivation necessary to intiate these programs.

Enter Our Graphics Contest!!!

We're sponsoring an exciting contest for those of you interested in designing graphics pictures. You could win one of many prizes — big and small! Just use your favorite graphics program on your favorite microcomputer (either a Commodore, Apple, Atari, or Color Computer) and create! Turn to page 134 for all the details!

Marjorie Morse

Marjorie Morse
Managing Editor

Updates and Microbes

Reverse Instructions

In Randall Hyde's article, "Parameter Passing in Assembly Language, Part 2" (61:94), Mr. Hyde gave examples for several processors of a string printing routine which used the return address on the stack to point to the string to be printed. In the 6809 routine, the return address is stored on the stack high byte first, followed by the low byte (which is the reverse of the order on the 6502 stack). As shown in the original article, for the 6809 example, the high byte pointing to the target string is incremented before the low byte. This will of course produce an incorrect address. The remedy is to reverse the order of the instructions as follows:

```
PRTLOOP   LDA [3,S]      6809 VERSION
          BEQ ALLDONE
          JSR PUTC       CHAR OUTPUT ROUTINE
          INC 4,S        INCREMENT LOW BYTE
          BNE PRTLOOP
          INC 3,S        INCREMENT HIGH BYTE
                         ON PAGE CROSSING
          BRA PRTLOOP
ALLDONE   INC 4,S        INCREMENT LOW BYTE
                         TO TRUE RTN ADDRESS
          BNE RTN
          INC 3,S        INCREMENT HIGH BYTE
                         ON PAGE CROSSING
RTN       PULS A,X
          RTS
```

Randolph D. Glickman
San Antonio, TX 78216

Print Control Bug

A large number of MICRO readers have written to me expressing interest in my Print Control routine published in MICRO (58:29). One of these readers, Richard C. Greig of Radian Technology, wrote to inform me of a bug when the routine is used to output program-listings. This bug will occur only rarely, but it can be a source of errors. The problem is this: Both PRNTCTRL and Applesoft use the internal Applesoft subroutine STROUT (STRing OUT) to print lines of text. Normally, this produces no problem. However, if, during a program listing, say, PRNTCTRL detects that the right margin has been reached *and*, on generating the carriage return, detects that an end-of-page has occurred, the printing of the title (and page number), via STROUT, at the top of the next page will displace the contents of the STROUT pointers and prevent the printing of the remainder of the line currently being listed.

See listing 1 for a patch to this routine. With the exception of a slight change in the locations of the title pointers (to $3A7, $3A8), the "patched" routine may be used as described in the original MICRO article.

John R. Vokey
Alberta, Canada

Listing 1

```
10  REM ***************************
20  REM *                         *
30  REM *      PRINT CONTROL      *
40  REM *          PATCH          *
50  REM *           BY            *
60  REM *      JOHN R. VOKEY      *
70  REM *            &            *
80  REM *       H. CEM KANER      *
90  REM *          1983           *
95  REM *                         *
97  REM ***************************
100 :
110  REM TO USE THIS PATCH, BLOAD YOUR
120  REM OLD VERSION OF PRINT CONTROL
130  REM AT $300 (768) AND THEN RUN
140  REM THIS PROGRAM.  AFTER INSTALLING
150  REM THE PATCH, BSAVE PRNTCTRL,A$300,L$D0
160 :
200 HEX$ = "03A2:98 48 A0 00 B9 C9 03 F0 06 20 5C DB C8 D0 F
5 AD C4 03 EE C4 03 85 44 20 42 AE 68 A8 4C 65 03 N D823G"
210  FOR I = 1 TO  LEN (HEX$): POKE 511 + I, ASC ( MID$ (HEX
$,I,1)) + 128: NEXT I: POKE 72,0
220  CALL  - 144: REM INSTALL PATCH
230  PRINT : PRINT "PATCH IS INSTALLED": END
```

(Continued)

Update

Thank you for publishing our article "Color Disk BASIC: Observations and Utilites" by Michael Dudgeon and W. C. Clements, Jr. (61:34). I would like to provide an additonal fact about the format of a machine-language program on disk that has come to our attention since the article was written. The end-of-program marker in table 4 is #$FF. It turns out that this is a flag to signal not only the end of the file, but it also informs BASIC that this is the last program segment contained in the sector. You can store as many machine-language programs as you can fit into a sector, each having its own different start address, if the end-marker in between them is 0. LOAD will read the sector, placing every program in the sector into memory at the location specified by its own start address. Thus a machine-language file can contain an arbitrary number of programs per sector, and each will be loaded as if it were there alone. The DISKLOOK program can be used to put those programs into a sector, since SAVEM will only store one program in a sector. These facts can be used to greatly compress storage of short machine-language programs, resulting in less of the waste of partial sectors that is a major inefficiency in the program-storage algorithms in Color Disk BASIC.

William C. Clements, Jr.
University, AL

Color Disk BASIC Revision

"Color Disk BASIC: Observations and Utilities" by Michael Dudgeon and W. C. Clements, Jr." (61:34), contained an error in text. The paragraph immediately before the heading "How Program Files Are Stored" on page 34 should begin as follows: "When a file is killed the first byte contained in the file name string is set to *zero* and the entries in the allocation table that correspond to the granules containing the file are set to $FF."

The Authors
University, AL

CoCo Correction

In John Steiner's June CoCo Bits (61:18), listing 10 should read as follows:

```
10 CLS: P = PEEK(487)*256 +
PEEK(488)....
(remaining characters as is).
```

Atari Mode 10 Painter Problems

The listing that appeared with Paul Swanson's program (62:66) contained several errors. Please turn to page 60 of this issue for a complete, correct listing.

Letterbox

Dear Editor:

We need help! The problem: my son, 13 years old, has his heart set on a computer camp this summer. No problem you say? Well we are in Germany and can't seem to find a list of camps that might be suitable. Neither the U.S. Consulate General here nor the U.S. Embassy could provide a lead to such camps — preferred location somewhere in Michigan.

Can anyone help? It would be greatly appreciated. We have an Apple Euro Plus sitting right here at home, so he's not an absolute beginner. (Quite a feat, incidentally, in Germany, where computers are still treated as beastly job killers and most people back away from them. Personal computers are just now beginning to make some headway).

Edelgard Simon
Hochallee 23
2000 Hamburg 13
Germany

SHUTTER Results

Dear Editor:

By coincidence, I was working on a camera shutter speed meter program for my VIC-20 when Mike Dougherty's

(Continued)

program appeared in the January (56:45) issue of MICRO. I would like to point out an apparent error in the way he interprets his final results and suggest an addition to this otherwise sound program that might be of more use to photographers.

The statement in his third to the last paragraph relating relative error to the expected time is misleading in that 1/3 f/stop errors do not correspond to plus or minus 33% of the expected time as measured. Modern shutter speeds are geometric in progression and not linear so that each marked speed is twice as fast as the previous marked speed starting at the slowest speed. Therefore, a plus 100% error in the measured time is equivalent to a 1 f/stop overexposure but a minus 50% measured error is also a 1 f/stop error on the underexposure side. Clearly plus and minus 1/3 f/stops can't be equal to plus and minus 33% of the expected time. The following formulas will correct this problem for photographers who may wish to modify this program to display the final results as equivalent f/stop corrections instead of simply percent error:

$FS = LOG(1 + (AVE/100))/LOG(2)$ and
$AVE = (2^{FS} - 1)*100$

where FS is the change in f/stops (plus or minus) and AVE is the average measured error in percent. Substituting into these formulas shows that plus and minus 1/3 f/stop error tolerances are photographically equivalent to +26% (over exposure) and −26.6% (under exposure) respectively based on the measured time.

Notice that these formulas use a sign convention that is opposite to those used by Mr. Dougherty. I use plus to indicate overexposure for both f/stops and percent error, whereas Mr. Dougherty uses minus to indicate overexposure to show that his lens must be stopped down this amount (less light) to compensate for the slower shutter speed (longer time). To avoid any sign confusion, I would suggest that a PRINT statement be added to the program based on the relative error calculation to prompt the user as to whether the shutter is running "slow" or "fast" or any other PRINTed message the user finds convenient.

This program also uses a spacing of 24.64 mm for the spacing between the curtain velocity sensors. Many of the newest 35 mm cameras have vertical traveling shutters (such as the Copal design) so the spacing would be larger than the 24 mm vertical film opening. Anyone building the sensor array should use a spacing of less than 24 mm so that it can be used with both horizontal and vertical traveling shutters.

Rick Replogle
RD #1 Box 455
New Enterprise, PA 16664

Mike Dougherty Replies

I would like to thank Mr. Replogle for pointing out an obvious error in my interpretation of the SHUTTER results. As noted in the letter, giving the correction factors in terms of f/stops is more useful than the traditional relative error. Given an expected shutter time, E, and a measured shutter time, M, the following computes the f/stop ratio, FS:

$$FS = LOG_2(E/M) = LOG_{10}(E/M)/LOG_{10}(2)$$

This f/stop ratio may be determined relative to the expected shutter time or measured time since:

$$LOG_2(E/M) = -LOG_2(M/E)$$

The specific ratio chosen will depend on whether the result is used as an f/stop difference from the expected value or as a correction factor from the measured value. In SHUTTER, I was concerned with computing a correction factor to the camera's setting.

Mr. Replogle's formula for FS may be derived from my FS formula as follows. Let the percent relative error, AVE, be defined according to Mr. Replogle's sign preference and formula:

$$AVE = ((M - E)/E)*100$$

Then using the f/stop ratio for the difference from the expected shutter time:

$$\begin{aligned} FS &= LOG_2(M/E) \\ &= LOG_2(1 - 1 + M/E) \\ &= LOG_2(1 + M/E - 1) \\ &= LOG_2(1 + M/E - E/E) \\ &= LOG_2(1 + (M - E)/E) \\ &= LOG_2(1 + (((M - E)/E)*100)/100) \\ &= LOG_2(1 + AVE/100) \\ FS &= LOG_{10}(1 + AVE/100)/LOG_{10}(2) \end{aligned}$$

Note that this definition of FS uses a sign convention opposite to SHUTTER'S sign convention.

The f/stop correction may be added to SHUTTER with the following line:

```
2185 PRINT "Shutter f/stop error:
     ";LOG(EXPECT/AVE)/LOG(2)
```

where the BASIC LOG function is the logarithm to the base 10. This gives the worst case 1/500th second shutter setting an f/stop correction of:

$$FS = LOG_{10}(2/2.268)/LOG_{10}(2)$$
$$FS = -0.181$$

An FS of −0.181 indicates that I should stop down an extra 18% to correct for my 1/500th shutter speed.

Mike Dougherty
7659 West Fremont Ave.
Littleton, Co 80123

MICRO

MICRO™

Commodore Compass

by Loren Wright

A New Column?

No, this is the same column that has run in MICRO for three years as 'PET Vet.' Only the name has been changed to indicate that we aren't talking about just PETs any more. In fact, if you look back over the last several months, you'll find that this column has contained very little about the PET. I haven't forgotten the PET—at least not yet. Commodore isn't pushing it (not that they ever did very much in the U.S.), and they may even have stopped making them. The VIC has already outsold the PET, and the Commodore 64 will soon do so.

The PET is better than either the C64 or the VIC in many ways. In fact, I wish I had a PET here at home for this word-processing task. I'm getting eyestrain staring at the TV. I also find the PET easier to program; after all, it has a built-in machine-language monitor and a numeric keypad. And you don't have to shift (as I am about to do) for !"#$%&'(). The fact remains, though, that I, like many hundreds of thousands of people, have chosen to buy a Commodore 64 to use at home.

The handwriting is on the wall: the PET is too expensive (not only to buy, but also for Commodore to produce and ship) for what it does and will soon disappear from the market. In spite of this, there are a lot of PETs out there and few of you will give them up very quickly. MICRO and I will continue to support the PET, but keep in mind that most of the new software and hardware will be for the newer machines. Remember that the new machines have a lot in common with their ancestor, the PET.

Help for 1541 Users

Are you getting the most from your CBM 1541 disk drive? Probably not. The first thing you may have learned was how to 'NEW' a disk, followed by LOADing and SAVEing BASIC programs. Then you may have learned how to list the directory, and perhaps scratch and rename files. If you're used to the cassette (and even if you aren't), you probably found the commands complicated. There are several easier ways.

One way probably came on the disk you got with your 1541. On my disk there are three files: VIC-20 WEDGE, C-64 WEDGE, DOS 5.1. LOAD the appropriate wedge program and RUN it. This will load DOS 5.1 and display a message on the screen. Your machine now has about 300 fewer bytes, but all of the points you found awkward are much easier. Many of the disk commands require enclosing a command string in quotes after you open a file for the command chan-

nel. For instance, to NEW a disk you had to OPEN 1,8,15,"N0:DISKNAME,01". With the wedge, all you have to do is type @N0:DISKNAME,01 and press RETURN. (Historical note: The program is called the wedge because the original PET program used the '>' character instead of '@'. '>' still works, for old PET people who can't break the habit, but you have to use the shift key. Also, the program 'wedges' itself into the BASIC command interpreter.) Following is a summary of the wedge commands and some common examples.

@ or >	send command to disk unit
/	LOAD program

The '/' will also load a machine-language program at the address where it was originally stored. '@' by itself will read the error channel and print the error number, error message, track and sector on the screen. Without the wedge, you have to write a program. '@$' lists the disk directory to the screen without destroying your current BASIC program. Rename is '@R0:NEW NAME=OLDNAME'; validate is '@V0'; scratch is '@S0:FILENAME'. With all these commands, do not use quotes around filenames or command strings.

Now that I've convinced you that the wedge is a good thing, you should store the appropriate loader program as the first program on each of your disks. This is easy: just LOAD it from the system disk and SAVE it after you 'new' the new disk. VIC users are now all set. Commodore 64 users must also save the DOS Support file. It's easy if you have a machine-language monitor, such as 64MON, HESMON, or MICROMON. If you don't, use Terry Peterson's BSAVE program on page 96 of this issue. After you have LOADed and RUN the wedge program, use BSAVE or your monitor to save the DOS Support, entering 'CC00' as the starting address and 'D000' as the ending address.

Your next disk drive learning project can be the easy-to-use sequential files, or perhaps you could try the faster relative files. If so, get a hold of *Bennett's* "Mail List" program (an excellent program itself) and go through Jim Strasma's series "It's All Relative" that concluded last month. The Mail List program is in the public domain and available from the author. See the first installment (December, 1982) for details.

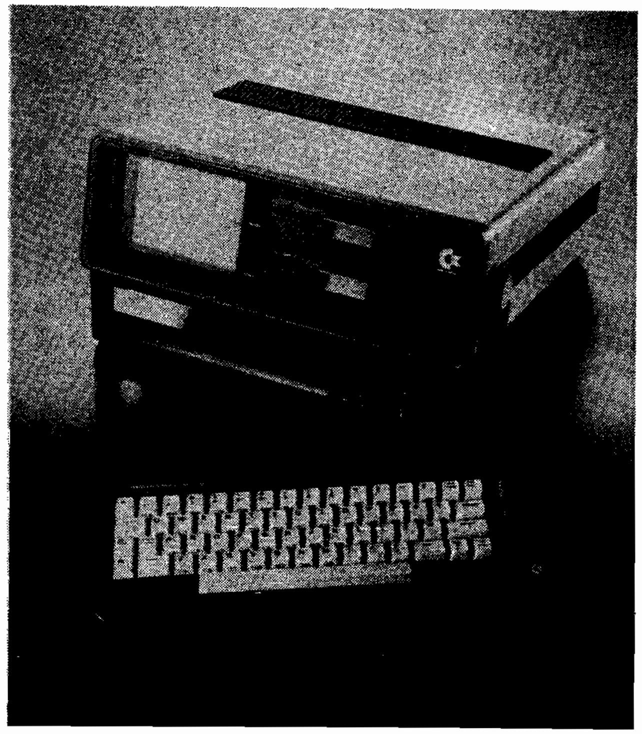
A little-known feature of the directory display is selective listing. Without the wedge, type 'LOAD "$2:MAST*"8'. With it, type '@$2: MAST*'. This will give you a selective display showing only the filenames beginning with 'MAST'. With dual drives, both drives will be checked. *(Continued on next page)*

Commodore Compass *(continued)*

Another useful program on the system disk is 'COPY-ALL' by Jim Butterfield. To use it, you need another 1541 addressed as unit 9 (see instructions in the manual or use the 'CHANGE UNIT' program on the system disk). This will copy files from one unit to the other, regardless of type and without loading them into memory. I have seen at least one commercial single-drive backup program. If you don't have access to a second 1541, you should consider getting such a backup program.

Does anyone have any idea how to modify a 1541 so that it behaves as drive 1 of unit 8? If this could be done, the BACKUP and COPY commands (which already exist) would be useful, and using programs such as disk-based assemblers and word processors would be a lot more convenient. Commodore has plans for a dual serial drive, but there is no telling how soon we'll see it.

Canadian Micro Distributors (500 Steeles Avenue, Milton, Ontario, L9T 3P7 Canada) is working on an accessory board called 'Turbo 1541', which will considerably speed the operation of the 1541. Rumored price is about $100. It will be a card that plugs inside the 1541, with a cable that connects to the VIC or C-64 port.



Commodore Business Machines Executive 64 System

Executive 64 Display at CES

Commodore displayed its attaché-style computer system at the Consumer Electronics Show in Chicago in early June. The system includes a Commodore 64-compatible computer (complete with sound, graphics, sprites, and 64K), two 5 1/4" disk drives and a 5" monitor. The system has a suggested retail price of $995.

MICRO

Design & Illustration: Paul Stephenson

ADVENTURE.
THE KEY IS YOUR COMPUTER

KONGO KONG

Climb ladders, avoid the barrels the crazy ape is rolling at you, and rescue the damsel. Fast machine code action.

Available for VIC-20 and COMMODORE 64

PLAYED WITH JOYSTICK OR KEYBOARD

$19.95

GRAVE ROBBERS

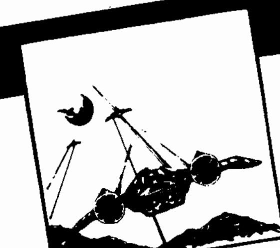
Introducing the first GRAPHIC ADVENTURE ever available on the VIC-20! With realistic audio-visual effects, you explore an old deserted graveyard and actually see the perils that lie beyond.

Available for VIC-20 and COMMODORE 64

PLAYED WITH KEYBOARD

$14.95

ANNIHILATOR



Defend your planet against the hostile aliens. All machine code makes this "Defender-like" program one of our best arcade games.

Available for VIC-20 and COMMODORE 64

PLAYED WITH JOYSTICK

$19.95

ADVENTURE PACK I

(3 programs)

Jack and the Beanstalk Defeat the Giant in your quest for his Golden Hen.

Computer Adventure Re-live the "excitement" of getting your computer. An adventure with a very different flavor.

Moon Base Alpha You must find a way to destroy the meteor that is racing towards your base, or else all moon colonies will be demolished!

Available for VIC-20 and COMMODORE 64

PLAYED WITH KEYBOARD

$19.95

ADVENTURE PACK II

(3 programs)

African Escape As the sole survivor of a plane crash, you must find your way out of the dark continent.

Hospital Adventure You are a spy whose mission is to complete the bungled assassination attempt on the evil dictator, who is recuperating in the hospital under heavy guard.

Bomb Threat Get back to town to warn the authorities of the bomb planted by the terrorists who left you prisoner at their hideout.

Available for VIC 20 and COMMODORE 64

PLAYED WITH KEYBOARD

$19.95

CHOMPER MAN

High speed machine action. Don't let the bullies catch you in a game packed full of machine code.

Available for COMMODORE 64

PLAYED WITH JOYSTICK OR KEYBOARD

$19.95

Programs for the
VIC-20 and the COMMODORE 64

All programs fit in standard memory. All programs are on cassette tape.

VICTORY SOFTWARE®

VICTORY SOFTWARE INC.
7 Valley Brook Road, Paoli, PA 19301
(215) 296-3787

Check your local dealer or order directly.
ORDERING We accept personal checks, money orders, VISA, and MASTER-CHARGE. Charge orders please include number and expiration date. Overseas orders please use charge or have check payable through a U.S. bank. Add $1.50 postage and handling per order. PA residents please add 6% sales tax.

VISA

master charge

MICRO™

From Here To Atari

Paul S. Swanson

Entering a long listing from a magazine is not an easy task. I have entered many programs from magazines and spent hours trying to get all of the mistakes out of them. Being a programmer helps because I can follow the logic in the program (usually) and find errors that way.

Letters I have received indicate that other people have the same problem. Often the person typing is not a programmer, which compounds the problem. Unfortunately, there is no simple solution.

There are a few routes to persue. If you get an error running a program you typed from a magazine, the line giving the error may or may not be in error. The error may be caused by a line the computer encountered earlier. Look for other lines in the program that use the same variable(s). Be careful of lower-case L (l) and the number one (1) and upper-case O and the number zero (0).

Another route to persue is to contact other Atari owners that may have entered the program. Even if the other person couldn't get it to work, comparing versions may correct the problems. Local Atari user groups are very good for this sort of collaboration.

If you have no modem on your Atari, find someone who does. A program like AMODEM (a public domain communications program) will allow an Atari with a modem to download programs from local bulletin board services, which are usually free. My last long program, which is the Mode 10 painter program (MICRO 62:66), was uploaded and is available on several bulletin board services in the Cambridge, MA, area, some of which were noted in the article (pg. 71). One of those bulletin boards does not exist anymore; the Cambridge AMIS board is no longer on line, so I uploaded the program to one called The Outpost at (617) 259-0181. MICRO has a bulletin board (still in the experimental stage) and programs in the magazine may be available in its download file. The number to call is (603) 883-1576. If all goes well, all of the listings in each issue of the magazine will spend a few months in MICRO's download file. You will need a modem on your Atari and a program that will allow downloading to disk files, or a friend with that set up.

Logo

The Atari version of Logo is now available on a cartridge. This version has turtle graphics using the equivalent of GRAPHICS 7 in BASIC, which is the same as the turtle graphics implementation in Pilot. Atari Logo sports up to four turtles. It uses the players for turtles and supplies alterable shape tables for them. I have heard one complaint about Atari Logo; it was that the turtles do not always point exactly in the direction they are heading. I have found this to be true. It looks like the turtles point in as many as ten directions but the heading can be defined much finer than that. Therefore, the turtle always points within 36 degrees of its true heading.

The turtle positions use a 320 × 240 coordinate grid, which means the turtle can be placed finer than the pixel count (which is 160 × 96) can display. All 128 colors are available, four of which may be assigned to the screen. Color indirection can be used by drawing then changing the color. Everything drawn will change to the new color, similar to the way the BASIC SETCOLOR statement works.

Atari Logo supports two sound channels to generate tones. A tone may be generated with controls on frequency, duration, and decay. If a tone is sounded on one of the channels and a second tone is requested for the same channel, the computer will wait for the first tone to finish before beginning the second tone.

Since I program mostly in BASIC and assembly language, I found myself a little handicapped when approaching Logo. There are no line numbers in Logo, which means the familiar GOTO, GOSUB, and IF...THEN with a line number doesn't exist. Instead, Logo is completely procedure-oriented, adding commands to the language by defining procedures. A procedure can also call itself, and this type of recursive programming opens up some interesting possibilities. I was unsuccessful at finding someone who could compare the Atari Logo to other versions on computers like the Apple. I will keep trying and, if successful, I will have such a comparison for next month.

Light Pens

If you built a light pen for your Atari using the instructions from "An Inexpensive Lightpen for the VIC-20, C-64, and Atari" by David Bryson (MICRO 61:82), then you were able to obtain a phototransister (which I haven't) or you are having some problems with precision. The phototransister Bryson uses has a 2-microsecond response time; the only ones I have been able to find have response times of around 8 microseconds, which really is not fast enough for drawing on the screen.

The timing is most important in determining the horizontal position of the pen. The scan line on the television is made up of color clocks, which are the width of a mode 7 dot (or mode 15 dot on the 1200XL). There are 160 color clocks across the normal screen (the width of the blue area in the text mode). Therefore, one color clock occurs in about .25 microseconds. In other words, the response time of the phototransister specified by David Bryson is within four color clocks horizontally, and the response time of the more common 8-microsecond phototransister is within 16 color clocks. To combat the

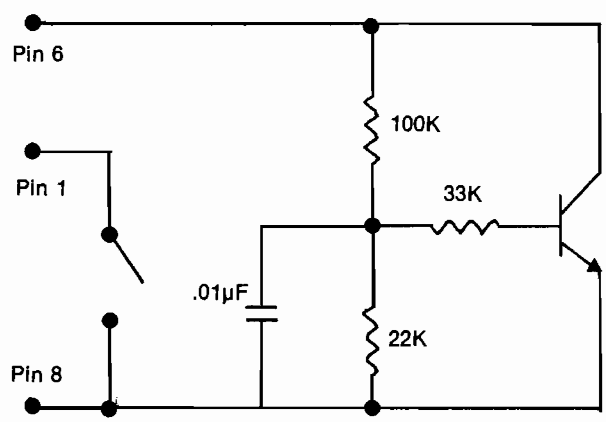
(Continued on page 18)

From Here to Atari *(continued)*

problem, I have developed a method for reading the pen. The method is not perfectly sound, but it usually works. Take multiple readings of the pen and use the lowest horizontal value after adjustments are made.

The vertical coordinate of the light pen is usually good the first time read. It denotes vertical position in units of two scan lines, so this number, obtained by a PEEK(565), will range from around 16 to around 111. The horizontal reading, which is a PEEK(564), has several complications. First, zero happens about two-thirds of the way across the screen. The left edge of the screen may be around 80 or 90. To reduce the horizontal reading, if the reading is below 40, add 228. Ignore readings, after this adjustment, that are below 80; these are false readings, indicating points that are not on the screen. Of the remaining readings, pick the lowest of 10 or 20 readings. This won't be perfect, but it will be fairly close. The differences in television sets and computer signals will make the readings of the left and right hand edges vary a little, but there will still be a count of 160 color clocks horrizontally.

In persuing the topic of light pens, I contacted General Electric and spoke to William Sahm, one of their application engineers. He gave me a circuit that helped enhance the response of a phototransister in this type of application. It required three resisters and a capacitor, all housed



Schematic for Light Pen with blessing circuit.
Keep all leads as short as possible to avoid external effects.

in the light pen itself. What it did was bias the transistor so that the range of light from the screen fell into the linear area of the transistor's response curve, making it much more reliable. The parts required for this are a .01 microfarad disk capacitor, 100K, 33K, and 22K resistors. Wire it according to the schematic below, keeping all leads as short as practical and all close to the phototransister. Be careful when soldering to the phototransister because these devices are very sensitive to heat.

Also, a better response may be obtained using twisted or ribbon cable (instead of the specified shielded cable) to run the signal from the light pen to the computer.

(continued)

From Here to Atari *(continued)*

Although shielded cable does cut down on the radio emissions from the wire, it also introduces a lot of unwanted capacitance. I used a ribbon cable with the ground and switch return on one side and the signal, which is connected to pin 6, on the other side of the ribbon. This also cuts down on the capacitance by keeping the signal away from the ground wires. With the biasing circuit in the pen, there is no longer any need for the 100K resistor between the signal line and the +5-volt line. The ribbon cable, if that is used, should be set up in the order indicated on the schematic. Another possibility is to use a four-conductor ribbon, ordering the wires as the light pen (connected to pin 6), an unconnected wire, the switch return (connected to pin 1), then the ground wire. Keep all leads, including the length of cable between the computer and the light pen, as short as is practical. This will cut down on the capacitance as well as the resistances from the wire. Since the signals are radio frequency signals, it also cuts down on the possibility of interference with radios in the area by making the antenna a little shorter. Three feet should be adequate for the length of the wire between the plug and the light pen.

You may contact Paul at 97 Jackson St., Cambridge, MA 02140.

MICRO™

MICRO™

CoCo Bits

John Steiner

The MC-10 Color Computer

Radio Shack's new MC-10 was introduced at the end of May. The new color computer uses a 6803 processor and has a sub-set of Extended Color BASIC. The language is more powerful than Color BASIC, yet not as powerful as Extended BASIC.

I have worked with an MC-10 and found its keyboard difficult to use. It does allow single key entry of BASIC keywords by using a control key, but unfortunately the control key is where the left shift key normally is, so all shifts must be done with the right hand. The MC-10 is capable of printing block graphics from the keyboard.

The computer has a serial I/O port for a printer or modem, and will make an inexpensive terminal. Curiously, line printer syntax is LPRINT rather than the standard CoCo PRINT # – 2. Bob Rosen of Spectrum Projects sent me a note with the following printer baud rate pokes.

BAUD	POKE
300	16932, 240
600	16932, 118
1200	16932, 60
2400	16932, 25
4800	16932, 10

There is no joystick input on the MC-10, and the expansion connector is a 34-pin connector opposite in polarity to the standard CoCo. Memory expansion will be available to 20K. Cassette I/O is at 1500 baud, however CoCo and MC-10 tapes are not compatible.

The Dragon 32

I recently received a letter from F. J. Philbrow of Cheshire, England. He has a Dragon 32, a computer similar to the Color Computer. Mr. Philbrow sent along a complete comparison chart of CoCo BASIC tokens and Dragon 32 tokens. Though there are many similarities, there are also many differences. If you would like the list, send me a stamped, self-addressed envelope. (See my address at the end of the column.) The Dragon will run much of the CoCo software, but there are distinct differences. For example, it uses two rows of 4116s instead of the 4164s CoCo uses.

Educational Software

The CoCo is being supported by several education software companies, in addition to the education software support from Radio Shack. While I was at Rainbowfest, I picked up an excellent educational software package from the Follett Library Book Company. The package is called MOPTOWN PARADE and has become a favorite of my four-year-old daughter.

Moptown provides an excellent example of how a computer can be used to teach the concepts of logical thinking. The program series is available on three cassettes or two disks and consists of eleven games. The simplest games are written for an age level of three to four years and teach the concepts of sameness and differentness. During the progression of the series, more abstract concepts are taught.

Another first rate package is Early Games For Young Children from Counterpoint Software. The nine games are attractive to my daughter, though they are not quite as interesting to her as Moptown Parade. Early Games covers numbers, addition, subtraction, and the alphabet. A drawing board is included for creating simple block graphic pictures. The unique picture menu makes it easy for young children to select a program they desire.

New CoCo Bulletin Board

Those of you with terminal software who want to contact me or send tidbits of CoCo info may do so via the Dakota Database. The system is up and running evenings, and by the time you read this should be available 24 hours daily. The bulletin board sports full upload and download capabilities, so you may upload a file and leave a private message to me on the E-mail system. I am the SYSOP.

For those who are curious, the system consists of a TDP-100 with 64K, two disk drives, a Hayes Smartmodem, and a Sanyo green screen monitor. Software to run the system was written by Silicon Rainbow Products. The data line can be reached by dialing 701-280-1928. I will pass along any other CoCo bulletin board numbers if you send them to me.

I also check into Compuserve, though only on a monthly basis. My user number is 73075,1735.

Color Mod for early TDPs

When I first got my TDP-100 I noticed a definite difference in high-resolution color graphics from my earlier CoCo. There is a problem in those early TDPs. Later model TDPs are coming out with a slight modification. Ron Krebs of Mark Data Products was kind enough to provide the correction and gave me permission to pass it along. You can tell if your machine has the modification already installed by looking near U9, the

(Continued on page 22)

CoCo Bits *(continued)*

video display generator and transistor Q3.

You will need to stop at an electronics supply store to obtain the following parts:

75 pf disk ceramic capacitor
33 Kohm ¼ watt resistor
27 microhenry choke JW Miller part no. 9230-54

You shouldn't have any problem finding the capacitor and resistor, but the choke might be a little more difficult. If an electronic parts store can't help, stop at a radio/tv repair shop; if they don't have it, they can probably get it.

Once you have the parts, wire them in series with the choke in the center. Put some insulated tubing around the assembly and solder the free resistor end to the emitter lead of transistor Q3. Connect the free capacitor lead to pin 33 of U9, the 6847 video display generator chip.

I easily installed the modification and have had much better looking color on high-resolution graphics displays.

Address of companies mentioned in this column:

Counterpoint Software, Inc.
Suite 140 Shelard Plaza North
Minneapolis, MN 55426

Follett Library Book Company
4506 Northwest Hwy (Rt 14 & 31)
Crystal Lake, IL 60014

Mark Data Products
24001 Alicia Parkway, No. 226
Mission Viejo, CA 92691

Silicon Rainbow Products
663 South Bernardo Ave Suite 225
Sunnyvale, CA 94087

Spectrum Projects
93-15 86th Drive
Woodhaven, NY 11421

You may contact Mr. Steiner at 508 Fourth Ave. NW, Riverside, ND 58078.

MICRO

MICRO™

Apple Slices

by Jules Gilder

Exciting things are happening in the Apple world. Apple has released a new version of DOS to software developers, more information on Mackintosh has surfaced, and Videx has just come out with a fantastic new display board that will let you get as many as 160 characters per line on a video display.

About eight months ago, rumors were rampant that Apple was planning to come out with a new DOS that would obsolete DOS 3.3. Cryptically referred to as XDOS, it was supposed to make it possible to transfer files between Apple IIs and IIIs. Well, it has finally been announced. Known officially as ProDOS, the new disk operating system has been released to software developers. You won't be able to get your hands on it, however, until the first quarter of 1984, when it will be released to the public.

Apple has not yet said how much the new DOS will cost, but the company has indicated that, unlike the DOS 3.3 upgrade, which required a hardware change in the form of new PROMs, the upgrade to ProDOS will not require any hardware changes (and here's the catch) to any Apple that has at least 64K of RAM. Is it just coincidence that Apple Language Card prices, which have been hovering around $100, have just jumped to $140?

Apple points out that ProDOS uses the same Unix-like hierarchical file structure, file-naming techniques, and data formats as SOS, the operating system used on the Apple III. Because of this, it will be possible to transfer data files from an Apple II to an Apple III and *vice versa.* A big plus for ProDOS is that it makes it possible to use files that are larger than 143K, which is the maximum amount of data that can be stored on a single DOS 3.3 floppy diskette. This ability to automatically span disk drives will make it possible to use programs that formerly were limited to systems with a hard disk drive.

It should be pointed out that while Apple says that ProDOS does not make DOS 3.3 obsolete, they are nevertheless encouraging software developers to use ProDOS instead of DOS 3.3 for new applications on the Apple II.

Apple II Prices to Drop

The much rumored Mackintosh computer, which an Apple spokesman says doesn't exist because it hasn't been officially announced, is scheduled to come out in the first quarter of 1984, according to reliable sources. The price of the machine is likely to be in the $1500 price range. This has been deduced from reports that Apple is making large quantities of the Mackintosh computer available to universities for about $1000 each. Rumors throughout Silicon Valley also peg the price of the Mackintosh near this value.

With the Mackintosh coming out at such a reasonable price (and it is for the computing power offered) the question that begs to be answered is, "What's going to happen to the Apple IIe?" The answer is, the price will probably go down significantly. With only a few dozen chips in the new IIe, manufacturing costs are substantially lower than they were for the old Apple II Plus. Therefore, it would not be surprising at all to see the Apple IIe drop to $600. And if Apple really wants to get aggressive and start competing with Commodore, which is currently selling a 64K computer for about $300, we might see the price of an Apple IIe drop even lower. If Apple does take on Commodore, we can expect to start seeing prices drop in October, in time for the Christmas buying season.

Mackintosh promises a lot of computing capability. Based on the Motorola 68000 microprocessor, the same one used in the Lisa computer, it is expected to come with 128K of RAM and a built-in, high-resolution video monitor. Industry sources indicate that the computer will be similar in many ways to the Lisa, sporting a mouse, multiple windows, and graphic icons, but will not be compatible with it. One indication of this incompatibility is that Lisa uses a specially designed double-sided 5 1/4-inch disk drive, while Mackintosh is expected to come with a built-in 3 1/2-inch micro floppy disk drive.

An Outstanding Video Display Board

What has to be the best video display board produced for the Apple II yet has just been introduced by Videx of Corvallis, OR. The company that brought thousands of Apple II owners 80-column capability has now doubled it and come out with a card that can give you as many as 160 characters per line. Dubbed UltraTerm, the new card features nine display modes and costs $379. These include the normal 40-column display, an 80-column by 24-row display — which emulates Videx's earlier Videoterm board, a 96 × 24 display, a 160 × 24 display, and five interlaced video display modes. The interlaced video mode results in a higher quality character in which the vertical elements of the character are more completely connected. They are impressive. Included in the interlaced mode are: 80 × 32, 80 × 48, 132 × 24, and 128 × 32 displays. Depending on what mode you are using, you can display as many as 4096 characters on the screen at one time.

In addition to increasing the number of characters you can display on a line, UltraTerm also gives a character-by-character selection of one of two sets of special character attributes that change the intensity of the display. Thus you can have normal and high-resolution characters displayed, or normal and inverse characters, or highlight and lowlight. The latter may be applied both to normal and inverse text.

All of UltraTerm's display modes are software selectable and the character set used for display has an 8 × 12 dot matrix. This character set includes the 96 printable ASCII characters. The lower-case characters in the set can even be entered from an unmodified keyboard by using the CTRL-A as a toggle between the two cases. In addition to the ASCII set, there is a 15-character line-graphics character set and a 7-character block-graphics font.

(Continued on page 26)

UltraTerm comes with full support for BASIC, Pascal, and C/PM. While there are not many programs available yet that take full advantage of the board, this can be expected to change rapidly. In the meantime, it will work in emulation mode with all Videoterm-oriented software. In addition, Videx will shortly make available a pre-boot program that will allow you to use UltraTerm and VisiCalc together to display a spreadsheet with 128 columns and 32 rows. They are also working on an Applewriter II pre-boot program. Those of you who use WordStar can start taking advantage of UltraTerm right away by simply reconfiguring your system with the INSTALL program that comes on the WordStar diskette.

There are certain caveats you should be aware of before you use UltraTerm. First, you'll have to remove all FLASH statements from any BASIC program that is going to be used with the board because these can have unpredictable results. Second, and more importantly, you have to have a good video monitor because not all of the display modes can be used with all monitors.

Two important monitor features that should be considered are persistence of the phosphor used on the display screen and video bandwidth, or resolution, of the display. UltraTerm requires a minimum bandwidth of 20 MHz to produce a sharp display in the 128-, 132-, or 160-character modes. The high-persistence phosphor is needed for the interlaced mode display, where characters are written to the screen only 30 times a second instead of 60. With a low-persistence phosphor, the display will flicker slightly. Videx recommends the Apple Monitor III for use with UltraTerm, although they point out that it cannot be used for the 160- or 96-character display modes. The Amdek 300A, however, will work well for all of UltraTerm's display modes.

Overall, this card is an excellent peripheral and we look forward to seeing more software adapted for use with it soon.

MICRO

You may contact Mr. Gilder at REDLIG Systems, Inc., 2068 79th St., Brooklyn, NY 11214.

Thinking about a 1200 Baud Phone Modem?
WAIT!
You need time.
And PRO-MODEM 1200 has it. A Real Time Clock/Calendar combined with an intelligent full 212A 300/1200 baud telephone modem. Plus the capability to expand into a full telecommunications system. Much more than just a phone modem.
When you're on-line, time is money. PRO-MODEM helps you save. By monitoring the duration and cost of your phone calls. By sending and receiving messages at preset times when the rates are lower. Unattended. With or without your computer.
Compare the $495 PRO-MODEM 1200 with any other modem on the market. For example, you'd have to buy both the Hayes Smartmodem 1200 plus their Chronograph for about $950 to get a modem with time base.
And PRO-MODEM 1200 does more. It lets you build a full telecommunications system with features like Rep Dialer, Incoming and Outgoing Message Buffering, Mailing List, Help Mode, Programmable Operating Instructions, a 12-Character Alpha-Numeric Time and Message Display, and easy to use PRO-COM Software.
There isn't space to describe it all here. See your local dealer for complete details now. It'll be well worth your time. And money.
Prometheus Products, Inc., 45277 Fremont Blvd., Fremont, CA 94538, (415) 490-2370
Pro-Modem 1200 from PROMETHEUS

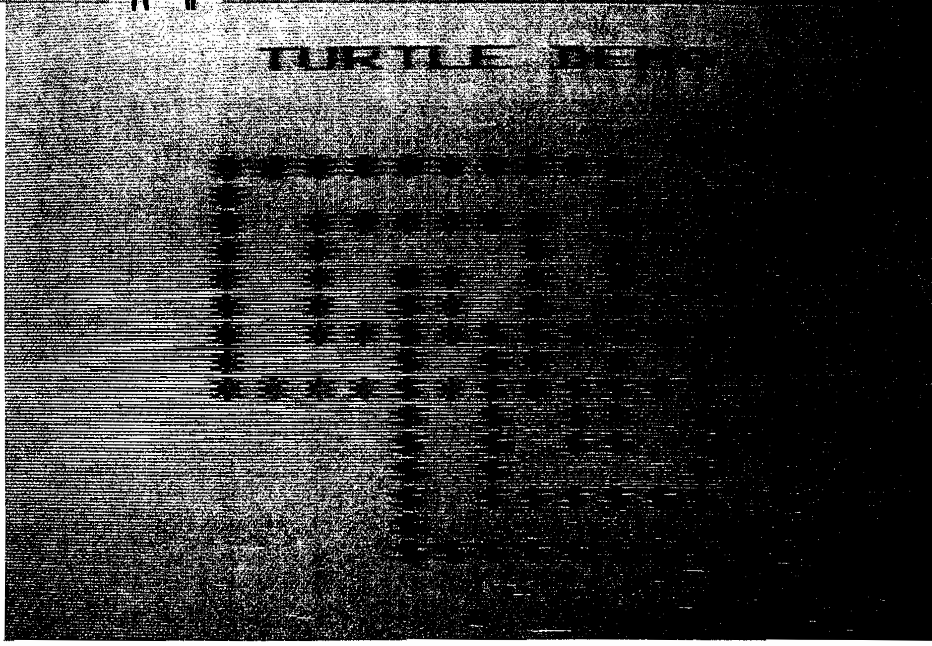


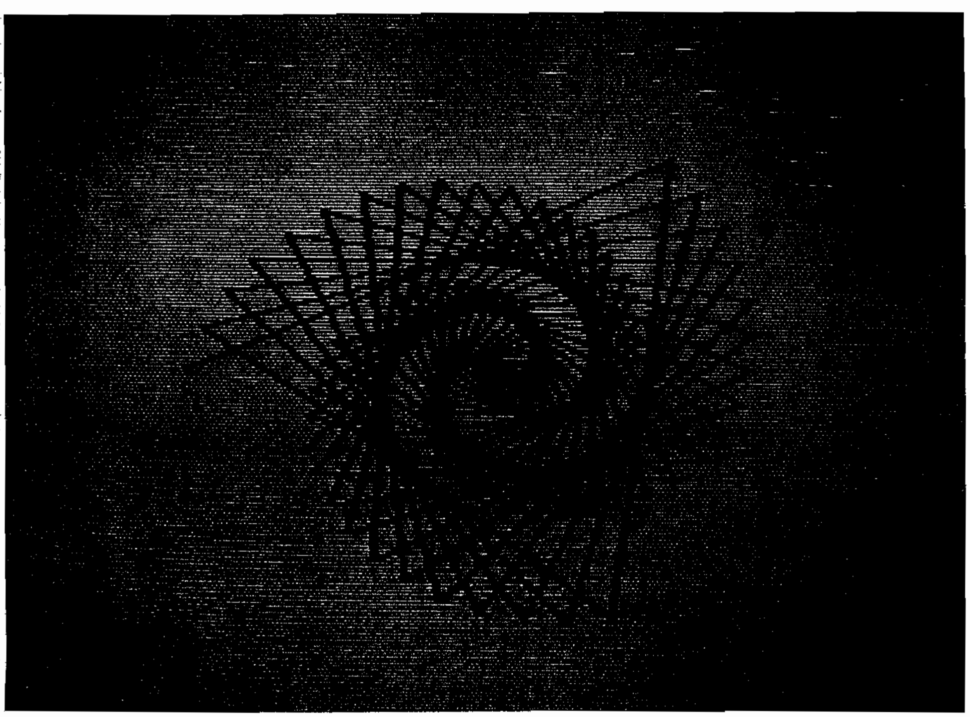
Figure 1: Two spirals formed of characters — VIC-20 Turtle Graphics.

Turtle Graphics FOR THE VIC-20 AND C 64

by David Malmberg

My TURTLE GRAPHICS language for the VIC-20 and Commodore 64 was originally designed to be an easy and fun way to teach my own two young children about computers and to introduce them to programming concepts. The basic idea of the language is to allow children (or a beginning programmer of any age) to give instructions to an imaginary Turtle that cause it to roam over the surface of the computer's display screen. As the Turtle moves, it can act like a paint brush and leave colorful pictures on the screen. As the computer novice becomes more adept at controlling the Turtle's artistic efforts, he or she is painlessly learning all of the basics of computer programming.

Figure 2: LOGO-like "Squiral" figure drawn with C64 Turtle II program.



TURTLE History and Philosophy

The original concepts of using the Turtle as a teaching tool were developed in the late 1960's by Seymour Papert of MIT's Artificial Intelligence Laboratory. Papert had been struggling to find an effective way to teach children about computers. He worked with Jean Piaget, the famous child psychologist, studying how children think and learn. Piaget convinced him that children learn best by self-discovery and by trial-and-error, and that the real challenge to educators is to provide both the environment and the tools to nourish this discovery process. Papert developed the Turtle Graphics capabilities of his LOGO language with this challenge in mind.[1]

Papert's early Turtle was a mechanical robot that could be programmed to move about the floor when given instructions such as FORWARD 30 and RIGHT 90. In time, this mechanical Turtle gave way to an electronic version — a cursor roaming over the surface of a video display unit, leaving colorful and artistic pictures in its wake. However, the philosophy of the Turtle as a programmable learning tool is still the same. By programming the Turtle and then watching the Turtle execute the program (through its actions), the child can experiment with ideas and get immediate feedback on whether or not the ideas work as expected. If not, the programmer can either try another approach or explore the mistake further. This ability to "debug" ideas and to gradually work towards a solution to a problem is the cornerstone of Turtle Graphics' implementation of the Piagetian view of learning.

In addition to Papert's LOGO language, Turtle Graphics capabilities have become a part of several computer languages including SMALLTALK and several versions of PASCAL and PILOT.

VIC-20 TURTLE GRAPHICS

TURTLE GRAPHICS for the VIC-20 comes in the form of a plug-in 8K ROM cartridge that takes control of the VIC when power is turned on. In place of the VIC's normal operating system and BASIC, the cartridge substitutes its own line editor, option menu, and the TURTLE GRAPHICS language. The TURTLE system is menu-driven for easy use and has an optional trace mode to help the beginning programmer follow the logic of a program one step at a time. The built-in editor allows easy insertion, deletion, and replacement of program lines. The editor also lets the programmer enter two-letter abbreviations for all commands; for example, CS in place of CLEAR SCREEN. However, for clarity these abbreviated commands are all expanded to their full English equivalents whenever a program is listed. Programs may be listed on a printer and saved on, or loaded from, tape or disk. The TURTLE cartridge is totally self-contained and will work with a standard 5K VIC. A 72-page manual with a full tutorial and numerous example programs is included with the cartridge.

The TURTLE GRAPHICS language has over thirty different commands including commands for color, sound, motion (both speed and direction), logical conditions, program branching, subroutines, and testing for a specific character in front of the Turtle on the screen. The words used for each of these commands were selected to be as clearly understood and obvious in meaning as possible. Using these commands the programmer can cause the Turtle to paint with characters, text, and graphic symbols in eight different colors. The range of tasks possible in TURTLE GRAPHICS extends from

(Continued on page 31)

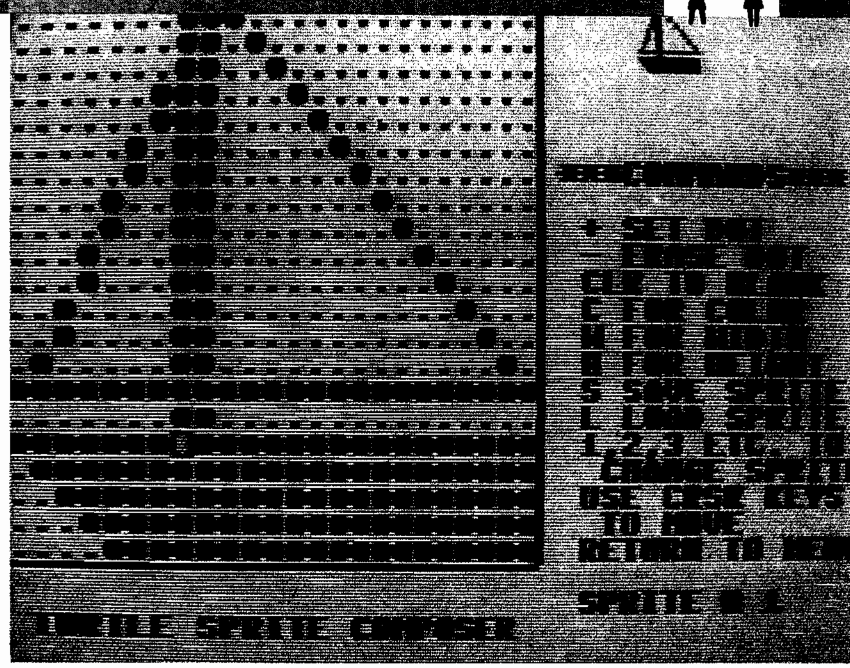
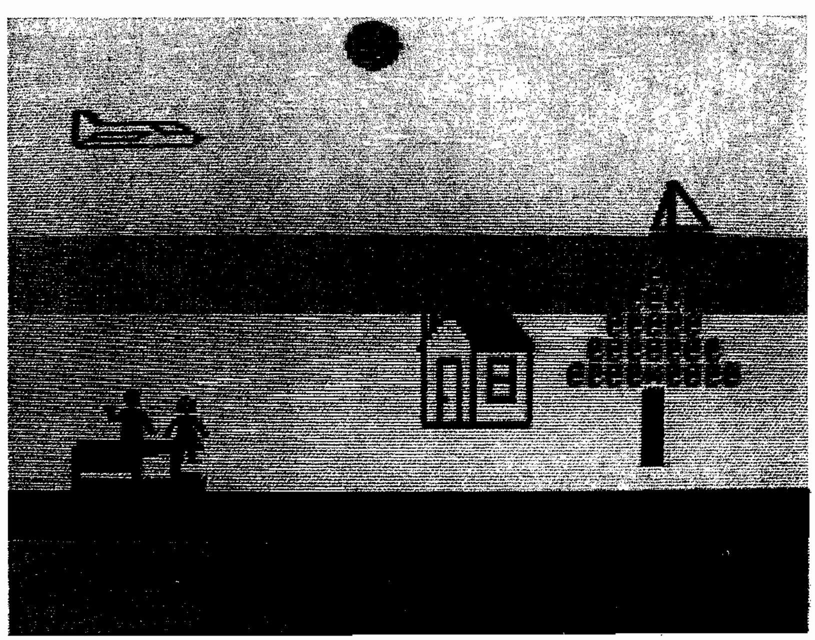


Figure 3: Turtle II sprite editor showing one of the built-in sprites.

Figure 4: Turtle II sprite demonstration uses seven sprites and low-resolution character graphics.



printing a simple message to having the Turtle draw a complex maze and then find its way out. As an example of the variety and simplicity of commands available in VIC TURTLE GRAPHICS, consider the following program, which draws two inter-connected spirals (one of blue asterisks and one of purple dollar signs). (See figure 1.)

```
 1 CLEAR SCREEN
 2 SCREEN COLOR CYAN
 3 MOVE TO 2-6
 4 PEN DOWN
 5 TURTLE COLOR BLUE
 6 TEXT TURTLE DEMO
 7 PEN UP
 8 MOVE TO 10-9
 9 CALCULATE X=10
10 CHARACTER TO *
11 USE SPIRAL
12 TURTLE COLOR PURPLE
13 CHARACTER TO $
14 LABEL SPIRAL
15 PEN DOWN
16 LOOP X
17 FORWARD INDEX
18 TURN RIGHT
19 LOOP END
20 ROUTINE END
21 STOP
```

The line numbers are used in editing the program only and play no actual role in the program's flow or logic.

Because of the standard VIC's limited memory, TURTLE GRAPHICS is confined to drawing with characters and pre-defined graphic symbols. However, since the VIC graphic set and color palette are fairly rich, the programmer may still draw intricate and imaginative pictures. For example, the manual gives sample programs for drawing a boat, an American flag, and drawing and solving a maze.

C64 TURTLE GRAPHICS II

TURTLE GRAPHICS II for the Commodore 64 is a superset of its VIC cousin. It is also cartridge-based (expanded to 16K of ROM) with its own line editor and menu-driven options, including a trace mode. TURTLE II contains all of the commands and capabilities of the VIC version. However, with over ten times the available memory in the C64 with which to work, TURTLE GRAPHICS II has some substantial enhancements over its VIC-20 counterpart. The most obvious improvement is that the programmer can draw with high-resolution (200 × 320 pixels) lines and curves as well as graphic characters. TURTLE for the C64 can therefore duplicate the full repertoire of graphic tricks found in LOGO. For example, the following short TURTLE II program will draw a "Squiral," a standard LOGO graphics design. (See figure 2.)

```
 1 REMARK LOGO-LIKE "SQUIRAL"
 2 HIRES
 3 SCREEN COLOR WHITE
 4 BORDER COLOR WHITE
 5 TURTLE COLOR BLACK
 6 PEN UP
 7 MOVE TO 100-160
 8 SET HEADING TO 90
 9 PEN DOWN
10 CALCULATE Y=0
11 LABEL ADD TWO
12 CALCULATE Y=Y+2
13 FORWARD Y
14 ROTATE RIGHT 123
15 TEST IF (Y>199)
16 IF FALSE JUMP ADD TWO
17 STOP
```

Another significant addition to TURTLE GRAPHICS II is complete support within the language for the Commodore 64's sprite capabilities. TURTLE II has its own built-in sprite editor and comes with eight pre-defined sprites. These include a sailboat, rocket, truck, ball, airplane, house, boy, and girl. Figure 3 shows the sprite editor displaying the sailboat. Using this editor the programmer can create unique sprites, change their color, length and/or width, and save them on tape or disk for later use. The manual also explains how sprites created and saved by the TURTLE editor may be loaded and used in a BASIC program.

Once the programmer has designed his or her sprites or selected from the pre-defined shapes, these sprites may be used in a TURTLE GRAPHICS II program. The available commands in the language have been expanded to over sixty. Using some of the new commands it is possible for the TURTLE II program to place a sprite on the screen, give it a direction and a speed, and send it on its way. Sprites may be moved with or without wraparound if they go off an edge of the screen. There are commands to make sprites visible or invisible, to freeze or thaw their motion, to check for collisions, and even to control their motion using a joystick. Using TURTLE GRAPHICS II's sprite commands allows the programmer to create original versions of simple games such as *Space Invaders* or *Breakout*.

One of the tutorials in the manual develops a game of *Tag* between two sprites (the rocket controlled by the joystick and the ball moving randomly) and calculates a score based on how long it takes to make the tag. Obviously the games will lack arcade speed and sophistication. However, they will still be a valuable and fun learning experience for the beginning programmer and will help to remove some of the mystery of how arcade games work.

All the sprite movements are handled during the hardware or "jiffy" interrupts every 1/60th of a second. Because of this the TURTLE II programmer need not worry about programming the actual sprite movement; i.e., placing a sprite, waiting a set time, changing the sprites coordinates, waiting some more, changing the coordinates again, *ad nauseam*. Instead, the programmer just aims the sprite, sets its speed, turns it loose, and forgets it. Using this feature the TURTLE II programmer may have as many as eight sprites on the screen while the Turtle is drawing in either hi-res (lines) or lo-res (characters) — with everything moving at the same time! Figure 4 shows one of the sample programs from the manual in which the Turtle draws a seashore environment and animates the scene with seven different sprites.

Conclusion

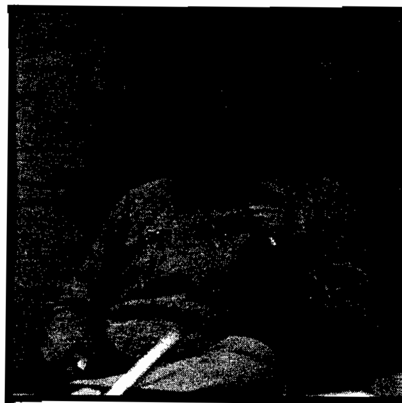
As conceived by Seymour Papert, Turtle Graphics is an exciting and effective way for children and other first-time programmers to develop a solid foundation in programming and computer concepts, as well as to sharpen their thinking and problem-solving skills. The TURTLE GRAPHICS language for the VIC-20 and C64 was designed to fulfill Papert's original vision and to exploit the tremendous sound, color, and graphic capabilities of these two computers.

1. Seymour Papert, *Mindstorms: Children, Computers, and Powerful Ideas*, Basic Books, 1980.

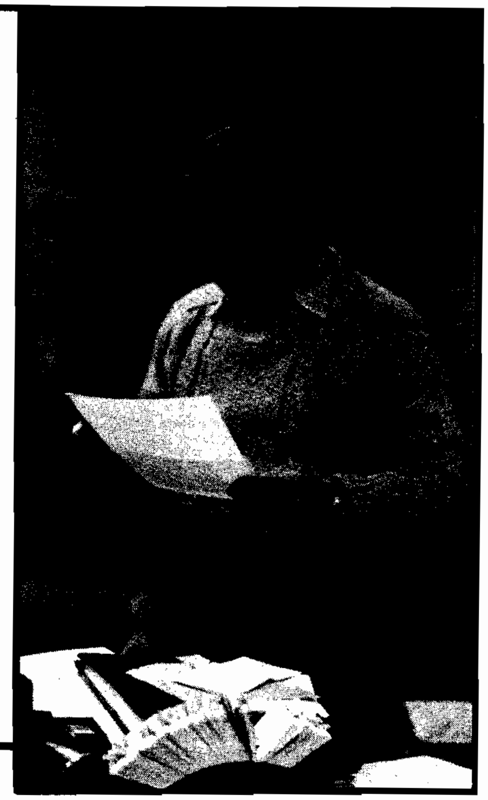
David Malmberg is the author of TURTLE GRAPHICS and TURTLE GRAPHICS II. He is also a Contributing Editor and a frequent writer for MICRO. You may contact Mr. Malmberg at 43064 Via Moraga, Fremont, California 94539.

MICRO™

Making More Than Money in the Silicon Valley



Nathan Schulhof of Silicon Valley Systems.



by Marjorie Morse

Silicon Valley Systems (SVS) can hardly be labelled "just another software company." Although it does have some of the usual characteristics of today's companies joining the computer fields (started by two people on a shoestring) SVS is still thriving, three years later. Many software and hardware companies never made it past the first year. SVS not only made it, but has produced several high quality software packages — some of the best in the industry.

Helping Handicapped Kids

One very special aspect of SVS is its dedication to helping disabled children in the San Francisco area learn about computers. Just about every month 10 to 20 volunteers from SVS gather together a few dozen computers and lots of software and visit a home or hospital for handicapped or disabled children.

In January this year they went to the Watership Home for the Mentally Retarded in Palo Alto. February brought them to Stanford Children's Hospital. Since then they have been to the Shriner's Burn Institute twice and plan to go again. During these visits the children are allowed to use the computers and any of the educational and game software the SVS crew has brought along.

Nathan Schulhof, president and founder of SVS, is enthusiastic about his company's volunteer project. Although each monthly venture is costly, Schulhof feels it is more than worth the time and money. "We grow from this," he says. "When you go out (to one of these hospitals or homes) you feel like a big person." Schulhof emphasizes that participating in these weekend adventures with the children makes a person realize the limit of his or her own problems. "We don't have problems, we have challenges. These kids have problems."

In addition to the computers and software, Schulhof also brings along a mime, a magician, and a singer to entertain the children. The volunteers pass out popsicles and balloons, teach the children how to operate the computers, and challenge them to many of the video games.

About the President

After talking with Nathan Schulhof for a few hours, it is easy to understand why his company is so successful, why he chose to embark on the weekend projects for the handicapped children, and why his employees are so willing to participate with him.

First a little history

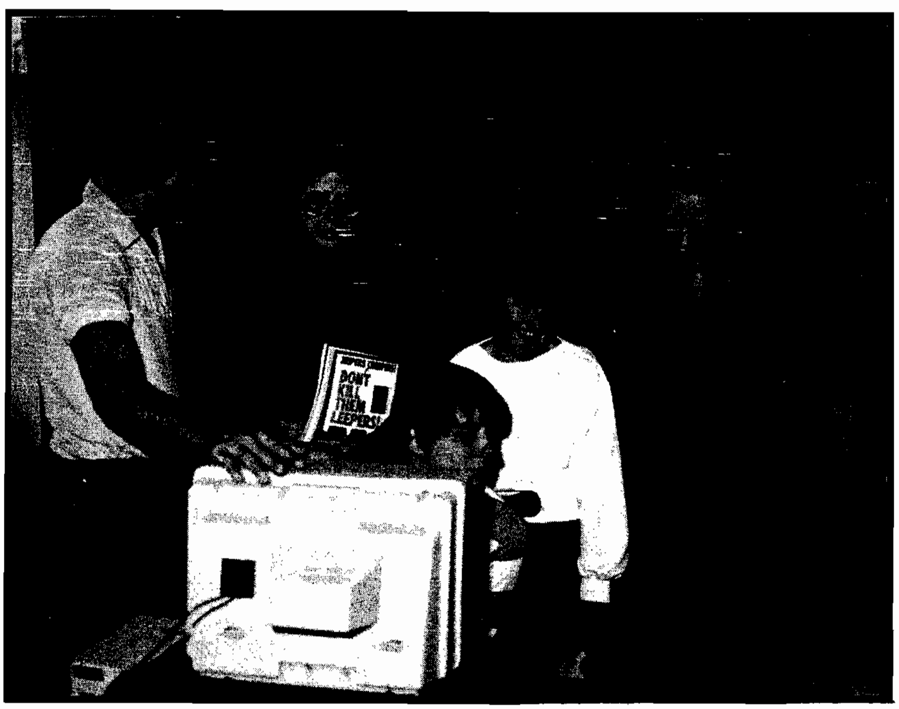
Schulhof actively stepped into the microcomputer industry in 1980 after he realized this new frontier was going to be taken over by businessmen. Schulhof considers himself a businessman first — which is clearly evident by his background. Past job positions include vice president of a land development company, an officer for a public company, an author and lecturer in the field of behavior modification, and a clinical psychologist on the staff of San Francisco General Hospital.

In 1980 Schulhof contacted Leonard Elekman, "one of the brightest and most creative engineers," and arranged for Elekman to build a word processor for the Apple. When *WORD HANDLER* emerged the next year, it was the first Apple word processor to provide high-resolution graphics. Schulhof and Elekman were now ready to start the production wheels rolling for their company, Silicon Valley Systems.

Two years later SVS employs more than 50 people and does $6 million in sales annually. Schulhof's company has been a tremendous success and his employees are content, happy, and loyal. He attributes these positive results in a large part to his psychology experience. "I have been a behavorial scientist for ten years dealing in such habits as drugs, marital problems, alcohol, and violent social problems in private practice, hospitals, government agencies and universities. I have been a law student for two years and a businessman and corporate builder for 12 years. Using my knowledge of people and their habits, the laws that society is based on, and my experience in business have been extremely helpful in guiding me in most of my business decisions."

Silicon Valley System's employees challenge residents to computer games.

Schulhof's positive and progressive attitudes obviously make for a friendly, exciting, and rewarding atmosphere for his employees. "Everyone is important," he says. Employees have the right to switch departments and Schulhof likes to promote from within.



Volunteers assist residents of the Watership Home for the Mentally Retarded.

"There is no bottom at Silicon Valley. This staff is not a good staff, it's a GREAT staff."

Customer service at SVS is also given careful attention. Schulhof offers free upgrades and lifetime guarantees for all his products. After all, they must keep up with the motto that has become part of their ad campaign: "Simply the best."

The Products

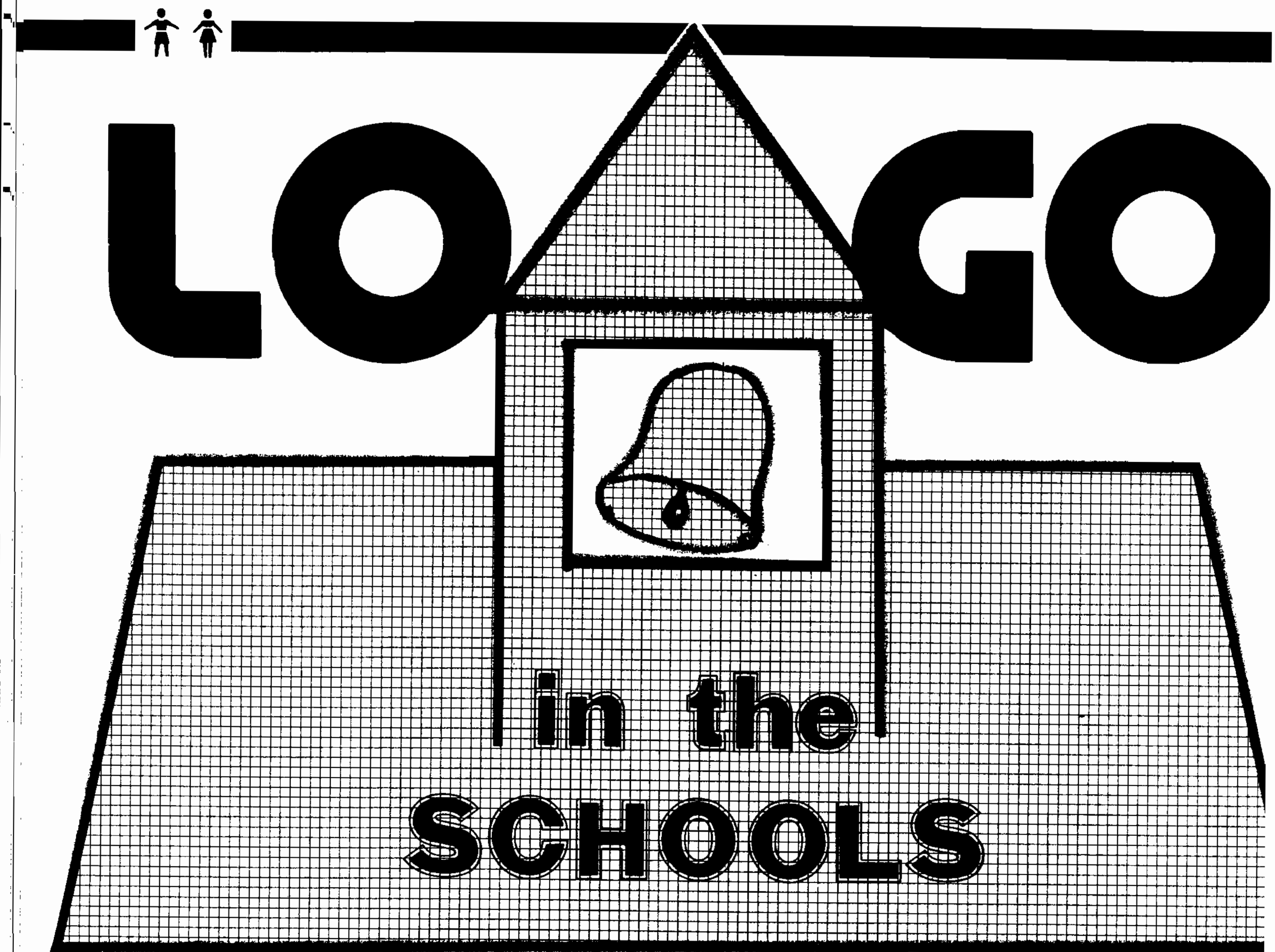
As mentioned before, SVS' first entry into the software market was Word Handler. This program, now offered as *Word Handler* II, is an easy-to-use, simplified word processor that comes on a copy-protected disk and creates non-standard text files. Word Handler uses the high-resolution graphics screen for display, eliminating the need for an 80-column card or lower-case adapter. In addition to normal word processing commands, Word Handler has a keyboard fill letter capability.

SVS'product line includes *List Handler* (which can interface with *Word Handler*), the *Turbocharger* for speeding up DOS, *E-Z Learner*, an educational program that stores and reviews questions and answers, *Rapid Reader, Apple Source, The Snapper, and Final Analysis.*

In keeping with their past generosity, SVS initiated a software give-away program this year. They plan to hand out over $1,000,000 worth of word processing and educational software to public schools. If your school maintains an active computer curriculum and would like to receive free software, contact SVS. Let them know who you are and what computers and software your school uses now. Schulhof says the program has been very successful thus far. "We've been receiving 150 letters a day for the last sixty days. We have boxes of requests."

What do these volunteer projects and give-away programs do for Nathan Schulhof and Silicon Valley Systems? "Sure I like the publicity," Schulhof admits. All this generosity is bound to create strong positive sentiments toward the company. But it is obvious that Schulhof and the people at SVS aren't just in it for the profits and publicity. "We want to share our enthusiasm and knowledge of the computer world of tomorrow with the kids of today," says Schulhof, sincerely. "These kids have changed my life."

MICRO



LOGO in the SCHOOLS

by Phil Daley

Many school systems are adopting Logo as a language to learn in the elementary grades. Logo allows for fast, interactive programming with immediate feedback, interesting graphics with simple commands, and a structured procedure-oriented approach that is both fascinating for the students and offers a sound basis for programming experience.

At Hillsboro-Deering Cooperative School in Hillsboro, NH, the Computer Department starts teaching Logo in the fourth grade. Students are encouraged to experiment with turtle graphics using Harold Abelson's book, *Apple Logo* (Byte/McGraw-Hill, Peterborough, NH), as a resource for basic shape programs. The school has 15 Apple IIs, so the students can work two to a station during their once-a-week assigned time. Students who are especially interested can also work after school.

Starting in the ninth grade, during the first semester students are taught BASIC programming and are required to write 50 elementary programs in BASIC. During the second semester, the students write the same 50 programs in Logo, allowing them to see the effects of a structured language on their programming techniques. This also acquaints them with using Logo as a regular programming language without the turtle graphics. Those students electing to continue their computer studies for the second year learn to do the same 50 programs in Pascal. This transition from Logo to Pascal is much easier than for students starting on BASIC and switching directly to Pascal. Logo gives them a sense of working with a text editor and language processing that, while much simpler to operate, is similar in structure to the Pascal operating system environment.

Included with this article are several examples of programs modified from the Abelson book and examples of original Logo programs by the Hillsboro students. Especially notable is the Math Drill program written by Schyler Jones for use by the younger students as both a math exercise and an example of programming techniques.

Programs and Graphics begin on page 36

In the flash of one second, Delta-10 can print the alphabet six times.

At 160 cps, 100% duty cycle, Delta-10 is built for speed. It's the dot matrix printer that quickly strikes up an intense working relationship with your computer.

There's no nonsense. Delta-10 performs with serial and parallel interfaces as standard. The 8K buffer gives Delta-10 the power to instantaneously store and print data as it frees your computer to continue its job. Turn Delta on and it can print three crisp copies simultaneously at the continuous speed of 160 cps.

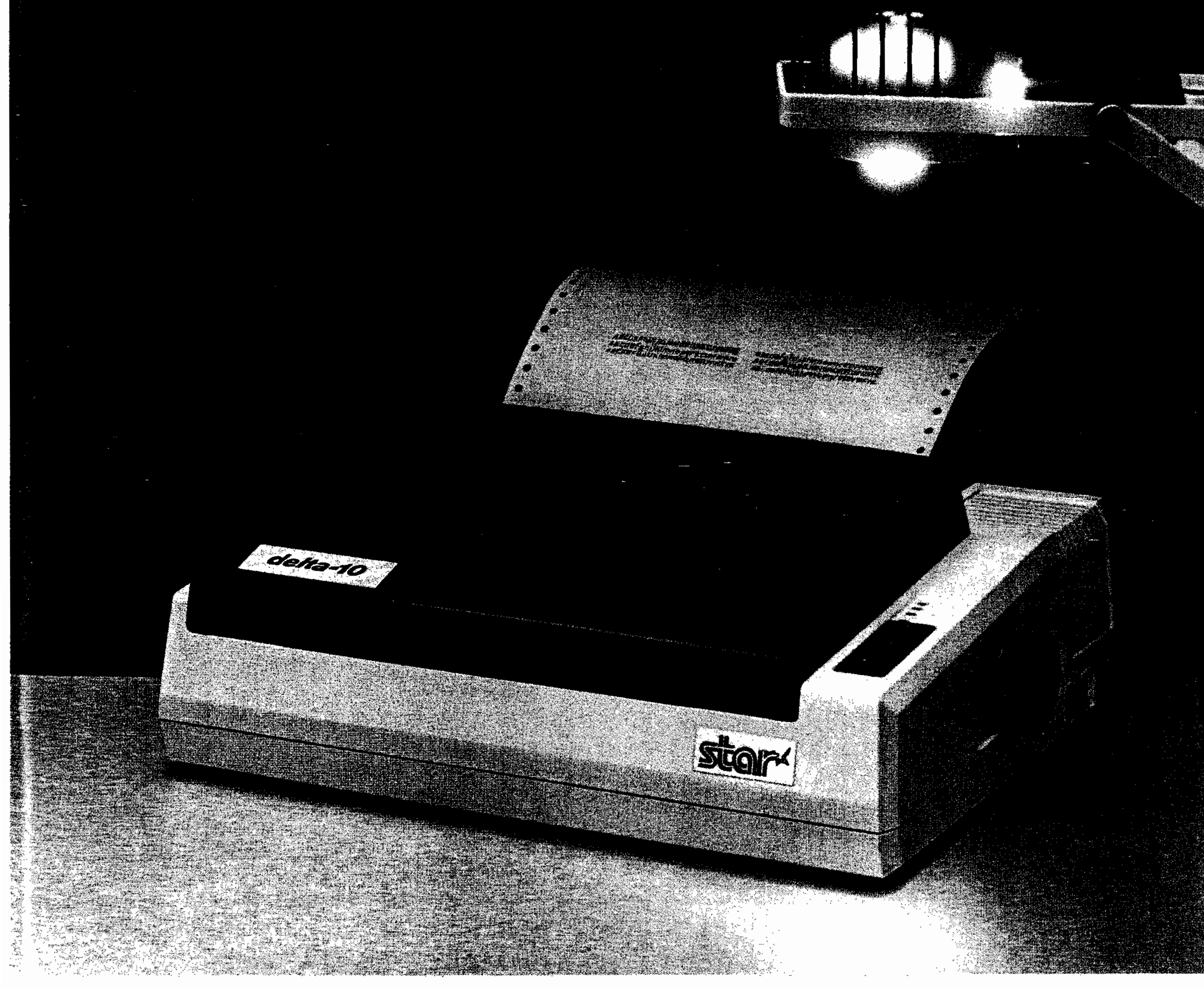
And Delta-10's performance isn't only measured in swiftness. There's the diversity of its character fonts, its true descender matrix, and its underlining ability.

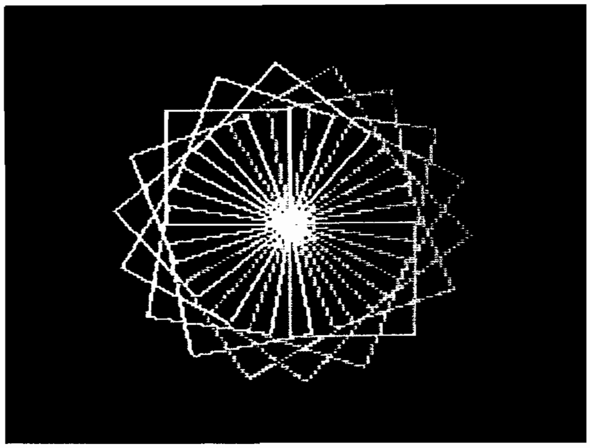
Star's new Delta-10. Designed for red-hot efficiency. And isn't that what you need?

star™
micronics • inc

THE POWER BEHIND THE PRINTED WORD.

Computer Peripherals Division
P.O. Box 612186, Dallas/Ft. Worth Airport, TX 75261





SPINBOX

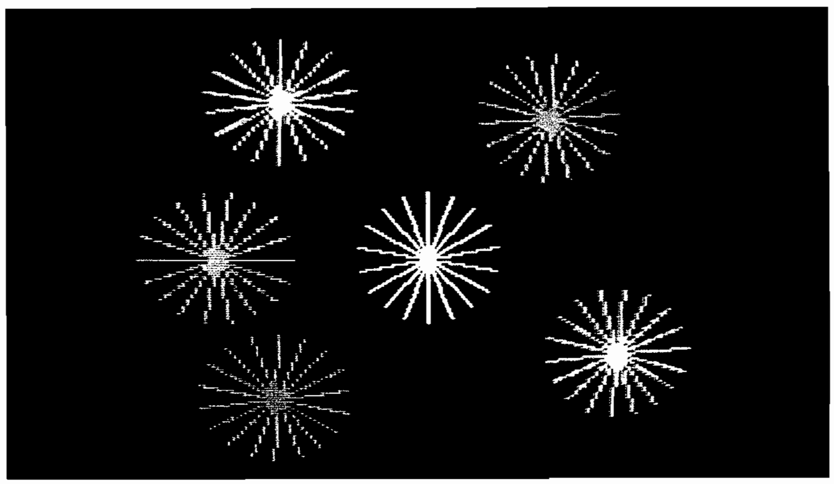
```
TO SPINBOX :SIZE
 MAKE "COLOR 1
 SPINSQR :SIZE
END

TO SPINSQR :SIZE
 HT FULLSCREEN
 MAKE "COLOR ( :COLOR  5.N2 )
 IF :COLOR > 5.8 THEN MAKE "COLOR 1
 PC INTEGER ( :COLOR )
 SQUARE :SIZE
 RT 20
 SPINSQR :SIZE
END

TO SQUARE :SIZE
 REPEAT 4 [FD :SIZE RT 90]
END
```

GROWSQUARES

```
TO START
 DRAW HT FULLSCREEN
 NOWRAP PC 1
 GROWSQUARES 1
END

TO GROWSQUARES :SIZE
 RSQUARE :SIZE
 RT 20
 GROWSQUARES :SIZE  2
END

TO RSQUARE :SIZE
 REPEAT 4 [FD :SIZE RT 90]
END
```

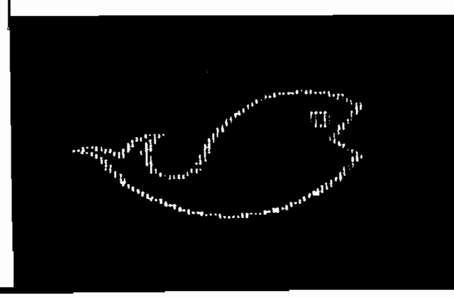
FISH

```
TO START
 HOME HT FISH PU RT 25
 FD 40 LT 70 FD 30 PD
 ARCRIGHT 3 360
 PU LT 30 FD 15 PD
END

TO FISH
 SPOT ARCRIGHT 50 100 RT 100
 FD 20 LT 100 FD 15 RT 100
 ARCRIGHT 75 100 ARCLEFT 15 50
 RT 160 ARCRIGHT 20 60
 LT 100 ARCRIGHT 20 60
 RT 140 ARCLEFT 25 25
 LT 50 FD 10 LT 50 FD 10
 LT 35 FD 10
END

TO SPOT
 BG 1 PC 2 PU LT 90 FD 50
 RT 110 PD
END

TO ARCLEFT :RADIUS :DEGREES
 ARCLEFT1 :RADIUS * 1.74N2 :DEGREES
END

TO ARCRIGHT :RADIUS :DEGREES
 ARCRIGHT1 :RADIUS * 1.74N2 :DEGREES
END

TO ARCLEFT1 :SIZE :DEGREES
 REPEAT :DEGREES [FORWARD :SIZE LEFT 1]
END

TO ARCRIGHT1 :SIZE :DEGREES
 REPEAT :DEGREES [FORWARD :SIZE RIGHT 1]
END
```



FIREWORKS by *Liz Douglas — 8th Grade*

```
TO FIREWORKS
 FULLSCREEN HT
 PU LT 90 FD 100 PD PC 5 FIRE
 PU RT 195 FD 200 PD PC 3 FIRE
 PU LT 120 FD 130 PD PC 4 FIRE
 PU LT 123 FD 200 PD PC 2 FIRE
 PU HOME PD PC 1 FIRE
 PU LT 40 FD 110 PD PC 3 FIRE
END

TO FIRE
 REPEAT 18 [RT 20 FD 35 BK 35]
END
```

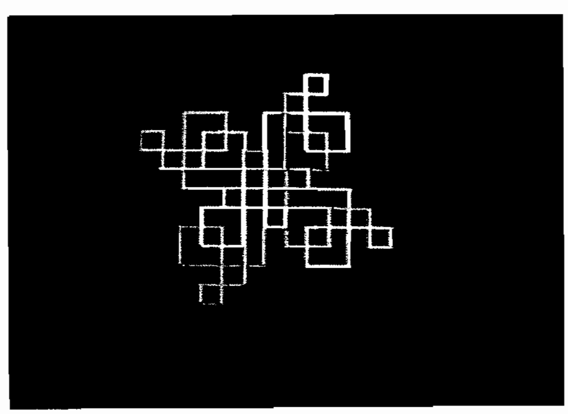


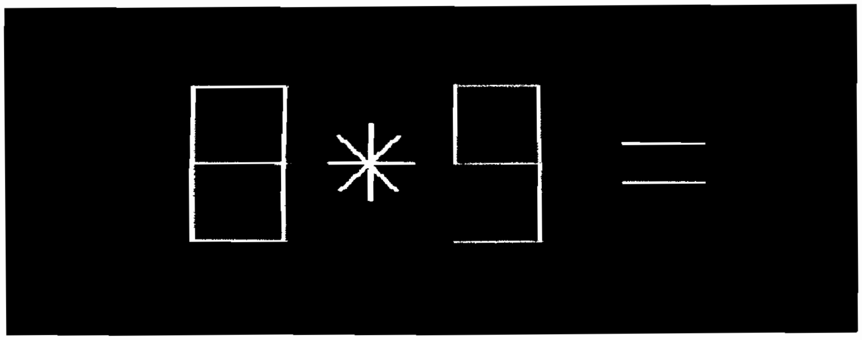
CRYSTAL

```
TO CRYSTAL
 HT FULLSCREEN SHAPE
 LT 45 FD 70 CRYSTAL
END

TO SHAPE
 MAKE "D 40 LINE :D LINE :D
 MAKE "D 20 LINE :D LINE :D
 MAKE "D 40 LINE :D
 MAKE "D 10 LINE :D LINE :D
 FD 20
END

TO LINE :DISTANCE
 FD :DISTANCE RT 90
END
```

JENGU by *Ben Daley — 4th Grade*

```
TO START
 CS FULLSCREEN
 MAKE "COLOR 1
 HT JENGU
END

TO JENGU
 SHAPE SHAPE LT 90
 MAKE "COLOR ( :COLOR  1 )
 IF :COLOR > 5 MAKE "COLOR 1
 PC :COLOR JENGU
END

TO SHAPE
 FD 40 RT 90 FD 40 RT 90 FD 20
 RT 90 FD 20 RT 90 FD 40 RT 90
 FD 10 RT 90 FD 10 RT 90 FD 20
END
```





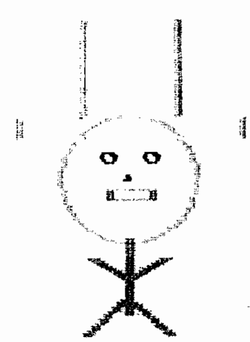
QUIZ *by Schyler Jones — Junior*

```
TO START
 MAKE COUNT 1
 MAKE SCORE 0
 QUIZ
END

TO SETCOLOR NUMB
 MAKE COLOR :NUMB
 IF :COLOR  5 THEN MAKE COLOR :COLOR - 5
END

TO NINE
 PC :COLOR FD 40 RT 180 LINE
 LT 90 LINE LINE LT 90 LINE LT 90 LINE
 LT 90 LINE
END

TO EIGHT
 PC :COLOR LINE RT 90 LINE
 RT 90 LINE RT 90 LINE RT 180
 LINE LINE LT 90 LINE LT 90 LINE
END

TO SEVEN
 PC :COLOR FD 40 RT 117
 PD FD 90 LT 117 FD 40
 LT 45 FD 10 PU
END

TO SIX
 PC :COLOR FD 40 RT 90 FD 40
 RT 90 LINE RT 90 LINE RT 90 LINE
 RT 90 LINE LINE RT 90 LINE
END

TO FIVE
 PC :COLOR FD 40 RT 180 PD FD 30
 LT 45 FD 15 LT 45 FD 30 LT 45
 FD 15 LT 45 FD 30 RT 90 FD 30 RT 90
 LINE
END

TO FOUR
 PC :COLOR RT 90 LINE LINE
 RT 180 FD 40 RT 90 LINE
 RT 90 LINE
END

TO THREE
 PC :COLOR FD 40 RT 180 LINE
 LT 90 LINE LT 90 LINE
 RT 180 FD 40 LT 90 LINE LT 90
 LINE
END

TO TWO
 PC :COLOR LINE
 RT 135 LINE PD FD 20 LT 45 FD 20
 LT 45 FD 20 LT 45 FD 20 LT 45
 FD 20 PU
END

TO ONE
 PC :COLOR PD LINE
 PU BK 20 RT 90 LINE
 LINE LT 135 PD FD 20 PU
END

TO QUIZ
 MAKE NUM1 ( RANDOM 9 )  1
 MAKE NUM2 ( RANDOM 9 )  1
 MAKE TYPE ( RANDOM 3 )  1
 IF :TYPE  1 THEN MAKE ANSWER :NUM1  :NUM2 MAKE SIGN
 IF :TYPE  2 THEN MAKE ANSWER :NUM1 - :NUM2 MAKE SIGN -
 IF :TYPE  3 THEN MAKE ANSWER :NUM1 * :NUM2 MAKE SIGN *
 DRAWNUMBER :TYPE :NUM1 :NUM2
 CLEARTEXT
 PRINT ( SENTENCE [HOW MUCH IS] :NUM1 :SIGN :NUM2 [?] )
 MAKE REPLY READNUMBER
 TEST :REPLY  :ANSWER
 IFTRUE MAKE SCORE :SCORE  1
 MAKE COUNT :COUNT  1
 IFFALSE PRINT SENTENCE [NO, THE ANSWER IS] :ANSWER
 TEST :COUNT  10
 IFTRUE PRINT ( SENTENCE [YOU SCORED] :SCORE [OUT OF A
 POSSIBLE 10] ) STOP QUIZ
 END

 TO READNUMBER
  OUTPUT FIRST REQUEST
 END

 TO NUMBER NUMB
  IF :NUMB  1 THEN ONE
  IF :NUMB  2 THEN TWO
  IF :NUMB  3 THEN THREE
  IF :NUMB  4 THEN FOUR
  IF :NUMB  5 THEN FIVE
  IF :NUMB  6 THEN SIX
  IF :NUMB  7 THEN SEVEN
  IF :NUMB  8 THEN EIGHT
  IF :NUMB  9 THEN NINE
 END

TO DRAWNUMBER TYPE NUM1 NUM2
 PU HT CS HOME LT 90 FD 80
 MAKE NUMB :NUM1
 SETCOLOR :NUMB
 NUMBER :NUMB HOME FD 40 LT 90 FD 20
 DRAWSIGN :TYPE HOME RT 90 FD 40 LT 180
 MAKE NUMB :NUM2
 SETCOLOR :NUMB NUMBER :NUMB
 HOME FD 30 RT 90 FD 80 EQUAL
END

TO DRAWSIGN TYPE
 PD PC 1 LINE
 IF :TYPE  2 THEN STOP
 BK 20 RT 90 BK 20 LINE
 IF :TYPE  1 THEN STOP
 RT 135 FD 14 LT 90 FD 5
 RT 180 DRAWSIGN 1
END

TO EQUAL
 PD PC 1 LINE PU LT 90 FD 20
 LT 90 FD 1 PD LINE PU
END

TO LINE
 PD FD 40 PU
END
```

HOW MUCH IS 8 * 9 ?

MAN *by Kim Zeoli — 8th Grade*

```
TO FACE
 CIRCLE .5555 4 MOVEF
 HAT EYES NOSE MOUTH BODY
END

TO CIRCLE :SIZE :COLOR
 PC :COLOR
 BG 1 HT FULLSCREEN
 REPEAT 360 [FD :SIZE RT 1]
END

TO MOVEF
 RT 90 PU FD 10 LT 90
 FD 21 PD
END

TO HAT
 PC 3 BG 1 LT 90 FD 30 RT 90
 FD 10 RT 90 FD 30 LT 90
 FD 50 RT 90 FD 45 RT 90
 FD 50 LT 90 FD 30 RT 90
 FD 10 RT 90 FD 30
END

TO EYES
 BG 1 PU FD 10 LT 90 FD 10 PD
 CIRCLE 5.N2 0
 RT 90 PU FD 20 LT 90 PD
 CIRCLE 5.N2 0
END

TO NOSE
 PC 0 BG 1 PU HOME
 RT 90 FD 32.5 PD LT 120
 FD 3 LT 120 FD 3 LT 120
 FD 3 HT
END

TO MOUTH
 PC 2 BG 1 RT 90 PU FD 10
 PD RT 90 FD 10 RT 90 FD 5
 RT 90 FD 20 FD 5 RT 90
 FD 10 HT
END

TO BODY
 PC 5 BG 1 LT 90 PU FD 20 PD
 FD 40 BK 20 LT 120 FD 20 BK 20
 LT 120 FD 20 BK 20 LT 120 FD 10
 RT 45 FD 30 BK 30 LT 90 FD 30
 BK 30 HT
END
```



MICRO

DECISIONS DECISIONS...

Make the right ones and you rule the galaxy!

CONQUERING WORLDS™

by
Walter Hochbrueckner

Strategy is the heart of the game. For weapons may win battles, but the right strategy wins wars! As the Supreme Commander of one isolated world you have little influence, and less power. But, you're ambitious. You dream of a united galaxy—united under your banner. The only solution is to plan, plot, evolve an effective strategy, then go on the offensive.

On-screen commands and your star map let you choose where in the galaxy you want to launch your attacks. But, to capture 32 solar systems (with up to 8 planets each) you must constantly make major decisions. How many warships are you willing to commit? How many must you hold in reserve to defend your base? And you must be eternally ready to switch tactics with the speed of thought, as the enemy is intelligent, well armed and may control many of the planets you've marked for conquest.

Will a united galaxy become reality or remain a dream? That's the exciting challenge that'll keep you intrigued for weeks on end.

For your Apple Computer. Only $29.95
at your computer store, or from:

DATAMOST

8943 Fullbright Chatsworth, CA 91311-2750 (213) 709-1202

VISA/MASTERCARD accepted. $2.00 shipping/handling.
(California residents add 6½% sales tax.)

The Silicon Blackboard

> *A master who lived as a hermit on a mountain was asked by a monk, "What is the Way?"*
>
> *"What a fine mountain this is," the master said in reply.*
>
> *"I am not asking you about the mountain, but about the Way."*
>
> *"So long as you cannot go beyond the mountain, my son, you cannot reach the Way," replied the master.*
>
> Zen Buddhist saying

by Emmalyn Bentley

Using computers in the school system provides us with a relatively new, untapped intellectual resource, and there are probably as many different approaches to the fulfillment of a proficient computer curriculum as there are educators. The enormous complexity of developing such a system is almost as overwhelming as the thought of rediscovering the wheel!

One of the major difficulties we face is creating an educational environment that does more than just inundate the student with an informational resource — the student must be taught to *use* that resource. More critical is the importance of developing skills in structural thinking, critical evaluation, and even programming, which is fast becoming a basic tool in today's society. The computer can, indeed, become a powerful learning mechanism. Andrew Molnar has said that "computing is so compelling a tool that it cannot be stopped." There is little doubt that by the turn of the century computers will be the major way of learning — at all grade levels and in all classroom subjects.

In the late 1940's a small exploratory movement planted the technological seed that has grown into the lucrative full-blown computer industry of the 1980's. It has been a vital movement — never stagnant these last 35 years. But enthusiastic intent often gets tripped up by the actuality of "doing." And this is the point at which we now stand. The computer, like Guttenberg's printing press, has the capability to radically alter the academic scene. We stand at the threshold of an exciting and stimulating time in educational history. It is also a time when we must train our future work force to meet the demands of an ever-expanding high-tech industry — an industry that is fast becoming globally competitive. As we step across that threshold we must determine which direction to take and who will lead us; who should be the motivating force behind this educational crusade? Teachers? School administrators? Computer professionals? The government? Or you, the parents of the children who will eventually be the nucleus of our computer-revolutionized world? What responsibility do you shoulder with regard to your children's educational future? How deeply should you become involved in implementing and directing computer curriculums for your particular school system? Perhaps you are one of the small minority who is waiting to see if computers are just a fad (although that is not likely if you are a MICRO reader). Believe it or not, there are educators who are pondering whether or not computers will be around in the future! But most of us realize that the time is fast approaching when the computer illiterate will be the uneducated and the unemployable.

According to Paul E. Tsongas (Senator from Massachusetts), "...our educational system is badly underfunded and failing to equip our citizens with even basic skills, let alone technological skills required for future jobs." Japan, on the other hand "...maintains a rigorous educational system with a heavy concentration on science and math...," two important areas woefully neglected in the United States.

In a speech at Massachusetts Institute of Technology in Cambridge, Sen. Tsongas spoke about Japan's successful efforts to capture "70% of the market for the most advanced commercial memory chip, the 64K RAM." They have also begun a "Fifth Generation Computer Project, whose goal is to develop and commercialize a seeing-hearing-speaking computer with powerful problem-solving capability." If we want to compete with Japan (and such countries as South Korea, Taiwan,

(Continued on next page)

Singapore, and Hong Kong) it is imperative that we upgrade our educational system. With proper funding, equipment, and direction we can maintain equal footing with our global competition and, perhaps, surpass them. The Japanese, for instance, while producing highly competent and productive workers, rely on rote learning and drill and practice. Our strength lies in encouraging creativity, experimentation, and innovation.

The rest of this article will, hopefully, provide you with some food for thought or, so to speak, data to mentally process. And when you finish reading, perhaps you will be inspired to take positive action to help create a deeply fulfilling and enriching educational environment for your sons and daughters. The following subject matter is based on the premise that we have excellent software and guidance in our school systems. (More later on what some schools are actually doing.)

Computers present a compelling advantage over our present educational system for several reasons. First, they allow the student to *actively* participate. Education begins very early in life with play, a personal learning process in which individuals interact with one another. The computer creates a similar environment of interaction in a visual manner. It should not, however, replace real events and experiences; it should and can provide a means for the user to gather information in a highly motivational way — a lot like play.

The computer can provide individualized education to each student in a unique fashion. It encourages "solo learning," allowing the student to work at his own pace without prejudice (conscious or subconscious). Some learning may shift from the school to the home as more and more personal computers find their way into our lives. According to *American Family* (the National Newsletter on Family Policy Programs Since 1977), "Home sales will overtake the now dominant school market shortly to capture 70% of the market by 1987...." But bear in mind that computers are not teacher-proof — they should support person-to-person education, not replace it. We must imbue our computer curriculums with respect for the importance of human relationships. Evaluation, direction, and disciplined study are still in the realm of the teacher's responsibility.

This creates a paradox. Acccording to Thomas Dwyer, professor of computing science at the University of Pittsburg for more than 10 years, "...the complexity inherent in human nature should 'drive' the relationship between technology and education; ...deep technology is of little value without a deep view of education...." The paradox lies in the fact that to make this philosophy work in the real world, we must depend a good deal on advanced technology!

Computerized education will enable the student to learn important ideas earlier in the educational stage than might be the case otherwise. New courses will have to be created to fulfill the needs of ordinary students who will be working beyond today's present standards. Teachers will have to be reeducated and new curriculums developed. This, in turn, will have a terrific impact at the college and university levels. If the grammar schools and high schools perform their tasks well, freshmen entering college will expect a truly sophisticated level of computer education. This expectation may very well be a determining factor in selecting which college to attend.

Drexel University in Philadelphia, Pennsylvania, has already addressed this problem. They announced in October 1982 that 1983 incoming freshmen would be required to own and operate their own personal computer, regardless of their course of study. President Dr. William W. Hagerty explains, "The policy will change both the way courses are taught and the way students learn.... (It) will also have a major effect on the faculty in as much as it will make them more creative and valuable."

In order to implement this policy, Drexel has made a deal with Apple to buy large quantities of a new, as yet unadvertised, computer (known at press time only as the Apple DU, but I suspect to be the Mackintosh). The 64K machine comes with video monitor, disk drive, built-in software, and a high-capacity disk for additional mass memory — all for $1,000, which can be financed through the university. Ray Ulmer, Director of Public Relations at Drexel, claims that "with its revolutionary user-interface, a student who has never used a computer will be productive in 30 minutes." And because many students at Drexel are in a work/study program, the fact that they have their own computers will no doubt make them more valuable to their employers outside the university environment.

Another fundamental problem that needs to be addressed is defining the goals of education — which is tantamount to holding the proverbial tiger by the tail. Let's look at some of the pitfalls to avoid. We must not look at education only in terms of a cost-effective delivery system in which the computer rather than the teacher provides information. We must not allow drill and practice to lull us into a sense of complacency. We must avoid mechanical teaching methods. Teacher and student should be viewed as part of the total curriculum team. And because the best of ideas fulminate in a cooperative spirit, the ideal situation would be that where student, teacher, and computer interact. Students should have a certain amount of freedom to make individual forays into educational exploration according to their interests, intellect, and emotional growth; but this should be tempered with the guidance and knowledge of an established educational past — something the computer cannot provide. *Teachers are not expendable!* The focus should be on the teacher-student relationship; the computer is simply a vehicle for enhancing that relationship.

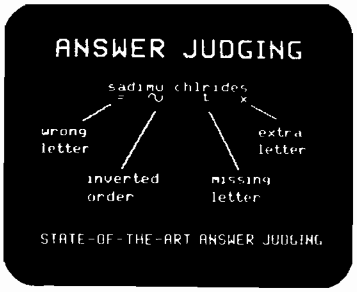
Now for the tough part. All of these marvelous features that computer education promises are for naught if we do not have the proper direction, the software and hardware, the terminology, the communication mechanisms, and the funds to make it work. How do you establish such a base? Here is what three different types of school systems did.

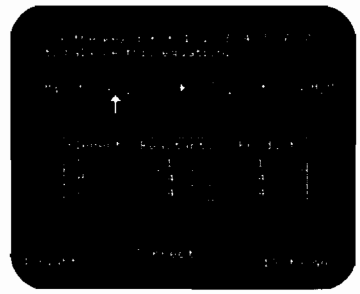
According to Richard Burpee, math teacher and computer coordinator for the Computer Awareness and Literacy Curriculum established in the Nashua, New Hampshire, public school system, choice of hardware was determined by the good educational software available; color was also a factor. In this case Apple was deemed the logical solution for the grammar schools. Sanders Associates, a high-tech corporation in the area, provided free courses for the teachers who, incidentally, held their own gradewise with Sanders professionals taking the same courses! Digital Equipment Corporation donated 75% of the hardware (PDP 1144's) at the high school level, and a federally funded block grant was used to fill out the curriculum needs. Having different systems at the two school levels eliminated duplicating libraries.

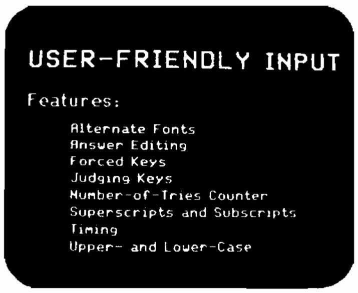
Because they do not yet have

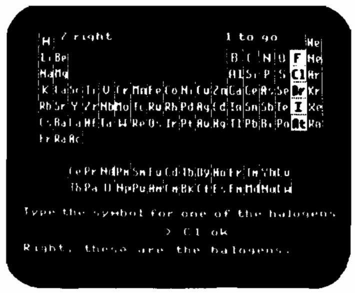
(Continued on pages 42 and 43)

COMPress
INTRODUCES 2 NEW INTERACTIVE COMPUTING PACKAGES
For the Apple









Now YOU can write professional quality interactive Computer-Assisted Instruction materials and simulations

EnBASIC™ can help YOU.

Authors Paul Tenczar, Stanley Smith, and Allen Avner have produced CAI and similar user-oriented software for more than 20 years. Here are routines and authoring aids critical to preparation of high-quality, user-friendly materials.

EnBASIC adds to Applesoft* BASIC. All features of BASIC are still present.

A flexible display design allows for:

- Proportional spacing, superscripts, subscripts, underlining, and automatically backspaced diacritic marks in text
- Double or standard size characters displayed anywhere on the High Res screen
- Lower-case characters on the Apple II plus*

Advanced input handling affords you:

- State-of-the-art answer-judging
- Automatic indication of spelling and typing errors
- Character-by-character error feedback for missing, extra, or wrong letters, inverted letter order, errors in accenting, capitalization, sub- or superscripting
- Synonym lists allowed as part of expected responses

You get all these features simply by specifying a correct response and including an EnBASIC command that implements spelling checks with feedback.

The Package

A **94-page manual** containing a tutorial on use of EnBASIC with examples, implementation hints, and technical details (available separately for $20.00 which may be applied to the full purchase price of $150.00).

A **pocket guide** to **EnBASIC** commands

A master and back up diskettes containing: the EnBASIC augmentation program, six ready-made character sets, four sizes of English letters plus Cyrillic and Greek together with editors which allow you to design your own character sets and redefine key set functions.

A sample program diskette

EnBASIC Package $150.00

Help your students master Introductory Chemistry with:

INTRODUCTION TO GENERAL CHEMISTRY

Professor Stanley G. Smith, Dr. Ruth Chabay, Elizabeth Kean

These materials are designed as a stand-alone supplement to an introductory course in General Chemistry — for students who have no previous Chemistry background. Simulated experiments used to develop basic concepts allow students to gain experience in collecting and interpreting data.

Graphics and animation help to develop an intuitive feeling for chemical concepts. Problem sets provide extensive practice solving problems — beginning wtih easy structured problems and progressing to more difficult, less structured. Diagnostic feedback is provided for student errors, and help is available from the programs when requested.

The first 7 diskettes in the series include:

1. The Chemical Elements (Periodic Table)
2. Inorganic Nomenclature
3. Chemical Formulas and Equations — (Balancing Equations)
4. Atomic Weights
5. Percent Composition
6. Chemaze
7. Gas Laws

Additional topics are being prepared.

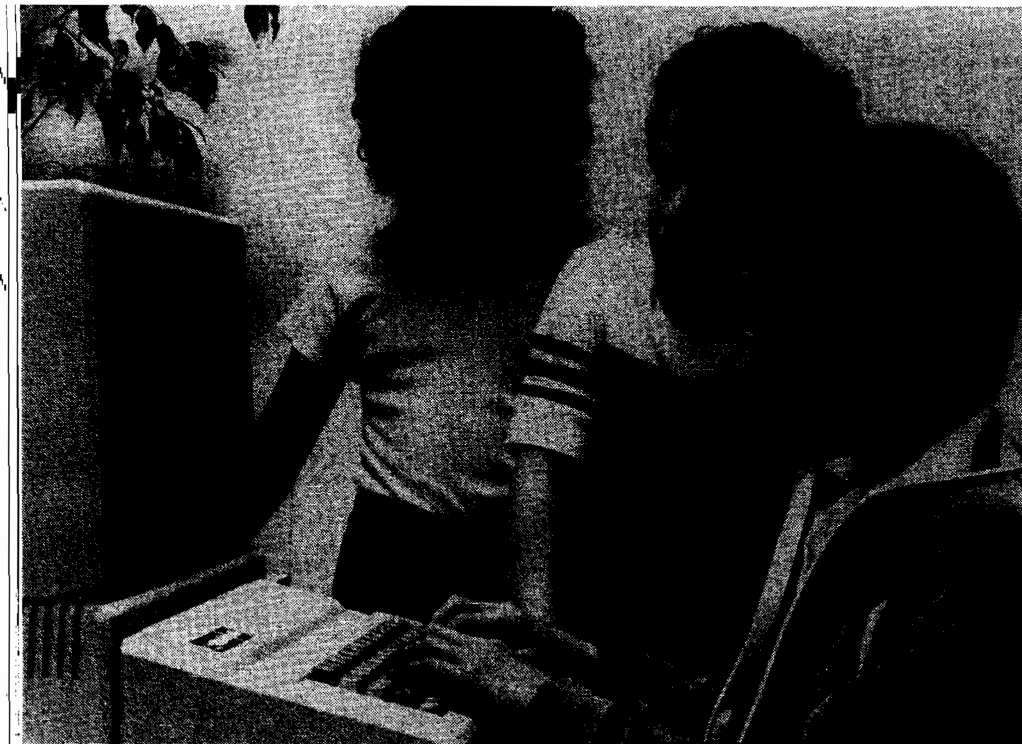
Disks are individually priced at $60.00 each, except Chemaze which is $40.00. If the first seven diskettes are purchased as a group, the price is $340.00. If you want to purchase the entire General Chemistry series, you may place a standing order for the complete package now. We will send you each diskette in the series as completed at the reduced rate of $50.00 per diskette. We will also GUARANTEE that the entire series will not exceed $700.00 regardless of the total number of diskettes when the package is complete regardless of any price increase through January 1, 1984. Backup diskettes for student use at the same location are available for $10.00 each.

A demonstration diskette is available

To order or receive our catalog, call or write today!

COMPress
DEPARTMENT MC
P.O. Box 102
Wentworth, NH 03282
(603) 764-5831 / 5225

A Division of Van Nostrand Reinhold
*Registered trademark of the Apple Computer Company
TM A trademark of Computer Teaching Corporation



Although the idea of computer education in the schools is still in its adolescence, there has been a flurry of activity across the country in anticipation of a future that is close to being the present. The following quotes are indicative of what is happening.

"Cupertino, Calif. — Apple Computer this week launched its program to donate a complete personal computer system to nearly 10,000 public and private schools throughout California...."
(Apple Computer, Inc.)

"Glassboro, New Jersey — Elementary and high school students from throughout the U.S. and Canada are preparing for the fourth Annual "Olympics of the Mind" World Finals Competition.... This year a special computer event will take place at the World Finals. Commodore, one of the sponsors of the World Finals, helped design the event and is supplying the computers for the contest...."
(OM Association, Inc.)

"Framingham, MA — Cullinet Software today announced a special joint program with the Massachusetts High Technology Council (MHTC) for the purpose of improving computer literacy among elementary and secondary teachers in Massachusetts school systems. The program was announced at the opening of Cullinet's new National Education Center in Framingham...."
(Massachusetts High Technology Council)

"I am pleased to announce that the IEEE Computer Society has not only created a Technical Committee on Computers in Education (TCCE), but the Educational Activities Board has established a precollege committee to study curriculum needs and requirements as well...."
(M. Dundee Maples in TCCE Tidbits)

"Montreal — A Canadian company, Logo Computer Systems, Inc. (LCSI) of Lachine, Quebec, is working with Apple Computer Inc. in supplying California schools with the world's newest computer programming software: LOGO. And, in so doing, is playing a key role in launching Apple's Kids Can't Wait Program...."
(Logo Computer Systems Inc.)

"Nashua, NH — During the past summer, a curriculum committee met for two weeks to develop a Computer Awareness and Literacy Curriculum for implementation in the Nashua School District...."
(Bicentennial PTO Orator)

"Walpole, Mass. — Playing computer games in the classroom is legal for the students at Fisher Elementary School; in fact it is even encouraged...."
(Bruce Zweig, Lightning Software)

"Coralville, Iowa — When teacher Jean Mether made room to move three Apple computers into her typing classroom already equipped with 35 manual Olympia typewriters, the setting was ripe for the computer revolution at Northwest Junior High School...."
(Bruce Zweig, Lightning Software)

"Microzine™ from WIZWARE™ — Be the first to get Microzine: a collection of exciting computer programs! ... Get Microzine for challenging software... fun and excitement!"
(Scholastic Arrow Book Club)

"It is not being overly pessimistic to view the microcomputer as the vehicle that may drive a technological and instructional wedge between home and school.

"On the other hand, it is not overly optimistic to think that educational computing could become the basis for cooperative, community-wide educational experiences for all families."
(Dr. Kenneth Komoski, Executive Director Educational Products Information Exchange (EPIE) Institute)

enough equipment, Mr. Burpee states, at the present time Nashua is using their computers "as an *object* of instruction rather than a *tool* of instruction in programming and data processing." He emphasized the importance of parent participation as another resource. Many teachers have not had experience on computers and welcome the added assistance. Teachers take note! If one of your students is computer knowledgeable, don't be embarrassed to ask him/her for help. Many young people have had a considerable amount of computer experience and are highly competent.

In Greenwich, Connecticut, The Mead School for Human Development (a private alternative humanistic school) has been fortunate to have a donor provide them with a number of TRS-80's, TIs, and an Atari (used in the art and music departments). Computer education is not required but it is encouraged. The expectation is that the students (aged 2 - 8th grade) will become hooked on computers early in life. Apparently they are. By the time they reach the first grade, many of Mead's students are well versed in using computer software and are becoming adept in LOGO.

Gaelen Canning, Director of the school, explains that because information is available in increasing amounts, the computer provides the students with another way to not only learn information but also to use it — intellectually and creatively. Children in grades 3 through 8 are encouraged to visit the "Responsive Environment Center" where they are introduced to a rich array of thinking and experiential materials (including computers), which encourage them to explore in their own way and at their own pace — to ask what they can do personally to enrich their lives. From the nursery years onward, the children are also offered specific workshops to learn programming (LOGO for first graders and above and later BASIC), word processing, graphics, and the use of the quality software available. Finally, The Mead School uses the computer in its learning specialties program to meet the specific needs and learning modes of children who need a one-to-one learning experience.

As in Nashua, parents of Mead students are involved in volunteering their time and expertise, particularly at the nursery and kindergarten levels.

Public schools located outside high-tech areas may have more difficulty instituting a computer curriculum, but with diligence and perseverence it can be achieved. Educators in Fargo, North Dakota (population 60,000), established a computer program in their vocational schools three years ago using state and local funds. Two years ago they implemented a course of study at the two high schools. This year they have tentatively adopted a program for kindergarten through ninth grade. According to John Steiner, a teacher in the Fargo public school district, the biggest problem was acquiring hardware, which was done through local bidding and local taxes. To date there are approximately 30 Apples, several Franklins, and two networking systems in the high schools, enabling students to share equipment and thereby cut down on hardware needs.

Fargo uses software from the Minnesota Educational Computing Consortium (MECC, 2520 Broadway Drive, St. Paul, MN 55113), which provides exceptionally good programs for the Apple, Atari, and TRS-80. In order to have unlimited access to these educational packages, the school system pays a small yearly license fee. MECC is an excellent resource for software and is available to the public for the licensing fee.

Teachers in Fargo attend one- to two-day in-service training workshops taught by experts from MECC, computer centers, and other knowledgeable computerists in the area. Unlike the Nashua school system or The Mead School, there is little parent input at this time. Hopefully that will change.

No matter where you live, it is vitally important that you become an involved parent. Some of the things that you can do to help your school system are to become aware of what has already been done then act on that knowledge; share your expertise and, if you have one, your computer; encourage your PTO to raise funds for equipment and software (is it wiser to spend money for a new computer that will last for years or take a field trip that lasts one day?); and put pressure on your elected officials to provide funding.

Some aspects of curriculum development with which you should become familiar are availability of good hardware and software, resource materials and manuals, training for resource people and teachers, funding, and educational organizations that can give you guidance. There are several research centers that can help you get started.

Technical Education Research Centers, Inc. (TERC, 8 Eliot St., Cambridge, MA 02138) is a non-profit public service corporation that provides assistance through planning services, faculty training programs, hardware and software information, and resource material. TERC is not affiliated with any particular machine or educational approach and is therefore able to offer unbiased information.

The Institute for Professional Development (IDP, 245 Nassau St., Suite D, Princeton, NJ 08540) is a non-profit, public service educational research and development corporation that is well informed about state-of-the-art developments worldwide. Its Advisory Board is composed of distinguished educators, scientists, and other professionals from the United States, Canada, England, India, and Australia. This past summer IDP sponsored a conference with workshops and seminars that addressed such subjects as "The Computer's Role in Education: Don't Think About Computers, Think About Education," "Educational Policy: Making Computers Count Rather than Counting Computers," and "Putting It All Together: The Total Curriculum Approach to Computer Literacy, K-12."

There are innumerable other organizations you can contact for information and help. Do your homework now and be the motivating factor in getting an effective computer curriculum into your school system. Put an Apple on your teacher's desk!

* * *

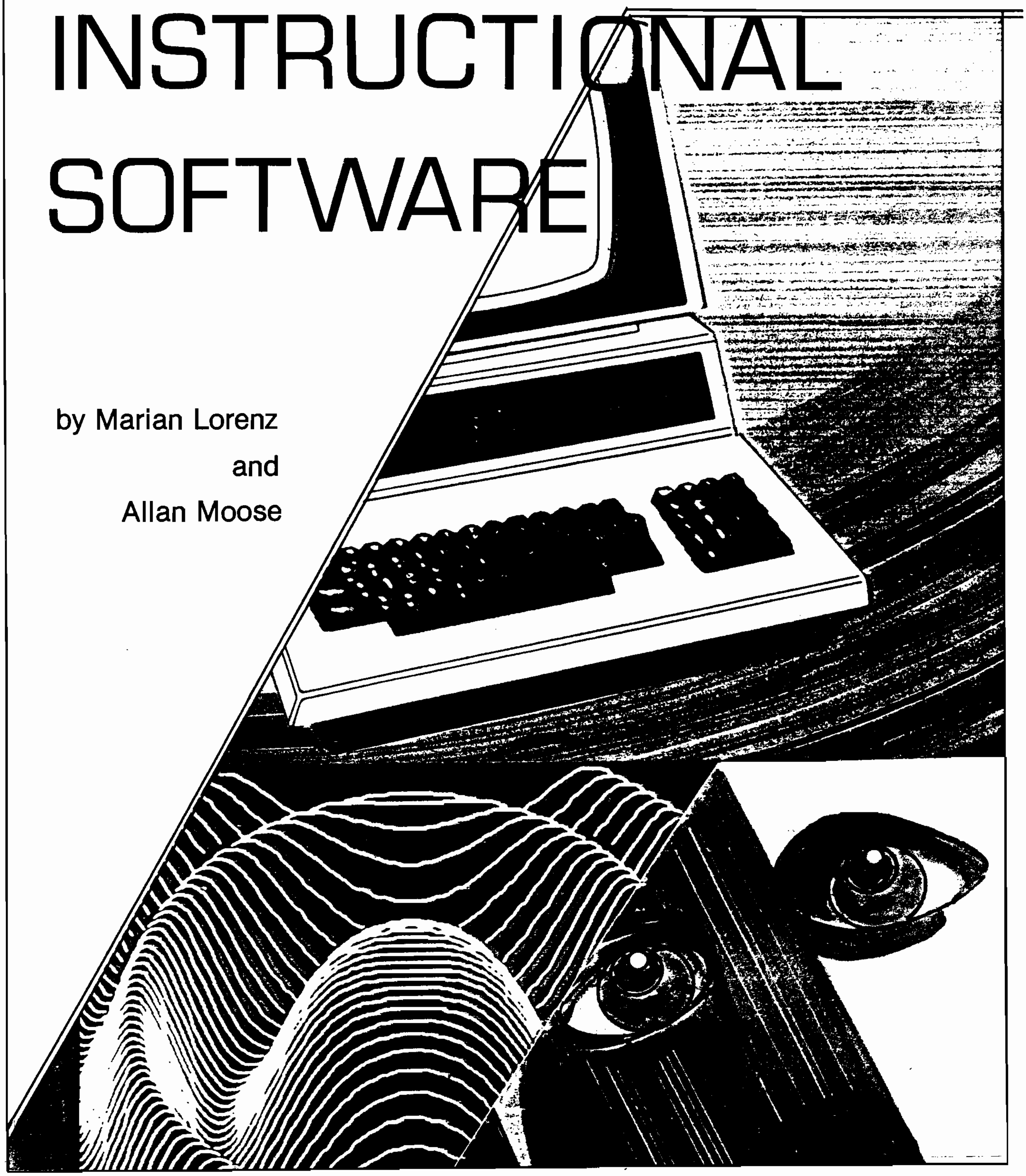
The ending of this article really is a beginning. In 1955 Dr. Rudolf Flesch wrote a book that took America by storm. It was entitled *Why Johnny Can't Read — and What You Can Do About It.* I quote from the preface, replacing the word "reading" with "computer literacy" and the word "book" with "article."

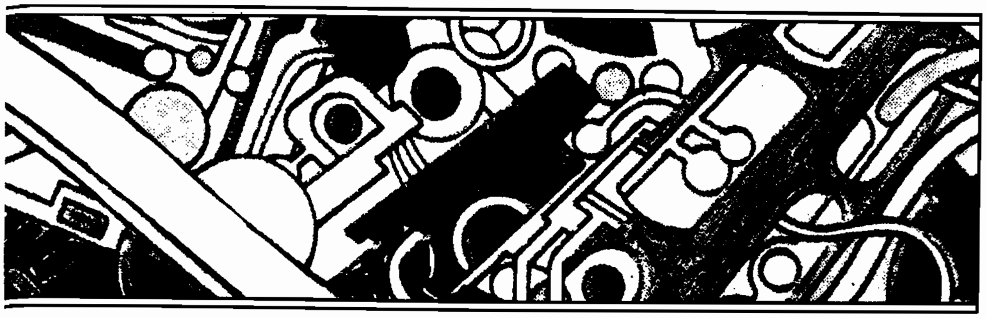
"Just as war is 'too serious a matter to be left to the generals,' so, I think, the teaching of computer literacy is too important to be left to the educators. This article, therefore, is not addressed to teachers... but to fathers and mothers."

MICRO

WRITING INSTRUCTIONAL SOFTWARE

by Marian Lorenz
and
Allan Moose





A discussion of the scope of applications for educational software and the various factors involved in designing a program.

In recent months many newspapers, magazines, and professional journals have carried articles that portray education in the United States in serious trouble. Many students do not get an adequate background in mathematics and science. In a large number of cases this is due to a lack of adequately trained teachers. Often students do not choose to take courses in these subjects because they are "dull," "boring," or just plain hard work. We believe that properly prepared educational computer programs can serve to help ameliorate some of the problems in our educational system.

The responsibility of providing effective education belongs to the educator. The microcomputer cannot take over that responsibility. However, the microcomputer can, if used appropriately, be an invaluable aid in the educational process. When properly programmed the computer can help the teacher make provisions for individualized instruction for each pupil. An effective program can provide an interactive learning experience that shows students that learning can be exciting and challenging. A computer can be programmed to adjust to the learning rates of individual students. Computers can provide immediate feedback, they are not judgemental, they don't get tired, and they can maintain a learner's attention. The student can be given more control over the learning process than occurs in a group lesson. In addition, there is privacy and freedom from peer pressure, which is important for remedial work.

Long-term research concerning the educational effectiveness of computers is necessarily limited. However, a number of findings suggest that students tend to learn faster by way of computer programs as compared to customary instructional methods; student retention rate is as good as, or superior to, customary instruction; a learner's attention can be maintained longer at a computer; it appears that using a computer in and of itself is motivating to the student; and computer drill/practice exercises are especially helpful for students who have problems with memorization.

Instructional software falls into five main categories. These are:

1. *Drill/Practice:* This type of program supplements previous instruction through reinforcement and practice. Drill and practice are an important part of the teaching-learning process. Traditional drill has turned a lot of children off to learning. However, computer drill can be organized to make practice fun. Computer drill programs can be used in the areas of math, spelling, history, geography, and other subjects that require memorization of facts or concepts. Since feedback is immediate, learning is enhanced and drill becomes more meaningful and productive.
2. *Tutorials:* These programs can be used as instructional units that teach rules and concepts. Tutorials are often used to re-teach work previously presented through conventional instructional methods but that has not been fully understood or mastered.
3. *Simulations/Problem Solving:* A simulation is a model of a situation in real life recreated by the computer. Programs of this nature enable the learner to have experience with environments that may be too expensive, dangerous, remote, or complex for classroom use. These types of programs allow the student to make use of known skills and concepts to develop new problem-solving strategies. Simulations teach students how to make decisions, think logically, and understand conceptual relationships. These types of exercises encourage students to understand problem situations and help them consider alternative designs and relations among the variables as opposed to applying some formula quickly simply to get the "right" answer.
4. *Games:* These programs allow students to apply skills and concepts in a game environment. They make provisions for learning rules and developing and revising strategies. There can be competition with the computer, with one's self and/or peers. In addition, games can be cooperative efforts — team games in which two or more students work to achieve a common goal, thereby fostering cooperation and positive peer interaction.
5. *Management:* Programs of this type are tools for the teacher or administrator. They can be used to schedule, test, keep records, and analyze student learning problems.

Keep in mind that the computer is only a tool for use in the educational process. Their effectiveness is going to depend on the quality of the software. Developing high-quality instructional software requires the merging of educational and technological expertise and a thorough understanding of programming techniques and the capabilities of the microcomputer. Furthermore, it requires a thorough understanding of educational principles and of the population for whom the program is being written. To a great extent the principles involved in writing high-quality educational software mirrors the principles of good programming.

In the remainder of this article we will present fundamental steps in preparing high-quality courseware. We have developed these ideas through a study of educational software reviews, attendance at conferences on microcomputers in education, conversations with teachers and parents, and our own

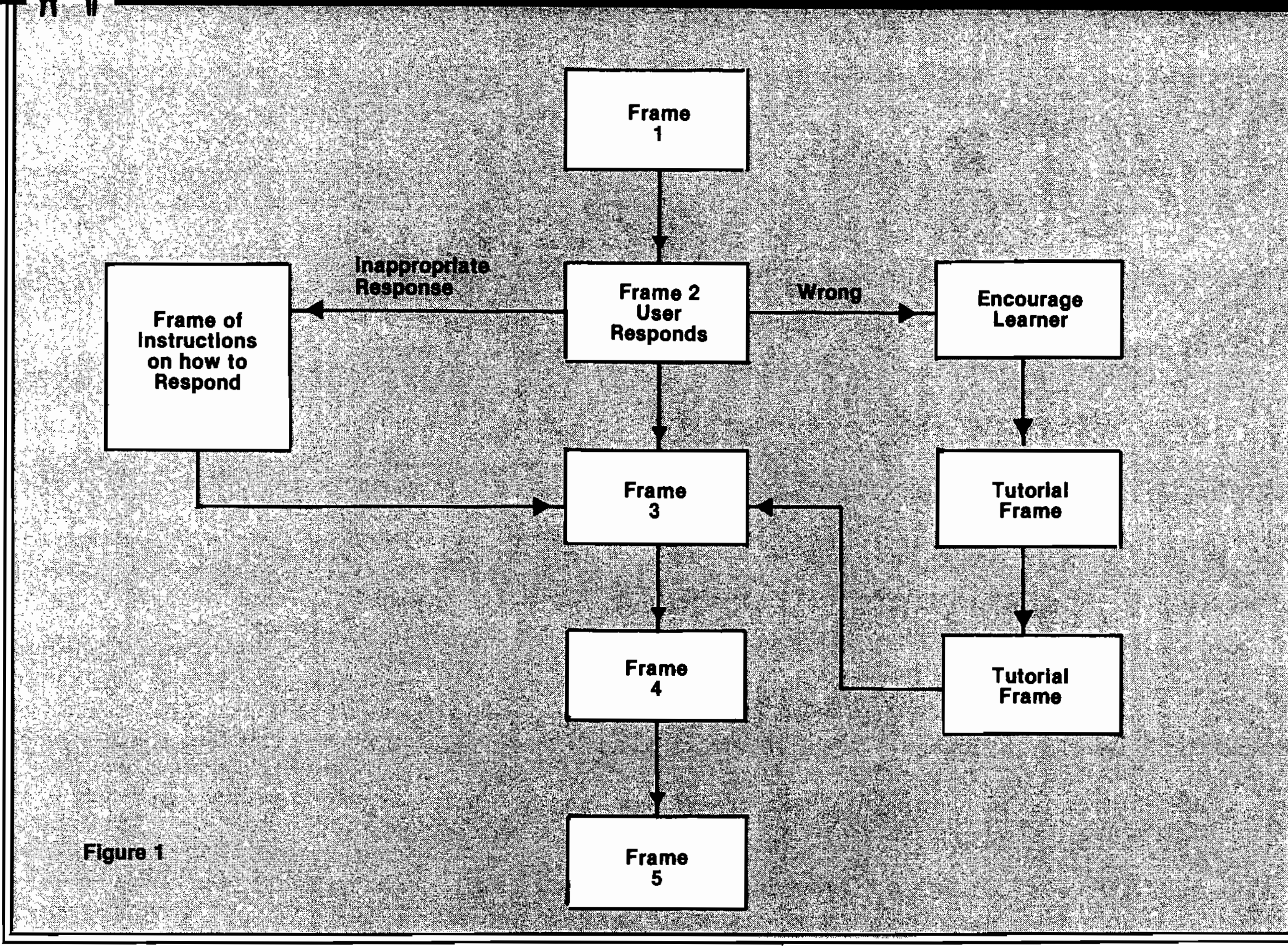


Figure 1

experiences in education.

The first step is to define your need or problem. This will help you decide what category of program will best deal with your subject matter.

The second step is to establish your goals and develop instructional objectives in terms of observable behavior. This will help you specify the content of the program and to determine what prerequisite skills, vocabulary, and concepts are necessary for the student to successfully learn the skills in your program. We cannot stress the importance of this step enough. If your primary experience is not in education (and even if it is) it would be worthwhile to consult one or more of the references we have listed at the end of this article.

The third step is to develop a program outline. This consists of a step-by-step guide indicating how each concept or skill will be developed and the order in which they will be presented. A teacher would call this a lesson plan. This outline will help you maintain consistency between the content and the program objectives. Indicate in the outline how you are going to determine if the pupil has learned the skills or concepts presented. Specify in your program outline what the correct responses are and how incorrect responses will be managed.

The fourth step is important in the development of effective instructional software: writing out the screen display, frame by frame. We suggest that you make use of sheets of graph paper, one for each frame. Each frame should be numbered and indicate what frame to go to if a response is correct, incorrect, or inappropriate. This method of program planning and writing makes editing easier. Here is where you start thinking about what the user will see and the subtle ways that this can affect learning. Some guidelines for planning screen displays are:

1. Design screen displays so they are easy to understand. For ease of reading, six lines of text, double spaced, serves the needs of most students. Break lines between phrases and avoid a crowded display. For young learners you might wish to use enlarged or colored text, if you have a computer that has this feature. Screen displays printed in capital letters are not as easy to read as "conventional" print with capitals and lower-case letters. A neutral color for the screen is also easier on the eyes than the typical blue screen.
2. Important information, new vocabulary, key words, and instructions can be highlighted by using inverse print or color. Use flashing words, letters, or phrases judiciously as they can be more distracting than attention getting.
3. Make sure you follow the established rules for punctuation, grammar, usage, and capitialization. Avoid spelling errors.
4. Have the learner respond frequently. Plan your program so that the user readily understands how to respond. For example, if you have a clock set at six-fifteen and the user must respond to "what time is it?", indicate clearly how to respond; i.e., ______hrs. and ______mins.; or ______mins, after ______. If the user does not understand

(Continued on page 48)

how to input his/her response it will cause unnecessary frustration.

5. Include frames that positively reinforce correct responses. Reinforcement should be variable and random, making use of behavior management principles. Do not make frames that follow incorrect responses punishing, nor exciting enough to encourage making errors.

6. Use branching to meet the individual needs of the learners. Branching reduces frustration for learners having problems as well as for pupils who have learned the material.

7. Graphics, sound, and color should be an integral part of the program. Use them to convey information, draw attention to important facts, or reinforce learning. Effective graphics are not simply an added attraction. Use sound judiciously. Too much can be wearing on the nerves. Do not use sound to indicate mistakes because it announces the student's errors to those around him/her.

A fifth step that you may want to use either at this point or in conjunction with your outline is to write a flowchart. This will help you maintain an overview of the total program. Since the computer will evaluate the pupil's responses, every conceivable response needs to be considered as far as this is possible. A flow chart of your frames will help you check this important aspect. Figure 1 is a simple example.

The sixth step is the programming portion of the software development. If you have followed the above steps this should be relatively straightforward.

The seventh step is editing the program. Go over the frames carefully, checking for spelling errors, content errors, grammar, syntax, and punctuation. Consider each frame in terms of readability and overall appearance. Monitor the "flow" of the program. At this point, consider the frame cards and the flowchart together to make sure every frame leads to another appropriate frame until the end is reached.

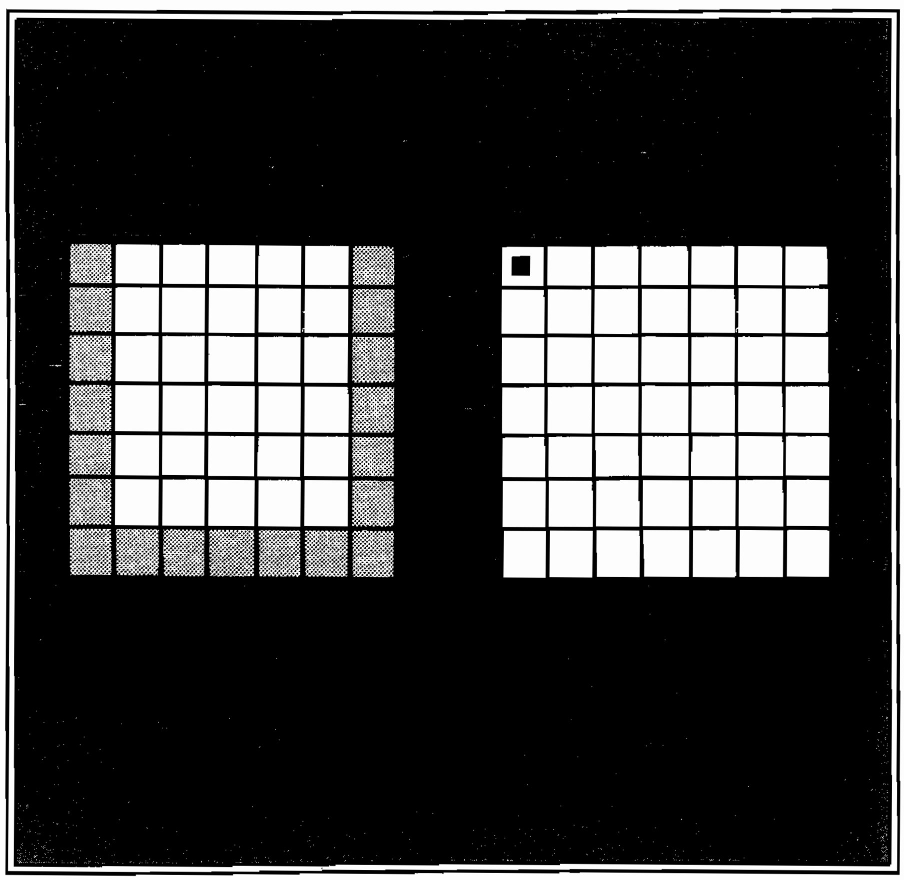
Step eight is the actual typing in of the program. If the program is lengthy, spread the typing over several sessions to avoid fatigue, which causes errors.

The ninth step is the process of "debugging." The first step in debugging is to print out the program and check for spelling, punctuation, and grammatical errors. If you do not have a printer available you will have to do this from the screen. Nothing ruins the impression that a user has of a program more than seeing typos. Next, run the program to make sure it will run to the end regardless of what branches are taken and/or what is typed in by the user. You want to insure that the program can be operated independently by the learner, that the program will not get hung up because of unexpected responses, or that the pupil cannot crash the program accidentally or deliberately. It is often helpful to have your program tested by at least one member of your target population.

The last step is the development of documentation or a user's guide. This is absolutely one of the more important steps and is unfortunately often neglected by many programmers. Even if you are a teacher or a parent writing for your own use, this is important. Perhaps there will be a time when you have many programs or when you want to swap programs with someone else. The guide should: 1. describe the program; 2. indicate for whom the program is intended; 3. specify what prerequisite skills are required; 4. list the program goals and instructional objectives; 5. note specialized vocabulary; 6. tell the user how to run the program and how to restore a crashed program; 7. provide information, suggestions, and/or materials to help the teacher and consequently the learner gain the most use from the program; 8. provide for evaluation of the extent to which the learner has achieved the program objectives.

We believe that computers have great educational potential for use both at home and in school. At computing conferences we have attended, the most frequent complaints heard are about the quality of educational software currently available. In this article we have attempted to furnish the reader with detailed suggestions on writing instructional software. Our suggestions reflect our belief that effective education takes careful thought and planning.

We shall illustrate some of the principles that have been presented with a spatial relations program we are developing. The need we identified is for a remedial program for children with learning disabilities. In particular, we have in mind children who have not developed adequate spatial relations concepts. Concurrently, the program is being designed for pre-academic children who are in the process of



developing spatial relations concepts. Finally, the program is being designed with parents and teachers in mind by structuring the code so it can be easily modified to meet the unique needs of a particular learner in accordance with the educational theory in the documentation. In a sense we had in mind the needs of both the child and adult.

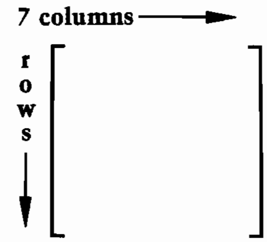
Therefore we stated our goals as: 1. to develop spatial relations concepts and, 2. to write easy-to-understand program code and documentation. The objectives are: 1. the child will organize a pattern or design as a unified whole; 2. the child will accurately reproduce a pattern or design; 3. the child will develop an organized approach to a task; 4. the child will develop the concept of directionality These objectives cannot be achieved through the use of the program alone. But with adequate documentation, a parent or teacher working with the child in conjunction with the computer can achieve them.

The screen displays consist of a pair of seven-by-seven grids. One, on the left of the screen, shows a design to be copied; the second, on the right, is the workspace where the child reproduces the shape. For example, one of the first displays is shown in figure 2.

Because color is useful in aiding visual discrimination we chose Atari Graphics Mode 7 which, with its four-color capability, allows the background, grid, and square to be in contrasting colors. The cursor in the right-hand grid is a player and is moved with the arrow keys. If the child wants to color in a particular square he/she presses the space bar. Since the program is designed to be remedial or tutorial, a square will fill in with color only if it is equivalent to a square on the master grid. Thus, the program is self-correcting and there is no need for branching in response to an inappropriate choice. The program is designed to respond only to the space bar and arrow keys. If the child tries to move the cursor out of the grid the cursor will not respond and a warning note sounds.

The frame-by-frame description of the screen display for this program is relatively simple and consists of choosing the sequence of patterns to be copied. One such sequence is shown in figure 3.

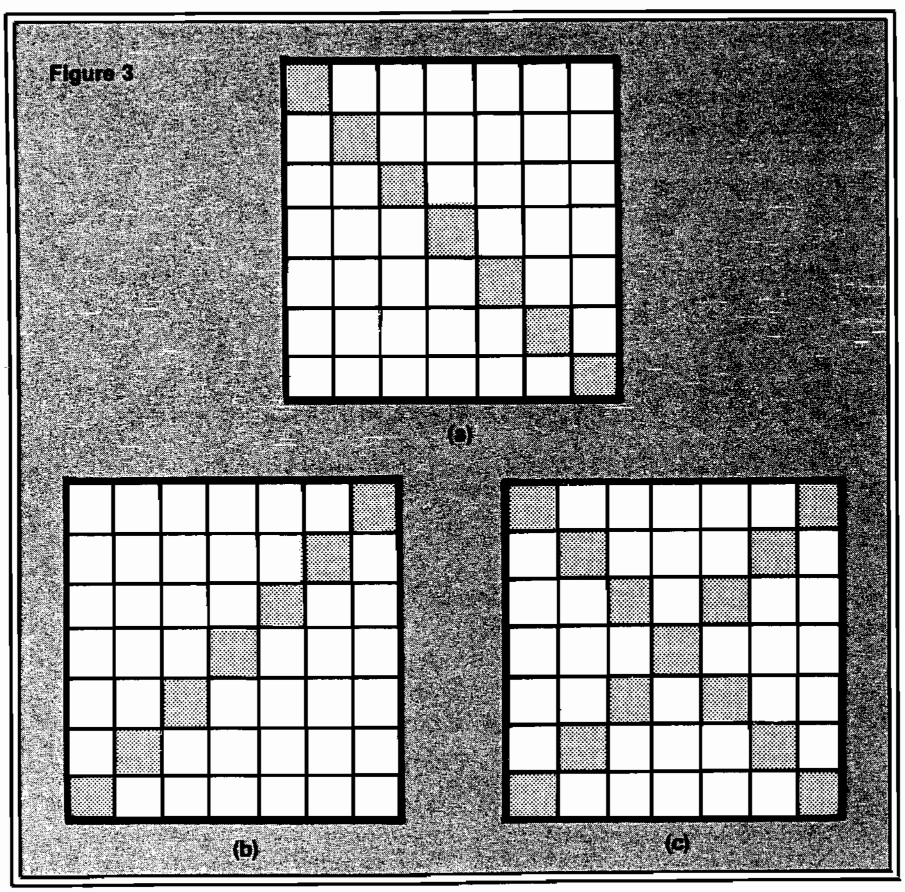
To make the program easy to modify, the patterns are made by combining basic line segments with a group of clearly defined subroutines. Each grid is represented in the program by a two-dimensional array. The array can be visualized as a 7-×-7 matrix that mimics the grid on the screen:



An individual square is filled in or left blank according to whether the corresponding matrix element is a 1 or 0, respectively. Thus, a row of 1's across the top of the matrix will become a straight line across the top of the grid on the screen. The ⊔ shape shown earlier is created by a segment of code such as:

```
320 REM ⊔ SHAPE SUBROUTINE
330 GOSUB VERT. BAR LEFT
340 GOSUB HORIZ. BAR BOTTOM
350 GOSUB VERT. BAR RIGHT
360 RETURN
```

followed by a routine that reads the matrix and fills in the appropriate squares. By changing lines 330 to 350 to call a different set of subroutines, a different figure can be drawn.



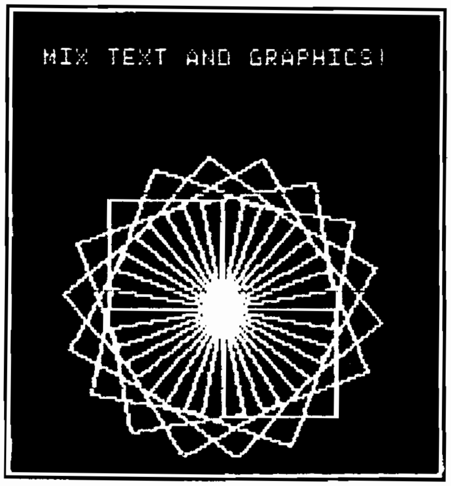
References

1. Bloom, B. *Taxonomy of Educational Objectives: Handbook I: Cognitive Domain.* David McKay, Inc., New York, 1956.
2. Krathwohl, D. R., Bloom, B. S., Masia, B. B., *Taxonomy of Educational Objectives: Handbook II: Affective Domain.* David Mckay, Inc., New York, 1964.
3. Mager, R. *Preparing Instructional Objectives.* Fearon Publishers Inc., California, 1962.
4. Siegal, E. and Siegel, R. *Creating Instructional Sequences.* Academic Therapy Press, California.
5. Steenbergen, F. *A Practical Guide to Writing Goals and Objectives.* Academic Therapy Press, California.

You may contact the authors at
156 Monell Ave., Islip, NY 11751.

MICRO

Hi – Res Characters for Logo



by Dan Weston

One of the weaknesses of the Apple II is the inability to combine text and graphics on the same screen. Many ingeneous utility programs are on the market to rectify this weakness. Unfortunately, none of them will work with any of the versions of Logo that run on the Apple.

You can get letters on the turtle graphics screen in Logo by using the turtle to draw them. This solution proves to be unworkable in most situations. Trying to define procedures to draw all twenty-six letters can fill your entire workspace. Even if you can fit them all in, the letters the turtle draws are thicker than normal letters, which may not be acceptable. With this in mind, I have written a group of Logo procedures that puts text on the hi-res screen without using turtle graphics.

Before I discuss how the Logo procedures work, I will explain briefly how characters are defined and how they are placed on the hi-res screen. If you want a more in-depth explanation, consult the references listed at the end of this article.

Characters are represented by a 7 x 8 grid of dots. Figure 1 shows samples of two characters mapped on to this grid. Each of the eight rows of a character can be represented by one byte. Actually, only the lowest seven bits of each byte are used to turn dots on or off. The eigth bit is used to control color. The bits are displayed from left to right, with bit 0 on the left and bit 6 on the right. The decimal value of each bit is marked above each column in figure 1. The decimal value of the byte for each row is listed to the right of the row in figure 1. You can see that a character may be defined by a series of eight bytes, corresponding to the eight rows of the grid.

Listing 1 contains the byte definitions for fifty-nine common characters. Each character is represented as a list of eight numbers. The name of each list corresponds to the character it defines; "C.! represents the exclamation point, and so on. The lists are kept this way to allow for easy modification. Listing 1 should be entered into the logo workspace along with the procedures that appear later in this article. (*Editor's Note:* It is necessary to type '.GCOLL' once in a while to avoid overflowing workspace.)

Once you have defined the characters, you need to know how to put them on the hi-res screen. The hi-res screen is a direct representation of bits in memory between address 8192 and 16384. Seven bits of each byte are used to turn dots on or off. The screen is forty bytes wide, giving 280 possible dots horizontally. The screen is twenty-four characters high, with eight bytes per character giving 192 possible dots vertically.

Finding the actual memory address for any particular byte on the screen can be tricky. Listing 2 is a table of beginning addresses for the twenty-four character rows on the left edge of the screen. Subsequent addresses as you move across the screen horizontally are found by adding the column number (0-39) to the beginning address. Each character row is made up of eight rows of dots. For any one character position, the address for the first byte is found by adding the column number to the beginning address of the row. The addresses for the seven subsequent bytes are found by repeatedly adding 1024 to the address of the first byte. An example should clarify this.

Example 1

Here is how you would put the letter "A" into character row 5, column 7:

1. Look up the beginning address of row 5 from listing 2: (8832)
2. Add the column number to this: (8832 + 7 = 8839)
3. Look up the first byte of "A" from figure 1 or listing 1 : (8)
4. Put that value into address 8839: .DEPOSIT 8839 8
5. Add 1024 to 8839: =9863
6. Get the next byte for "A" : (20)
7. Put it in address 9863 : .DEPOSIT 9863 20
8. Add 1024 to 9863: = 10887
9. Get the next byte for "A" : (34)
10. Put it in address 10887 : .DEPOSIT 10887 34

Continue this pattern until all eight bytes for "A" have been put into memory. Figure 2 shows the result of this operation. The addresses are shown on the left of each row, and the byte values on the right. This is essentially the process that the Logo procedures listed below will use to put text on the hi-res screen. Note that this method of adding 1024 to each address will only work if the twenty-four beginning locations given in listing 2 are used. This makes the procedures less flexible, but infinitely simpler.

Listings 3 and 4 contain the procedures that will put text on the hi-res screen in Logo. These procedures use the normal text cursor positioning primitives built into Logo to guide placement of the text on the hi-res screen. The user should position the cursor as if to put text on the text screen before calling the hi-res procedures. If the turtle screen is currently being viewed, then the cursor will not be visible to the user, but will still act as a stalking horse for the hi-res routines.

HPRINT is the top-level procedure that will be most often called by the user. It may take a word or a list as input. HPRINT tests its input to see if it is a list or a word and routes it to the appropriate subprocedure for processing. Lists are passed to PICKWORD where the component words are picked out and passed to PUTWORD. Words input to HPRINT are passed directly to PUTWORD.

PUTWORD first checks to see if its input is the empty word. If it isn't, PUTWORD calls PUTCHAR with the correct starting address and the list of eight bytes for the first character of the word.

The address is determined by taking the current cursor column, output by COLUMN, and adding it to the beginning address of the current cursor row, output by ROWADDRESS. ROWADDRESS uses the output of ROW to look up the address from the values in listing 2, much as we did in step 1 of example 1 above.

The list of eight byes for the first character of the word are determined by the output of GETBITS. This procedure looks up values from listing 1 by combining its input with "C. ROWADDRESS and GETBITS are both lookup procedures and work in much the same way.

PUTCHAR is where the actual bits are placed into screen memory. PUTCHAR starts with the address for the top row of a character and a list of the eight bytes needed to define that character. It then places the first byte into memory at the starting address. PUTCHAR then adds 1024 (defined as "NEXTLINE in listing 2) to the address and calls itself recursively with all but the first byte of the list. This will continue until all eight bytes have been put into memory. You should see the similarity to example 1 here. PUTCHAR takes advantage of logo's ability to modify inputs to procedures without affecting the value of global variables.

You can see the same fundamental structure in PICKWORD, PUTWORD, and PUTCHAR. All three operate on the first element of their input, and then cycle recursively with the BUTFIRST of that input until the input is empty. This technique has wide application in logo programming.

PUTCHAR was called by PUTWORD to place the first character of a word on the screen. When PUTCHAR finishes displaying that character, control is passed back to PUTWORD. PUTWORD then calls MOVECURSOR to move the cursor to the next character position.

MOVECURSOR moves the unseen text cursor so that ROW and COLUMN will continue to give appropriate values. MOVECURSOR will call RETURN if the right edge of the screen has been reached and RETURN will handle the wrapping to the next character line. RETURN will also route text to the upper left corner of the screen if it is called from the lower right corner. There is no provision for scrolling.

Once PUTWORD has processed the first character of its input, it calls itself recursively with the BUTFIRST of its input. It will do this until all the characters have been placed on the screen. Because Logo generally treats a space as a deliminator rather than a character, PUTWORD calls MOVECURSOR one extra time just before it stops to place a space after each word that it processes. Most of the time this will be fine, but you may find that you want to remove this step in PUTWORD.

HTEST is a sample procedure to show how HPRINT can be used. Its first step, which is optional, is to clear the hi-res screen. Then it places the text cursor at the upper left corner of the screen. HPRINT is then called with all the characters defined in listing 1 as input. This is a good way to see if the character definitions are to your liking. You might want to customize some of the characters. Although these procedures are too slow to do effective character animation, you might find some use for non-standard characters.

If you find that you are not using all the characters that have been defined, it will be to your advantage to erase the unused character definitions from your workspace, freeing up extra nodes for other procedures that will use HPRINT. These procedures are intended mainly for labeling pictures and graphs. They do not intercept normal keyboard input and route it to the hi-res screen and they do not scroll. They are, as one high school basketball coach once said succinctly of his team, "big and dumb and slow," however I think you will find them useful, and also instructive as to what can be done with Logo beyond turtle graphics.

1. Pelczarski, Mark, "Graphically Speaking," *Softalk*, October, 1982, pg. 240-242.
2. Wagner, Roger, "Assembly Lines," *Softalk*, April 1983, pg. 247-254.
3. Wagner, Roger, "Assembly Lines," *Softalk*, May 1983, pg. 185-190.
4. *Apple II Reference Manual*, Apple Computer Co., pg. 18-19, 21.

Dan Weston teaches a self-contained eighth grade in Brooks, Oregon. He may be contacted at 195 23rd NE, Salem, OR 97301.

Listing 1

```
MAKE "C.! [8 8 8 8 8 0 8 0]
MAKE "C." [20 20 0 0 0 0 0 0]
MAKE "C.# [0 0 20 62 20 62 20 0]
MAKE "C.$ [28 42 10 28 40 42 28 0]
MAKE "C.% [0 36 18 8 36 18 0 0]
MAKE "C.& [4 10 10 4 42 18 44 0]
MAKE "C.' [16 16 0 0 0 0 0 0]
MAKE "C.( [8 4 2 2 2 4 8 0]
MAKE "C.) [8 16 32 32 32 16 8 0]
MAKE "C.* [0 42 28 62 28 42 0 0]
MAKE "C.+ [0 8 8 62 8 8 0 0]
MAKE "C., [0 0 0 0 0 8 8 4]
MAKE "C.- [0 0 0 62 0 0 0 0]
MAKE "C.. [0 0 0 0 0 0 8 0]
MAKE "C./ [0 32 16 8 4 2 0 0]
MAKE "C.0 [28 34 50 42 38 34 28 0]
MAKE "C.1 [8 12 8 8 8 8 28 0]
MAKE "C.2 [28 34 32 24 4 2 62 0]
MAKE "C.3 [28 34 32 28 32 34 28 0]
MAKE "C.4 [34 34 34 62 32 32 32 0]
MAKE "C.5 [62 2 2 30 32 32 30 0]
MAKE "C.6 [28 34 2 30 34 34 28 0]
MAKE "C.7 [62 32 32 16 8 4 2 0]
MAKE "C.8 [28 34 34 28 34 34 28 0]
MAKE "C.9 [28 34 34 60 32 34 28 0]
MAKE "C.: [0 0 8 0 8 0 0 0]
MAKE "C.; [0 0 8 0 8 8 4 0]
MAKE "C.< [0 0 16 8 4 8 16 0]
MAKE "C.= [0 0 0 28 0 28 0 0]
MAKE "C.> [0 0 4 8 16 8 4 0]
MAKE "C.? [28 34 32 24 8 0 8 0]
MAKE "C.@ [28 34 42 58 26 2 60 0]
MAKE "C.A [8 20 34 34 62 34 34 0]
MAKE "C.B [30 34 34 62 34 34 30 0]
MAKE "C.C [28 34 2 2 2 34 28 0]
MAKE "C.D [30 34 34 34 34 34 30 0]
MAKE "C.E [62 2 2 30 2 2 62 0]
MAKE "C.F [62 2 2 30 2 2 2 0]
MAKE "C.G [28 34 2 50 34 34 60 0]
MAKE "C.H [34 34 34 62 34 34 34 0]
MAKE "C.I [62 8 8 8 8 8 62 0]
MAKE "C.J [32 32 32 32 32 34 28 0]
MAKE "C.K [34 18 10 6 10 18 34 0]
MAKE "C.L [2 2 2 2 2 2 62 0]
MAKE "C.M [34 34 54 42 42 34 34 0]
MAKE "C.N [34 34 38 42 50 34 34 0]
MAKE "C.O [28 34 34 34 34 34 28 0]
MAKE "C.P [30 34 34 30 2 2 2 0]
MAKE "C.Q [28 34 34 34 42 18 44 0]
MAKE "C.R [30 34 34 30 10 18 34 0]
MAKE "C.S [28 34 2 28 32 34 28 0]
MAKE "C.T [62 8 8 8 8 8 8 0]
MAKE "C.U [34 34 34 34 34 34 28 0]
MAKE "C.V [34 34 34 34 34 20 8 0]
MAKE "C.W [34 34 42 42 42 42 20 0]
MAKE "C.X [34 34 20 8 20 34 34 0]
MAKE "C.Y [34 34 34 20 8 8 8 0]
MAKE "C.Z [62 32 16 8 4 2 62 0]
MAKE "C.\ [0 2 4 8 16 31 0 0]
```

Listing 2

```
MAKE "R.23 (9168)
MAKE "R.22 (9040)
MAKE "R.21 (8912)
MAKE "R.20 (8784)
MAKE "R.19 (8656)
MAKE "R.18 (8528)
MAKE "R.17 (8400)
MAKE "R.16 (8272)
MAKE "R.15 (9128)
MAKE "R.14 (9000)
MAKE "R.13 (8872)
MAKE "R.12 (8744)
MAKE "R.11 (8616)
MAKE "R.10 (8488)
MAKE "R.9 (8360)
MAKE "R.8 (8232)
MAKE "R.7 (9088)
MAKE "R.6 (8960)
MAKE "R.5 (8832)
MAKE "R.4 (8704)
MAKE "R.3 (8576)
MAKE "R.2 (8448)
MAKE "R.1 (8320)
MAKE "R.0 (8192)
MAKE "NEXTLINE (1024)
```

Listing 3

```
TO PICKWORD :LIST
 IF :LIST = [] THEN STOP
 IF LIST? FIRST :LIST THEN PICKWORD FIRST :LIST!
    ELSE PUTWORD FIRST :LIST
 PICKWORD BF :LIST
END

TO PUTWORD :WORD
 IF :WORD = " THEN MOVECURSOR STOP
 PUTCHAR ( ROWADDRESS + COLUMN ) GETBITS!
    FIRST :WORD
 MOVECURSOR PUTWORD BF :WORD
END

TO HPRINT :INPUT
 IF LIST? :INPUT THEN PICKWORD :INPUT!
    ELSE PUTWORD :INPUT
END

TO HTEST
 DRAW
 CURSOR 0 0
 HPRINT [ABCDEFGHIJKLMNOPQRSTUVWXYZ,.;!
    -:1234567890!"#$%&'()*=+?><@]
END

TO MOVECURSOR
 TEST COLUMN < 39
 IFT CURSOR ( COLUMN + 1 ) ROW
 IFF RETURN
END

TO PUTCHAR :ADDRESS :CHARBYTES
 IF :CHARBYTES = [] THEN STOP
 .DEPOSIT :ADDRESS FIRST :CHARBYTES
 PUTCHAR :ADDRESS + :NEXTLINE BF :CHARBYTES
END

TO ROWADDRESS
 OP THING WORD "R. ROW
END

TO COLUMN
 OP .EXAMINE 36
END

TO GETBITS :CHAR
 OP THING WORD "C. :CHAR
END

TO ROW
 OP .EXAMINE 37
END

TO RETURN
 TEST ROW = 23
 IFT CURSOR 0 0
 IFF CURSOR 0 ( ROW + 1 )
END
```

Listing 4

```
TO HTEST
CS SETCURSOR [0 0]
HPRINT [ABCDEFGHIJKLMNOPQRSTUVWXYZ 12345
 67890:*-=;+/?.>,<!"#$%'()]
END
TO HPRINT :INPUT
IF LISTP :INPUT [PICKWORD :INPUT]!
 [PUTWORD :INPUT]
END

TO PICKWORD :LIST
IF EMPTYP :LIST [STOP]
IF LISTP FIRST :LIST [PICKWORD FIRST :LIST]!
 [PUTWORD FIRST :LIST]
PICKWORD BF :LIST
END

TO PUTWORD :WORD
IF EMPTYP :WORD [MOVECURSOR STOP]
PUTCHAR (ROWADDRESS + COLUMN) GETBITS FIRST :WORD
MOVECURSOR
PUTWORD BF :WORD
END

TO MOVECURSOR
TEST COLUMN > 39
IFT [SETCURSOR LIST (COLUMN + 1) ROW
IFF [RETURN]
END

TO PUTCHAR :ADDRESS :CHARBYTES
IF EMPTYP :CHARBYTES [STOP]
.DEPOSIT :ADDRESS FIRST :CHARBYTES
PUTCHAR :ADDRESS + :NEXTLINE BF :CHARBYTES
END

TO ROWADDRESS
OP THING WORD "R. ROW
END

TO COLUMN
OP FIRST CURSOR
END

TO GETBITS :CHAR
OP THING WORD "C. CHAR
END

TO ROW
OP FIRST BF CURSOR
END

TO RETURN
TEST ROW = 23
IFT [SETCURSOR [0 0]]
IFF [SETCURSOR LIST 0 (ROW+1)]
END
```

MICRO™

Figure 1

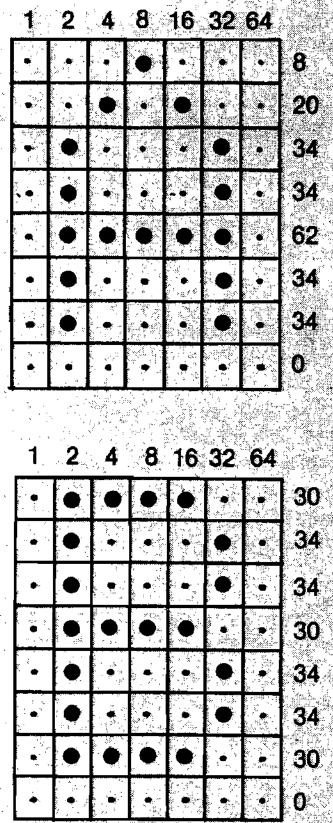
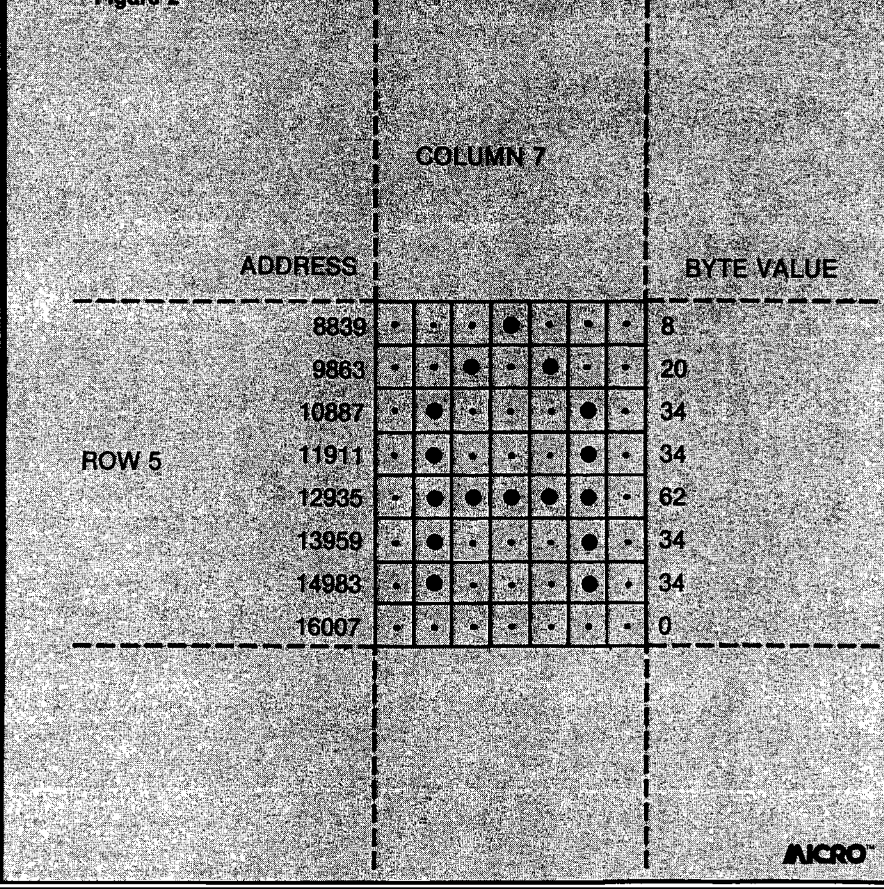


Figure 2



Educational Software Resource List

Arizona

Southwest Ed Psych Services, Inc.
P.O. Box 1870
Phoenix, AZ 85001

Evans Newton Inc.
7745 East Redfield Rd.
Scottsdale, AZ 85260

California

Advanced Learning Technology, Inc.
4370 Alpine Rd.
Portola Valley, CA

Anaheim Publishing Co.
2632 Saturn St.
Brea, CA 92621

ATARI Program Exchange
Atari, Inc.
60 East Plumerai Dr.
San Jose, CA 95150

Children's Television Workshop
20525 Mariani Ave.
Cupertino, CA

CMA/Micro Computer Division
55722 Santa Fe Trail
Yucca Valley, CA 92284

Computer Curriculum Corp.
P.O. Box 10080
Palo Alto, CA 94303

Control Data Publishing
4455 Eastgate Mall
San Diego, CA 92121

Educational Software Inc.
Soquel, CA

Educational Systems Software
23720 El Toro Rd. Suite C
P.O. Box E
El Toro, CA 92630

Edusoft
P.O. Box 25606
Berkeley, CA 94702

Edu-Ware Services, Inc.
28035 Dorothy Dr.
Agoura, CA 91301

Fullmer Associates
1132 Via Jose
San Jose, CA 95120

The Learning Co.
4370 Alpine Rd.
Portola Valley, CA 94025

Lighting Software
P.O. Box 11725
Palo Alto, CA 94301

Opportunities for Learning
8950 Lurline Ave.
Chatsworth, CA 91311

Storybooks of the Future
P.O. Box 4447
Santa Clara, CA 95054

Teachercenter
2725 Sand Hill Rd.
Menlo Park, CA 94025

Telephone Software Connection, Inc.
P.O. Box 6548
Torrance, CA 90504

Skill corp. Software, Inc.
1711 McGraw Ave.
Irvine, CA 92714

Colorado

Data Transform Inc.
616 Washington, Suite 106
Denver, CO 80203

Learning Systems Ltd.
P.O. Box 9046
Fort Collins, CO 80525

Connecticut

K8 Software
P.O. Box 248
Canton, CT 06019

MARCK
280 Linden Ave.
Branford, CT 06405

Queue
5 Chapel Hill Dr.
Fairfield, CT 06432

Florida

Aquaris Publishers, Inc.
P.O. Box 128
Indian Rocks Beach, FL 33535

Illinois

Borg-Warner Educational System
600 West University Drive
Alrington, IL 60004

Educational Teaching Aids
159 W. Kinzie St.
Chicago, IL 60610

Electronic Courseware System, Inc.
Box 2374, Station A
Champaign, IL 61820

Encyclopedia Britannica
Educational Co.
425 North Michigan Ave.
Chicago, IL 60611

Follett Library Book Co.
4506 Northwest Highway
Crystal Lake, IL 60014

Jadee Enterprises
1799 Meadowlake
Charleston, IL 61920

Micrograms. Inc.
P.O. Box 2146
Loves Park, IL 61130

Midwest Visual Equipment Co.
6500 North Hamlin
Chicago, IL 60645

Science Research Associates, Inc.
155 North Wacker Dr.
Chicago, IL 60606

Scott Foresman and Company
1900 E. Lake Ave.
Glenview, IL 60025

Indiana

Meka Publishing Co.
9120 Galaxie
Indianapolis, IN 46227

Iowa

Conduit
100 Lindquist Center
P.O. Box 338
University of Iowa
Iowa City, IA 52244

Kentucky

Van Nostrand Reinhold, Co., Inc.
7625 Empire Dr.
Florence, KY 41042

Louisiana

Cross Educational Software
1802 N. Trenton, Box 1536
Ruston, LA 71270

Maryland

Capital Systems Group, Inc.
11301 Rockville Pike
Kensington, MD 20895

Computer Science Press, Inc.
11 Taft Court, Dept. CMC 383
Rockville, MD 20850

Media Materials, Inc.
2936 Reminton Ave.
Baltimore, MD 21211

Massachusetts

Addison-Wesley Publishing Co.
Computer Software and Applications
Reading, MA 01867

The Answer
Houghton Mifflin
One Beacon St.
Boston, MA 02107

EduTech, Inc.
634 Commonwealth Ave.
Newton Centre, MA 02159

J. L. Hammett Co.
Box 545
Braintree, MA 02184

Learning Tools, Inc.
686 Massachusetts Ave.
Cambridge, MA 02139

Memory Bank
P.O. Box 76
Newton, MA 02168

Milton Bradley Educational Div.
443 Shaker Rd.
East Longmeadow, MA 01028

Spinnaker Software Corp.
215 First St.
Cambridge, MA 02142

Terrapin, Inc.
380 Green St.
Cambridge, MA 02139

Tycom Associates
68 Velma Ave.
Pittsfield, MA 01201

Window, Inc.
469 Pleasant St.
Watertown, MA 02172

Michigan

Alternate Source
704 Pennsylvania St.
Lansing, MI 48906

Comm Data Computer House Inc.
P.O. Box 325
Milford, MI 48092

Compu-tations, Inc.
P.O. Box 325
Troy, MI 48099

Hartley Courseware Inc.
Dept. CND, Box 431
Dimondale, MI 48821

MCE Educational Programs
Interpretive Education, Inc.
157 South Kalamazoo Mall
Kalamazoo, MI 49007

Sensible Software, Inc.
6619 Perham Dr.
West Bloomfield, MI 48033

T.H.E.S.I.S.
P.O. Box 147
Garden City, MI 48135

Minnesota

Computer Courseware Services
300 York Ave.
St. Paul, MN 55101

T.I.E.S.
1925 West County Rd. BZ
Saint Paul, MN 55113

Missouri

Data Security Concepts
P.O. Box 31044
Des Peres, MO 63131

Milliken Publishing Co.
1100 Research Blvd.
St. Louis, MO 63132

New Hampshire

Applied Educational Systems
RSD 2, Box 213
Dunbarton, NH 03301

Dolphin Computer-Based Instruction
TSC, A Houghton Mifflin Co.
Box 683
Hanover, NH 03755

Entelek
P.O. Box 1303
Portsmouth, NH 03801

New Jersey

Academic Software
c/o Software City
22 East Quackenbush Ave.
Dumont, NJ 07628

Educational Micro Systems, Inc.
P.O. Box 471
Chester, NJ 07930

Hayden Book, Co.
50 Essex St.
Rochelle Park, NJ 07662

New Mexico

The Programmers, Inc.
P.O. Box 1207-211 Cruz Alta
Taos, NM 87571

New York

Computer Motivated Learning Lab
Random House School Div.
201 East 50th St.
New York, NY 10022

DynaComp, Inc.
1427 Montroe Ave.
Rochester, NY 14618

Educational Audio Visual Inc.
Pleasantville, NY 10570

Gessler Publishing Co.
900 Broadway, Suite 10A
New York, NY 10003

Harcourt Brace Jovanovich
School Dept.
757 Thrid Ave.
New York, NY 10017

Harper and Row/CONDUIT
110 East 53rd St.
College Div. Suite 3D
New York, NY 10022

Holt, Rinehart, and Winston
383 Madison Ave.
New York, NY 10017

Instructional/Communications Tech.
10 Stepar Place
Huntington Station, NY 11746

J & S Software
140 Reid Ave.
Port Washington, NY 11050

K-12 Micromedia
P.O. Box 17
Valley Cottage, NY 10989

Kensington Microware
300 E. 54th St.
New York, NY 10022

Krell Software
1320 Stony Brook Rd. Suite 219
Stony Brook, NY 11790

McGraw-Hill School Div.
1221 Ave. of the Americas
New York, NY 10020

Mercer Systems, Inc.
87 Scooter Lane
Hicksville, NY 11801

Microcomputers Corp.
P.O. Box 8
Armonk, NY 10504

Microcomputer Workshops
103 Puritan Dr.
Port Chester, NY 10573

The Micro Center
P.O. Box 6
Pleasantville, NY 10570

Random House School Div.
201 East 50th St.
New York, NY 10022

Reader's Digest Services, Inc.
Educational Div.
Pleasantville, NY 10570

Right On Programs
Div. Computeam Inc.
P.O. Box 977
Huntington, NY 11743

Sunburst Communications
39 Washington Ave. Box 40
Pleasantville, NY 10570

Tara Ltd.
P.O. Box 118
Selden, NY 11784

Teach Yourself Computer Software
2128 West Jefferson Rd.
Pittsford, NY 14534

North Carolina

SOSS
3408 Dover Rd.
Durham, NC 27707

Ohio

South-Weston Publishing Co.
5101 Madison Rd.
Cincinnati, OH 45227

Oklahoma

Dorsett Educational Systems, Inc.
Box 1226
Norman, OK 73070

Oregon

Avant-Garde Creations
P.O. Box 30160
Eugene, OR 97403

Dilithium Software
P.O. Box 606
Beaverton, OR 97075

Quality Educational Designs
P.O. Box 12486
Portland, OR 97212

Quicksoft
P.O. Box 10854
Eugene, OR 97440

Pennsylvania

Victory Software Inc.
7 Valley Brook Rd.
Paoli, PA 19301

Scandura Training Systems, Inc.
1249 Greentree Lane
Narberth, PA 19072

Rhode Island

Jamestown Publishers
P.O. Box 6743
Providence, RI 02940

Tennessee

Educational Computing Systems, Inc.
106 Fairbanks
Oak Ridge, TN 37830

Micro Learningware
P.O. Box 2134
N. Mankto, TN 56001

Texas

Armadillo International Software
P.O. Box 7661
Austin, TX 78712

Color Software Services
P.O. Box 1723, Dept. E
Greenville, TX 75401

Developmental Learning Materials
One DLM Park
Allen, TX 75002

Gamco Industries, Inc.
Box 1911
Big Spring, TX 79720

George Earl
1302 South General McMullen
San Antonio, TX 78237

Powell Associates, Inc.
3724 Jefferson Suite 205
Austin, TX 78731

Sterling Swift Publishing Co.
1600 Fortview Rd.
Austin, TX 78704

Texas Instruments
P.O. Box 10508
Mail Station 5849
Lubbock, TX 79408

Virginia

Microsoftware Services
P.O. Box 776
Harrisonburg, VA 22801

Reston Publishing
11480 Sunset Hills Rd.
Reston, VA 22090

SEI
P.O. Box 7266-CD
Hampton, VA 23666

Washington

Attn: Publicity
Bertamax, Inc.
101 Nickerson, Suite 202
Seattle, WA 98109

Expert Systems, Inc. (ESI)
P.O. Box 9
Redmond, WA 98052

Micro-80 Inc.
2665 Busby Court Rd.
Oak Harbor, WA 98277

MICRO Mountain
14617 N.E. 169th St.
Woodinville, WA 98072

Wisconsin

Magic Lantern Computers
406 South Park Street
Madison, WI 53715

Educational Resource Books/Directories

1983 Classroom Computer News Directory of Educational Computing Resources
Intentional Educations, Inc.
341 Mt. Auburn St.
Watertown, MA 02172

Selected Microcomputer Software
Opportunities for Learning, Inc.
8950 Lurline Ave.
Chatsworth, CA 91311

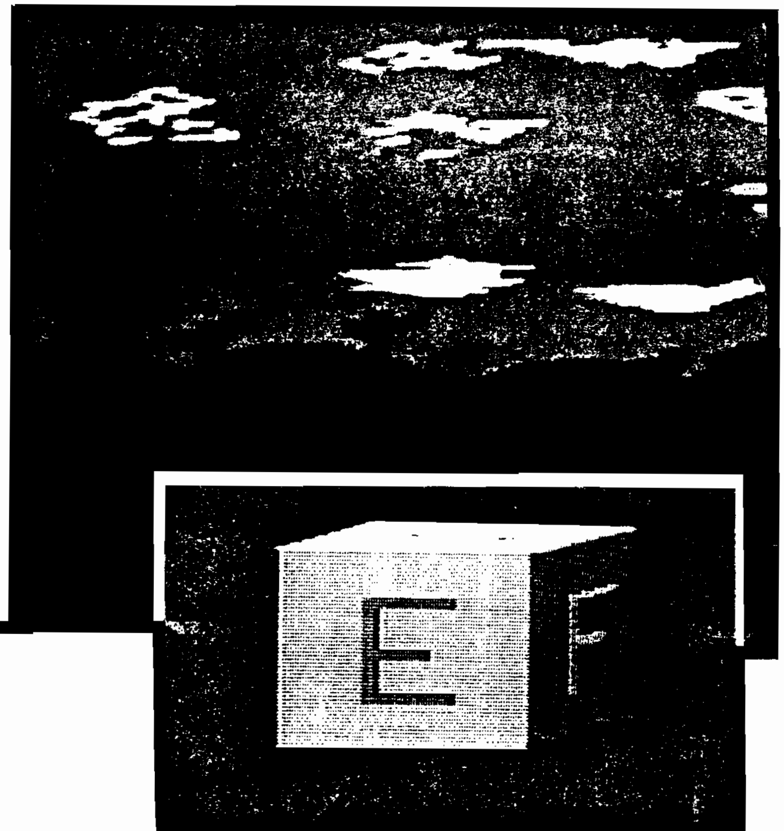
The Software Catalog, Microcomputers and Minicomputers
Elsevier Science Publishing Co., Inc.
52 Vanderbilt Ave.
New York, NY 10017

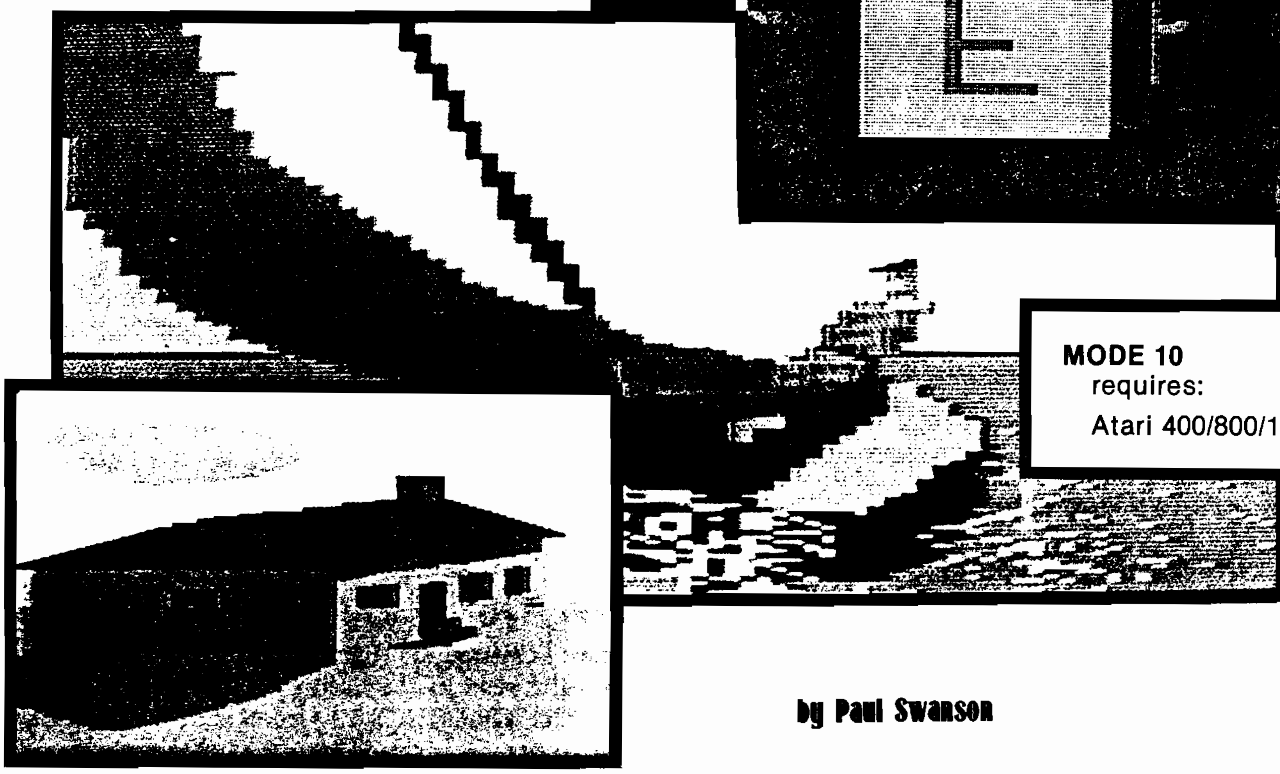
PC Clearinghouse Software Directory
PC Clearinghouse, Inc. Publishers
11781 Lee Jackson Highway
Fairfax, VA 22033

Swift's Educational Software Directory
Sterling Swift Publishing Co.
7901 South 1-35
Austin, TX 78744

MICRO™

Mode 10 Atari Painting Program Part III





MODE 10
requires:
Atari 400/800/1200

by Paul Swanson

The first two parts of this article explained how to operate the mode 10 painter program and how to make alterations easily. This part explains how to add routines to generate several shapes (given two screen locations).

Adding More Functions

The first thing to consider when adding new routines is the operator's point of view about how the routines should work. Operation must be kept simple and easy to remember. Many shapes can be defined very simply by defining two points. For example, a circle can be defined by one point at the center and one point on its circumference. A rectangle can be defined by the two points at opposite corners. A line is another shape that shouldn't be ignored, and two points, by definition of a line, determine a line.

The method of operation that is used in the alteration described here requires positioning the cursor in the two locations, in the correct order, then pressing two keys — one to institute shape drawing and the other to select which shape to draw. The shapes include a circle, a rectangle, and a line. The circle and the rectangle may be outlined or filled.

Program Alteration

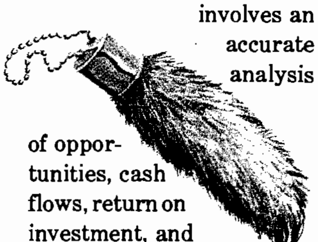
The first statements to alter are the ones that define the Help screen. Line 360 in the original version had one of the fill letters in it; last month's alteration removed that when the arrow keys were implemented for fill. Therefore, there is now an available blank space on the Help screen. A command for drawing shapes can be listed there. (Refer to the new listing of the mode 10 program accompanying this article.) The letter S is selected for implementing the shape-drawing routine.

(Continued on page 60)

How to become a real estate millionaire

YOU NEED MORE THAN LUCK

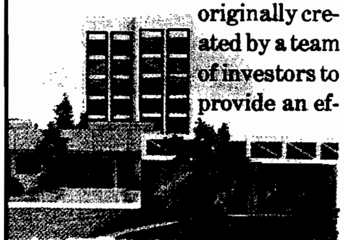
No matter what the economic situation . . . or the time of year . . . there's money to be made in Real Estate. More high-living millionaires owe their wealth to investments in Real Estate than anything else. But it takes more than luck. More than being in the right place at the right time. The key to success involves an accurate analysis



of opportunities, cash flows, return on investment, and profitability. The knowledgeable investor can beat the Real Estate game . . . make money . . . even create an empire!

HELP IS HERE

R.E.I.P. (Real Estate Investment Program) will help you master the complex world of Real Estate. It takes your input, probes for the right information, scrutinizes the data, the percentages, the opportunities . . . and automatically delivers the facts! In a matter of minutes! Whether you're interested in land, homes, apartments, commercial properties, or any phase of Real Estate . . . R.E.I.P. can help you reap bigger profits.



R.E.I.P. was originally created by a team of investors to provide an efficient decision-making tool for use in acquisition and disposition of real property investments. During a three year testing period, several systems were installed in brokerage offices to refine and polish R.E.I.P. The success of these early systems has resulted in the R.E.I.P. program which is being offered today.

R.E.I.P. BENEFITS THE SALES PROFESSIONAL, THE BUYER, AND THE SELLER

$ R.E.I.P. provides the sales professional with the ability to create comprehensive presentations . . . to explore a variety of deal structures in a short time by eliminating the tedious pencil and calculator process. With R.E.I.P., the sales person is able to forecast investment performance in changing market environments.

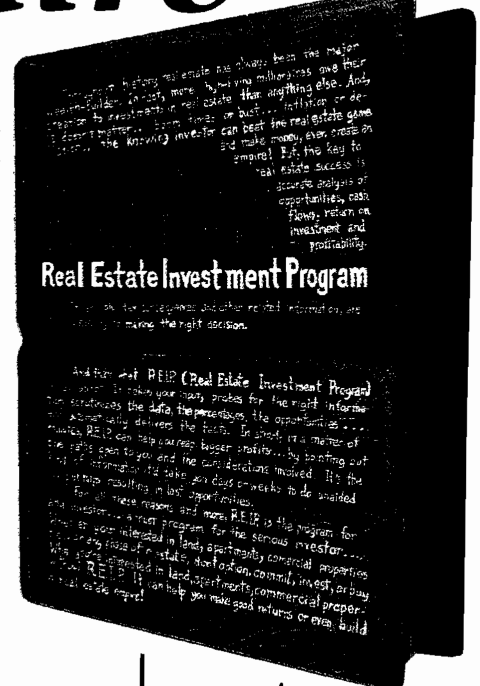
$ The prospective buyer of Real Estate is confronted with many questions. The ability to forecast and analyze an investment is essential in order to find successful and profit-making answers. R.E.I.P. can generate additional profits by performing the analytical steps quickly, without becoming lost in a forest of numbers.

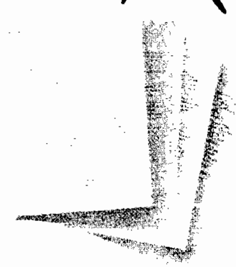
$ R.E.I.P. allows the seller to restructure a deal using different financing plans and income projections in order to maximize profits. The seller will get a realistic view of how the property should be priced, and the methods available to alter the pricing.

$ R.E.I.P. offers all involved parties a clear, concise presentation of the property, enhancing communication. And it does in minutes what would normally take days or weeks for someone who knew all the ins and outs.

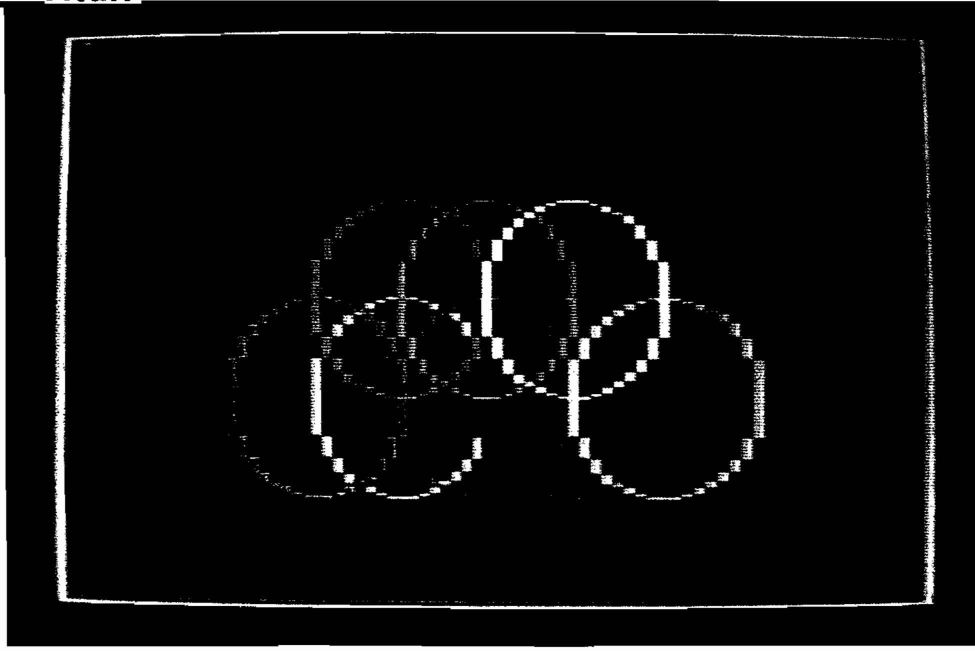
For all these reasons and more, R.E.I.P. is THE PROGRAM for any investor . . . a MUST PROGRAM for the sophisticated investor. We believe R.E.I.P. is the most detailed Real Estate Investment Program available for personal or small business computers. Don't option, commit, invest or buy without R.E.I.P. Get a jump on the competition . . . you'll be on your way to becoming a Real Estate Millionaire!

R.E.I.P. is available for Apple* and IBM-PC* computers. $179.95





DATAMOST
8943 Fullbright Ave.
Chatsworth, CA
91311-2750
(213) 709-1202



Four variables will remember the dot positions, which are selected before the S key is pressed. These variables store coordinate pairs of the pixel locations and are named YA, XA, YB, and XB. They are all initialized to zero on line 580.

To make this work, there must be provision for constantly updating these variables when the trigger button is pressed. YB and XB will hold the most recently selected pixel and YA and XA will hold the next most recently selected pixel. The program is interrupted at line 1160 by a GOTO 1220 in order to insert the extra statements required. The statements starting at 1220 will update YA, XA, YB, and XB when the trigger button is pressed and the cursor is moved. If the check for a moved cursor were not performed, then the two points would be the same location if the trigger button was held down too long. After this checking is done, another GOTO resumes normal processing.

When the S key is pressed, whatever the last two values are at that time will be used to form the shape. If the shape cannot be drawn for some reason, the shape-drawing routine simply returns to the mode 10 screen.

The shape selections are displayed on a standard mode zero text screen. The same method is implemented for shape selection as was used in implementing the load/save selector. The screen memory is saved in a buffer (GOSUB 20000) and a mode zero screen is used. This is done starting at line 2000, with the test for the S key in-

New Listing for Mode 10 Painting Program

(This Listing incorporates changes from Part II and Part III. Errors from Part I have been corrected)

```
10 REM *** MODE 10 PAINTER ***
12 REM ***     PROGRAM       ***
14 REM ***                   ***
16 REM *** Designed by       ***
18 REM ***  Paul S Swanson ***
20 REM ***                   ***
22 ? "↖↓↓↓            MODE 10 PAINTER PROGRA
M"
24 ? "             FOR ATARI COMPUTERS"
26 ? :? "Program by Paul S. Swanson"
28 ? :? :? "Initializing..."
30 REM +++ INITIALIZATION  +++
40 REM -- JOYSTICK READ TABLE --
50 DIM JOY(15,1),A$(2)
60 FOR RDG=1 TO 15:FOR DIR=0 TO 1:READ J
OY:JOY(RDG,DIR)=JOY:NEXT DIR:NEXT RDG
70 DATA 0,0,0,0,0,0,0,0,1,1,1,-1,1,0,0,0
,-1,1,-1,-1,-1,0,0,0,0,1,0,-1,0,0
80 REM -- POSITION STRINGS --
90 REM -- ON 1K BOUNDARY   --
100 DIM X$(1):A=ADR(X$):B=INT(A/1024+1)*
1024:DIM XX$(B-A-1):PMSTART=B/256
110 REM -- DISPLAY LISTS    --
120 DIM HELPDL$(64),SELDL$(64)
130 REM -- SCREEN AREAS     --
140 DIM HELPSC$(256),SELSC$(256)
150 REM -- INITIALIZE DL'S --
160 HELPDL$="PPPPPPPPB♥♥||||A"
170 SELDL$="PPPPPPPPO♥♥▪▪▪▪▪▪◘A"
180 ADRSETUP=5000
190 A=ADR(HELPDL$):GOSUB ADRSETUP:HELPDL
$(LEN(HELPDL$)+1)=A$
200 A=ADR(SELDL$):GOSUB ADRSETUP:SELDL$(
LEN(SELDL$)+1)=A$
210 REM -- INITIALIZE SCREENS --
220 HELPSC$=" ":HELPSC$(256)=" ":HELPSC$
(2)=HELPSC$
230 SELSC$="♥♥♥♥♥♥♥♥♥"
240 FOR I=17 TO 255 STEP 17
250 FOR J=1 TO 2:SELSC$(LEN(SELSC$)+1)=C
HR$(I):NEXT J:NEXT I:SELSC$(39,40)="♥♥"
260 SELSC$(256)=" ":SELSC$(41)=SELSC$
270 REM -- PUT SCREEN ADDRESSES
          INTO DISPLAY LISTS
280 A=ADR(HELPSC$):GOSUB ADRSETUP:HELPDL
$(10,11)=A$
290 A=ADR(SELSC$):GOSUB ADRSETUP:SELDL$(
10,11)=A$
300 REM -- INITIALIZE PLAYER2 --
310 DIM PL2$(128)
320 PL2$="♥":PL2$(128)="♥":PL2$(2)=PL2$
330 REM -- HELP SCREEN TEXT --
340 HELPSC$(1,40)="                HELP SCR
EEN                "
350 HELPSC$(41,80)="OPTION - HELP screen
    |↑↓←→ Fill on     "
360 HELPSC$(81,120)="SELECT - Color Sele
ction|S Shape drawing"
370 HELPSC$(121,160)="START - Change Pen
 Color|C Cancel fill     "
380 HELPSC$(161,200)="1,2 - Increment
    |D load/save      "
390 FOR I=1 TO LEN(HELPSC$):N=ASC(HELPSC
$(I)):N1=(N>127):N=N-N1*128
392 N=(N-32)*(N>31 AND N<96)+(N+64)*(N<3
2)+N*(N>95)+N1*128:HELPSC$(I,I)=CHR$(N):
NEXT I
```

(continued)

serted at line 3014. Lines 2000 through 2199 are reserved for handling the selection and the return to the mode 10 screen. All of the shape routines are written as subroutines.

Drawing the Shapes

The simplest shape to draw is the line. That routine is fully contained in line 2200. Just PLOT a point at XA,YA and DRAWTO XB,YB.

The rectangle outline routine is not much more complicated. That routine is fully contained in line 2300. Four lines are drawn to connect the four corners, which have coordinates defined by all four combinations of XA and XB with YA and YB.

A filled rectangle, done at line 2400, uses a FOR...NEXT loop to draw lines along the complete lengths of the top and bottom. For both rectangles, the sides are parallel to the sides of the screen.

The circle routines require the SIN and COS function and, as is indicated at line 2500, are computed in degrees. Both circle routines start by checking that no part of the circle will be off the screen.

There is an adjustment required because of the pixel shape. The coordinate formulae used assume equal units horizontally and vertically, so the vertical coordinates are adjusted by a factor of four. The vertical coordinate units are equal to four vertical lines.

The circle-outline routine first PLOTs a single point at the location corresponding to zero degrees. The FOR...NEXT loop that follows uses DRAWTO to fill in the rest of the sides.

The filled-circle routine draws circumscribed rectangles to fill in the circle. Note that the PLOT and DRAWTO statements are similar to the ones used for the rectangle outline. The two equations using the trigonometric functions in line 2630 determine the offsets in each direction from the center of the circle. The statements that follow draw a rectangle forming the corners with the center coordinates and these offsets. Since four quadrants are drawn using this method, the loop need contain only the logic for one quadrant, which is the reason the FOR...NEXT loop ends at 90.

Using the New Routines

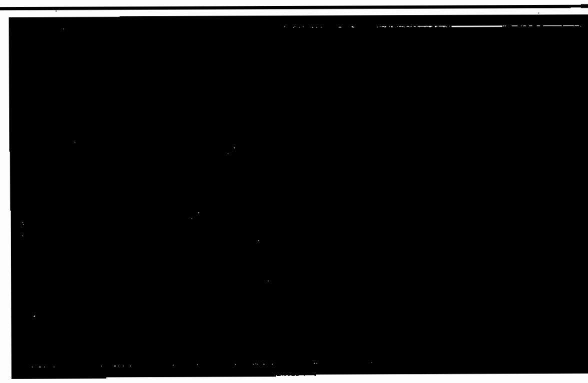
When using these new shape routines, you should be able to draw

(Continued on page 63)

Painting Program Listing *(continued)*

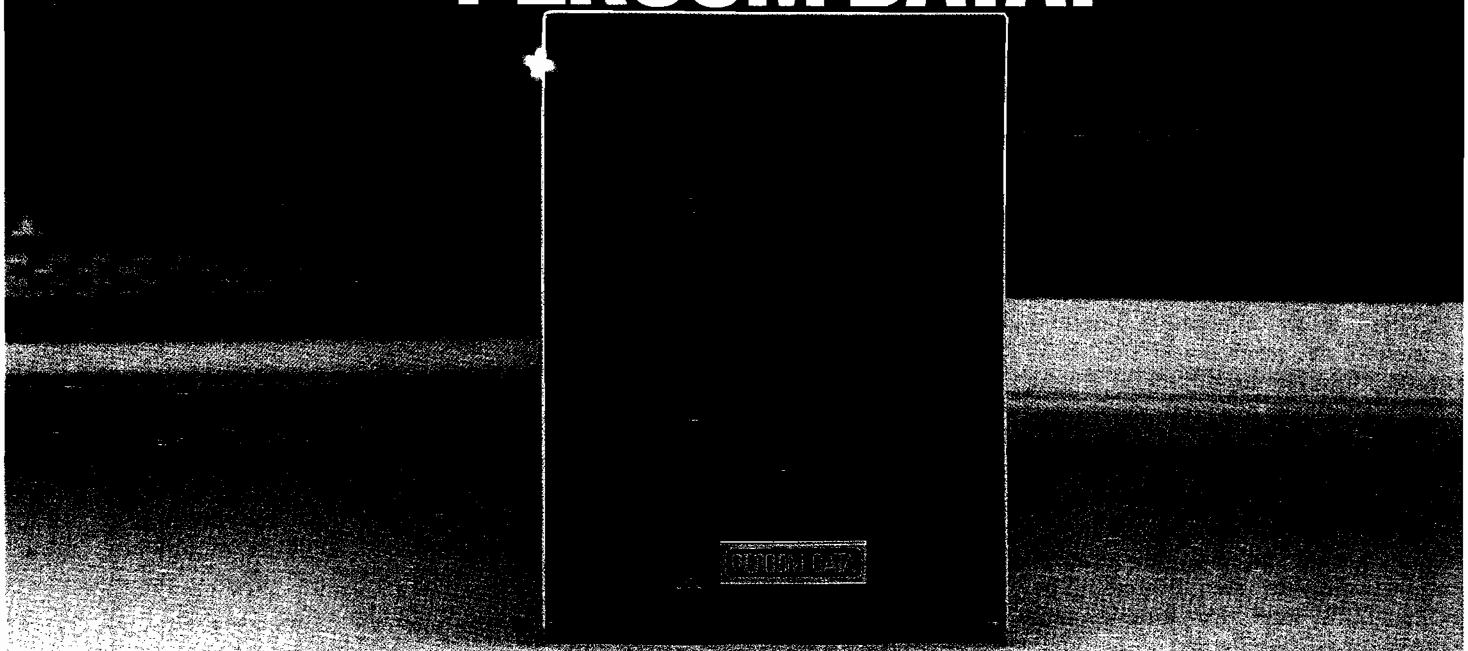
```
400 REM --- SET UP MODE 10 SCREEN
410 GRAPHICS 10
420 REM --- USE RANDOM COLORS ---
430 COL=25:FOR REG=704 TO 712
440 POKE REG,COL:COL=COL+25:NEXT REG
450 POKE 704,0
460 REM -- DEFINE CONSTANTS --
470 CONSOL=53279
480 CBASE=704
490 DMACTL=559
500 GRACTL=53277
510 HPOSP1=53249
520 PMBASE=54279
530 SIZEP1=53257
540 BEGIN=1000
550 KB=764
560 GTIA=623
570 NMIEN=54286
580 YA=0:YB=YA:XA=YA:XB=YB
600 REM -- INSTALL DLI ROUTINE --
610 RESTORE 7000:LOC=1536
620 READ N:IF N<256 THEN POKE LOC,N:LOC=
LOC+1:GOTO 620
630 POKE 512,0:POKE 513,6
640 REM -- ALTERNATE SCREENS --
650 DIM ALTSC1$(256),BUFF$(8192)
660 ALTSC1$="♥":ALTSC1$(256)="♥":ALTSC1$
(2)=ALTSC1$
670 FOR I=1 TO 240 STEP 40:ALTSC1$(I,I+2
3)=SELSC$(I,I+23):NEXT I
900 REM -- OTHER DIMS --
910 DIM RCOL(9),F$(12),Q$(40),FILE$(14),
LINE$(80)
920 REM -- INITIALIZE COUNTERS, ETC.
930 X=39:Y=96
932 VFILL=0:FILLFLAG=0
940 UNDERCURSOR=0
950 CURSORFLAG=0
960 CURSORCOUNT=0
970 SELCOLOR=1
980 FLASHCOUNT=0
982 INCREMENT=1
990 REM ***                      ***
992 REM *** MAIN PROGRAM TEXT ***
994 REM ***                      ***
996 REM -- READ JOYSTICK/CONSOL --
1000 STK=STICK(0):CURSORCOUNT=CURSORCOUN
T+1:IF CURSORCOUNT<4 THEN 1060
1002 IF FILLFLAG=0 AND VFILL=0 THEN 1040
1010 FLASHCOUNT=6-FLASHCOUNT
1020 SOUND 0,60,10,FLASHCOUNT:FOR DELAY=
1 TO 2:NEXT DELAY
1030 SOUND 0,0,0,0
1040 CURSORFLAG=1-CURSORFLAG:COL=UNDERCU
RSOR+CURSORFLAG:IF COL>8 THEN COL=0
1050 COLOR COL:PLOT X,Y:CURSORCOUNT=0
1060 IF STK<>15 OR STRIG(0)=0 THEN 1080
1070 SWITCH=PEEK(CONSOL):IF SWITCH<>7 TH
EN 4000
1072 IF PEEK(KB)<>255 THEN 3000
1074 GOTO BEGIN
1080 POKE 77,0
1110 COLOR UNDERCURSOR:IF STRIG(0)=0 THE
N COLOR SELCOLOR
1120 PLOT X,Y
1130 REM --- MOVE CURSOR ROUTINE --
1140 X=X+JOY(STK,0)*INCREMENT:Y=Y+JOY(ST
K,1)*INCREMENT
1150 X=X-INT(X/80)*80:Y=Y-INT(Y/192)*192
1160 LOCATE X,Y,UNDERCURSOR:GOTO 1220
1170 CURSORFLAG=0:CURSORCOUNT=4:IF (FILL
FLAG=0 AND VFILL=0) OR STRIG(0)=1 THEN G
OTO BEGIN
1172 REM -- FILL ROUTINE --
1180 X1=X:Y1=Y:COLOR SELCOLOR
1190 X1=X1+FILLFLAG*INCREMENT:IF X1>79 O
R X1<0 THEN GOTO BEGIN
1192 Y1=Y1+VFILL*INCREMENT:IF Y1>191 OR
Y1<0 THEN GOTO BEGIN
1200 LOCATE X1,Y1,TESTEND:IF TESTEND=SEL
COLOR THEN GOTO BEGIN
1210 PLOT X1,Y1:GOTO 1190
1220 IF STRIG(0)=1 THEN 1170
1230 IF X=XB AND Y=YB THEN 1170
1240 XA=XB:XB=X:YA=YB:YB=Y
```

(continued)



ATARI COMPUTER OWNERS:

Pick the positively perfect, practical, peripheral package, from PERCOM DATA!



That's right... the positively perfect PERCOM DATA 5¼" floppy disk drive with a BUILT-IN PRINTER-PORT, for your Atari® 400/800 is now available!

Until now, Atari computer owners who wanted to hook a printer to their computer had only one choice... spend about $220 for an interface device. THOSE DAYS ARE OVER. PERCOM DATA has built a parallel printer-port right into its new AT88 PD model. Now you can add a quality disk drive system AND have a place to plug in a printer... WITHOUT BUYING an interface.

The AT88 S1 PD™ disk drive operates in both single density (88K bytes formatted) and double density (176K bytes formatted).

What more could you want? NO INTERFACE... a high quality PERCOM DATA disk drive... AND a built-in PRINTER-PORT... all with a price of $599.

Pick up a positively perfect PERCOM DATA disk drive, with printer-port... pronto!

For the name of an authorized PERCOM DATA Dealer near you, call our **TOLL-FREE HOTLINE 1-800-527-1222 NOW,** or write for more information.

Perfectly Priced

$599.

PERCOM DATA
CORPORATION

Expanding Your Peripheral Vision

DRIVES • NETWORKS • SOFTWARE

11220 Pagemill Road, Dallas, Texas 75243 (214) 340-5800
1-800-527-1222

Painting Program Listing *(continued)*

```
1250 GOTO 1170
1997 REM ---
1998 REM --- SHAPES ROUTINES ---
1999 REM ---
2000 IF YA=0 AND YB=0 THEN GOTO BEGIN
2010 GOSUB 20000
2020 GRAPHICS 0
2030 ? "          ◆◆ SHAPE SELECTOR ◆◆":?
2040 ? " 1. Line"
2050 ? " 2. Rectangle (outline)"
2060 ? " 3. Rectangle (filled)"
2070 ? " 4. Circle (outline)"
2080 ? " 5. Circle (filled)"
2090 ? :? "Press number of selection";
2100 CLOSE #3:OPEN #3,4,0,"K:":GET #3,N:
CLOSE #3:N=N-48
2110 GRAPHICS 10:FOR I=0 TO 8:POKE I+CBA
SE,RCOL(I):NEXT I
2120 FOR I=0 TO 8190 STEP 256:A=USR(ADR(
Q$),BUFF+I,SCREEN+I):NEXT I
2130 LOCATE X,Y,UNDERCURSOR
2140 IF N<1 OR N>5 THEN GOTO BEGIN
2150 COLOR SELCOLOR
2160 GOSUB N*100+2100
2170 PLOT X,Y:CURSORFLAG=0:CURSORCOUNT=4
:LOCATE X,Y,UNDERCURSOR:GOTO BEGIN
2199 REM --- LINE ---
2200 PLOT XA,YA:DRAWTO XB,YB:RETURN
2299 REM --- RECTANGLE (OUTLINE) ---
2300 PLOT XA,YA:DRAWTO XB,YA:DRAWTO XB,Y
B:DRAWTO XA,YB:DRAWTO XA,YA:RETURN
2399 REM --- RECTANGLE (FILLED) ---
2400 FOR I=XA TO XB STEP SGN(XB-XA):PLOT
 I,YA:DRAWTO I,YB:NEXT I:RETURN
2499 REM --- CIRCLE (OUTLINE) ---
2500 DEG :R=SQR(((YA-YB)^2)/16+(XA-XB)^2
)
2510 IF XA<R OR (79-XA)<R THEN RETURN
2520 IF YA/4<R OR (48-YA/4)<R THEN RETUR
N
2530 PLOT XA+R,YA
2540 FOR I=0 TO 360 STEP 5:DRAWTO XA+R*C
OS(I),YA+R*SIN(I)*4:NEXT I
2550 RETURN
2599 REM --- CIRCLE (FILLED) ---
2600 DEG :R=SQR(((YA-YB)^2)/16+(XA-XB)^2
)
2610 IF XA<R OR (79-XA)<R THEN RETURN
2620 IF YA/4<R OR (48-YA/4)<R THEN RETUR
N
2630 FOR I=0 TO 90:COL=R*SIN(I):ROW=R*CO
S(I)*4
2640 PLOT XA+COL,YA+ROW:DRAWTO XA-COL,YA
+ROW:DRAWTO XA-COL,YA-ROW:DRAWTO XA+COL,
YA-ROW
2650 DRAWTO XA+COL,YA+ROW:NEXT I:RETURN
2990 REM --
2992 REM -- KEYBOARD INTERPRET ROUTINE
2994 REM --
3000 N=PEEK(KB):POKE KB,255:IF N=7 THEN
VFILL=0:FILLFLAG=1:GOTO BEGIN
3002 IF N=14 THEN FILLFLAG=0:VFILL=-1:GO
TO BEGIN
3004 IF N=15 THEN FILLFLAG=0:VFILL=1:GOT
O BEGIN
3010 IF N=6 THEN VFILL=0:FILLFLAG=-1:GOT
O BEGIN
3012 IF N=31 OR N=30 THEN GOTO 8000
3014 IF N=62 THEN 2000
3020 IF N=18 THEN FILLFLAG=0:VFILL=0:GOT
O BEGIN
3030 IF N<>58 THEN GOTO BEGIN
3040 GOSUB 20000
3050 GRAPHICS 0:? "              DISK TRANS
FERS":?
3060 ? " <1>  SAVE PICTURE"
3070 ? " <2>  LOAD PICTURE"
3080 ? " <3>  RETURN TO CURRENT PICTURE"
:?
3090 ? "PRESS NUMBER OF SELECTION--";
3100 CLOSE #3:OPEN #3,4,0,"K:":GET #3,N:
CLOSE #3
3110 N=N-48:IF N<1 OR N>3 THEN 3100
3120 GOTO N*100+3100
3200 ? "↖              SAVE PICTURE":? :
DIRECTION=8:GOSUB 10000:? "SAVING PICTUR
E":TRAP 40000
3210 FOR I=0 TO 8:? #3;RCOL(I):NEXT I
3220 FOR I=1 TO 8160 STEP 80:? #3;BUFF$(
I,I+79):NEXT I
3230 CLOSE #3:GOTO 3050
3300 ? "↖             LOADING PICTURE":?
:DIRECTION=4:? :GOSUB 10000:? "LOADING P
ICTURE"
3310 FOR I=0 TO 8:INPUT #3,RCOL:RCOL(I)=
RCOL:NEXT I
3320 FOR I=1 TO 8160 STEP 80:INPUT #3,LI
NE$:BUFF$(I,I+79)=LINE$:NEXT I
3330 CLOSE #3:GOTO 3050
3400 GRAPHICS 10:FOR I=0 TO 8:POKE I+CBA
SE,RCOL(I):NEXT I
3410 FOR I=0 TO 8190 STEP 256:A=USR(ADR(
Q$),BUFF+I,SCREEN+I):NEXT I
3420 LOCATE X,Y,UNDERCURSOR:GOTO BEGIN
3989 GOTO BEGIN
```

complete pictures easier and f[…] There are a few rules to follow to n[…] pictures even simpler to draw. Certain shapes can be easily derived from the ones in the routines. For example, a target can be drawn by selecting the color of the outermost circle first. Draw the circle, then select the color, center, and radial point for the next ring, and draw that circle. Each circle drawn will erase all but the toroid (donut shape) required for the ring. Similar images can be drawn using the filled rectangles.

When a shape is drawn, notice that the values stored in YA, XA, YB, and XB are not altered. A second shape may be superimposed directly. For example, draw a filled circle, then change color and draw a circle outline. The filled circle will simply be outlined in the second selected color.

Line drawings are easier because these values are not altered. To complete a line drawing, find a continuous path through it. Draw the first line, then move the cursor to the end of the next line in sequence. Press the trigger button and select the line shape; a line will be drawn to there from the end of the first line. This process can be continued throughout the figure and colors may be changed between lines, since color selection also does not alter the coordinates.

Add Images to Your Own Programs

The data files produced from this painter program can be read into other BASIC programs easily and used for partial or whole screen displays. They are mode 10 screen and so must be displayed as GTIA mode 10 screens. This means they can't be mixed with other screens without using a display-list interrupt to control the hardware register PRIOR at location 53275 (with a shadow at 623). To institute a mode 10 screen, the Atari Operating System writes a \$80 (decimal 128) to the shadow, which, in turn, gets written to the hardware register during the vertical blank interrupt. The screen is otherwise identical to a mode eight screen.

To load the screen data into memory, study the loading routine in this program and simply mimic it. Write it out to a mode 10 screen created with a GRAPHICS 10 statement. To form a custom display list requires obeying some memory boundary

(Continued on next page)

restrictions that are explained in *De Re Atari*, a publication of Atari, Inc. that I have mentioned in my column several times. That publication also explains the basics of implementing a display-list interrupt for mixing screens and what PRIOR does when a GTIA mode is implemented.

Other Additions

There are many possibilities for adding other shapes and features to this program. The scheme to plot out the shapes in this article has a provision built into it to plot shapes that require three points. The two that are saved to define the shape are not necessarily the current cursor position stored in X and Y. It is possible to set one point by positioning the cursor and hitting the trigger, then moving to a second point and doing the same, then moving to a third position and hitting the S key without hitting the trigger. When the program goes to the shape selector, XA, YA, XB, and YB will have the coordinates of the two points defined by hitting the trigger. X and Y will hold the coordinates of the current cursor position, providing the third point. A very simple routine could add a triangle, a skewed rectangle, or a circle fit to the three points (any three non-linear points define a circular arc). The three points could even define the center of a circle, the radius, and a central angle for drawing a pie-shaped segment, filled or outlined.

When drawing pictures in which a third dimension is simulated, the general rule is to draw the objects farthest away first, contrary to the way a scene is normally interpreted. A simple program alteration in the load/save routine would solve that problem. This routine would load a picture stored on disk or cassette over a picture in memory using only those pixels that have color other than the background color overwriting the corresponding pixel in memory. This would allow one picture to serve as background with several foregrounds added to it for new pictures — the same way in which many cartoons are created.

The number of functions that can be added to this program is limited only by the amount of available memory and by your own imagination.

You may contact Mr. Swanson at 97 Jackson St., Cambridge, MA 02140.

Painting Program Listing *(continued)*

```
3990 REM --
3992 REM -- FUNCTION KEY INTERPRETER
3994 REM --
4000 FOR I=1 TO 7:I=PEEK(CONSOL):NEXT I:
GOSUB 5020:MODERES=PEEK(GTIA)
4010 ON SWITCH GOTO BEGIN,BEGIN,4100,BEG
IN,4200,4300,BEGIN
4100 POKE GTIA,0:A=LEN(HELPDL$):POKE 560
,ASC(HELPDL$(A-1)):POKE 561,ASC(HELPDL$(
A))
4102 IF PEEK(CONSOL)<>7 THEN 4102
4110 IF STICK(0)=15 AND PEEK(KB)=255 AND
 PEEK(CONSOL)=7 THEN 4110
4120 POKE GTIA,MODERES:GOSUB 5030:GOTO B
EGIN
4130 GOTO BEGIN
4140 IF STRIG(0)=1 THEN 4120
4150 GOTO 4140
4200 A=LEN(SELDL$):POKE 560,ASC(SELDL$(A
-1)):POKE 561,ASC(SELDL$(A))
4210 A=ADR(ALTSC1$):GOSUB 5000:SELDL$(10
,11)=A$
4220 MSG=6010:MAXSEL=8:GOSUB 5040:COLNO=
SELECTION
4230 A=ADR(SELSC$):GOSUB 5000:SELDL$(10,
11)=A$:COLSAV=PEEK(CBASE+8):POKE CBASE+8
,8
4240 POKE GTIA,192:MSG=6020:MAXSEL=15:GO
SUB 5040
4250 POKE CBASE+8,SELECTION*16:COLUSED=S
ELECTION
4260 POKE GTIA,64:MSG=6030:MAXSEL=15:GOS
UB 5040
4270 POKE CBASE+8,COLSAV:COLUSED=COLUSED
*16+SELECTION
4280 POKE CBASE+COLNO,COLUSED:GOTO 4140
4300 A=LEN(SELDL$):POKE 560,ASC(SELDL$(A
-1)):POKE 561,ASC(SELDL$(A))
4310 A=ADR(ALTSC1$):GOSUB 5000:SELDL$(10
,11)=A$
4320 MSG=6000:MAXSEL=8:GOSUB 5040
4330 A=ADR(SELSC$):GOSUB 5000:SELDL$(10,
11)=A$
4340 SELCOLOR=SELECTION:GOTO 4140
4990 STOP
4992 REM +++
4994 REM ---  SUBROUTINES ---
4996 REM ---
4998 REM ---  Conv't A to address
           in A$
4999 STOP
5000 HI=INT(A/256):LO=A-HI*256
5010 A$=CHR$(LO):A$(2)=CHR$(HI):RETURN
5020 SHI=PEEK(561):SLO=PEEK(560):RETURN
5030 POKE 561,SHI:POKE 560,SLO:RETURN
5040 RESTORE MSG:READ F$
5050 PL2$(50,56)="◢+*I◢◢◢":BASE=58
5060 FOR I=1 TO LEN(F$):N=(ASC(F$(I))-32
)*8+57344
5070 FOR J=0 TO 7:PL2$(J+BASE,J+BASE)=CH
R$(PEEK(J+N)):NEXT J
5080 BASE=BASE+8:NEXT I
5090 POKE NMIEN,192:POKE DMACTL,42:POKE
GRACTL,2:POKE PMBASE,PMSTART
5100 POKE SIZEP1,0:SELECTION=0:POKE 512,
0:POKE 513,6:MAXSEL=MAXSEL+1
5110 POKE HPOSP1,SELECTION*8+72
5120 STK=STICK(0):IF STK=15 AND STRIG(0)
=1 THEN 5120
5130 IF STRIG(0)=0 THEN POKE HPOSP1,0:PL
2$(75)=PL2$(74):RETURN
5140 SELECTION=SELECTION+(STK=7)-(STK=11
)
5150 SELECTION=SELECTION-INT(SELECTION/M
AXSEL)*MAXSEL:POKE HPOSP1,SELECTION*8+72
5160 SOUND 0,135,10,6:FOR DELAY=1 TO 50:
NEXT DELAY:SOUND 0,0,0,0
5170 GOTO 5120
6000 DATA COLR
6010 DATA CHGE
6020 DATA HUE
6030 DATA LUM
7000 DATA 72,169,14,141,19,208,169,0,141
,10,212,141,27,208,141,26,208,104,64,256
8000 INCREMENT=32-N:GOTO BEGIN
10000 ? "ENTER FILE SPEC - MAX. 8 CHARAC
TERS:"
10010 INPUT FILE$
10020 IF LEN(FILE$)<2 THEN 11000
10030 TRAP 11000
10050 OPEN #3,DIRECTION,0,FILE$:RETURN
11000 ? "[ERROR] - NOT A VALID NAME":FOR
I=1 TO 300:NEXT I:CLOSE #3:GOTO 3050
20000 Q$="hh.Oh.Nh.Qh.P ♥HIN.PC♥PW♦":BUF
F$(8192)="♥"
20010 DLIST=PEEK(560)+PEEK(561)*256:SCRE
EN=PEEK(DLIST+4)+PEEK(DLIST+5)*256:BUFF=
ADR(BUFF$)
20020 FOR I=0 TO 8190 STEP 256:A=USR(ADR
(Q$),SCREEN+I,BUFF+I):NEXT I
20030 FOR I=0 TO 8:RCOL(I)=PEEK(I+CBASE)
:NEXT I
20040 RETURN
```

MICRO™

Alspa Computer, Inc.

Price-performance leader. Includes Z80A, 8" ds/dd drives, 3 serial + 1 parallel port, winchester port, networking. Prices start below $1500. DEALER / OEM inquiries invited.

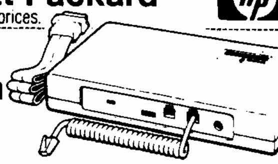
SPECIALS on INTREGATED CIRCUITS

6502	7.45	10/ 6.95	50/ 6.55	100/ 6.15
6502A/6512A	8.40	10/ 7.95	50/ 7.35	100/ 6.90
6520 PIA	5.15	10/ 4.90	50/ 4.45	100/ 4.15
6522 VIA	6.45	10/ 6.10	50/ 5.75	100/ 5.45
6532	7.90	10/ 7.40	50/ 7.00	100/ 6.60
2114-L200		2.45	25/ 2.30	100/ 2.15
2716 EPROM		4.90	5/ 4.50	10/ 4.00
2532 EPROM		6.90	5/ 6.75	10/ 6.45
6116 2KX8 CMOS RAM		6.90	5/ 6.75	10/ 6.45
4116 RAM		8 for 14		
Zero Insertion Force 24 pin Socket (Scanbe)				2.00

Hewlett Packard

Write or call for prices.

Anchor Automation Signalman Modems

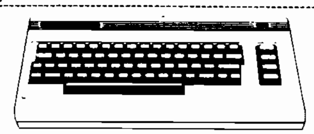


FREE SOURCE MEMBERSHIP WITH SIGNALMAN

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

Mark I RS232	(99)	79
Mark II for Atari 850	(99)	79
Mark IV for CBM/PET with software	(169)	119
Mark V for Osborne (software available)	(129)	93
Mark VI for IBM Personal Computer	(279)	195
Mark VII Auto Dial/Auto Answer	(179)	119
Mark VIII Bell 212 Auto Dial/Answer	(399)	319

DC HAYES Smartmodem 219
DC Hayes Smartmodem 1200 545



PROM QUEEN for VIC	170
Apple Emulator for Commodore 64	89
Screenmaker 80 COLUMN CARD for C64	145
Solid Oak 2 Level Stand for C64 or VIC	29
C64/VIC Switch (networking)	125
BACKUP V1.0 tape copier for C64 or VIC	20
CARDBOARD/6 Motherboard - VIC	64
CARDAPTER/1 Atari VCS Adapter - VIC	69
CARDPRINT Printer Interface - C64/VIC	64
CARDBOARD/3s Motherboard - VIC	32
CARDRITER Lightpen - C64/VIC	32
CARDRAM/16 RAM Expansion - VIC	64
Complete CARDCO Line in stock	
CIE and VIE IEEE Interfaces in stock	
BASM kCompiler/Assembler for C64	89

APPLE—FRANKLIN ITEMS

KRAFT Apple Joystick	43
16K RAM Card for Apple	59
Solid Oak 2 Level Stand for Apple	29
Serial Card for Apple	99
MPC RAM/80 column card for IIE	139
Z80 Softcard and CP/M (Microsoft)	235
RANA Elite I with Controller	389
Parallel Printer Interface/Cable	79
Apple Dumpling (Microtek) Printer Interface	115
Apple Dumpling with 16K Buffer	160
Grappler + Interface	140
Kraft Products for Apple in stock	
DC Hayes Micromodem II	299
PFS: File	100
PFS: Report	100
Videx 80 Column Card	209
Hayden Software for Apple 20% OFF	
PIE Writer Word Processor	120

commodore

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

PETSCAN I $245 base price

Allows you to connect up to 30 CBM/PET Computers to shared disk drives and printers. Completely transparent to the user. Perfect for schools or multiple word processing configurations. Base configuration supports 2 computers. Additional computer hookups $100 each.

COMPACK $115

Intelligent Terminal Package for PET, CBM, C64
Includes ACIA Hardware / STCP Software

VE-2 IEEE to Parallel Interface 110

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

VIDEO ENHANCER for Commodore 64 89
Realize video quality equal or better than composite monitor using standard color TV.

SCREENMAKER 80 Column Adapter for C64 145
Provides big screen capability for business applications.

VIC 20 Products		VIC Sargon II Chess	32
BACKUP V1.0	20	**VIC GORF**	32
VIC RAM Cards in stock		Meteor Run (UMI)	39
VIC SuperExpander	52	VIC Radar Ratrace	24
VIC 16K RAM	95	Amok (UMI)	20
Thorn EMI Software		Snakman	15
HES Software		Rubik's Cube	13
VIC Omega Race	32	Programmers Reference	15
Spiders of Mars (UMI)	39	**FROGGER**	25
Programmers Aid	45	VIC Adventure Series	

VICTORY Software for VIC and C64

Street Sweepers (VIC)	12	Kongo Kong (VIC)	16
Night Rider (VIC)	11	Cosmic Debris (VIC)	12
Annihilator	16	Adventure Pack I	16
Adventure Pack II	16	Metamorphosis	11
Educational Pack I	11	Trek	12
Strategy Pack I	16	Grave Robbers	12

Commodore 64 Programmers Reference Guide	16
MicroChess for C64—8 levels of play	19
Compute!'s First Book of PET/CBM	11
C64 or VIC SWITCH	125
POWER ROM Utilities for PET/CBM	78
WordPro 3+/64	69
WordPro 4+ - 8032, disk, printer	295
SPELLMASTER spelling checker for WordPro	170
VISICALC for PET, ATARI, or Apple	189
PET-TRAX PET to Epson Graphics Software	40
SM-KIT enhanced PET/CBM ROM Utilities	40
Programmers Toolkit - PET ROM Utilities	35
CALC RESULT for C64	135
PET Spacemaker II ROM Switch	36
COPYWRITER Word Processor for C64	69
2 Meter PET to IEEE or IEEE to IEEE Cable	40
Dust Cover for PET, CBM, 4040, or 8050	8
CmC Interfaces (ADA1800, ADA1450, SADI in stock)	
ZRAM - CBM 64K RAM, Z80, CP/M	550
Programming the PET/CBM (Compute!) — R. West	20
Compute! First Book of VIC	11
HES MODEM with Software	65
HES Software and Hardware in stock	
UMI products in stock	
OMNICALC (HES) Spreadsheet for C64	79

FlexFile for PET/CBM/ C64 $110

Database, Report Writer with calculations, Mailing Lists.

FORTH for PET/C64 full FIG model — Cargill/Riley	$50
Metacompiler for FORTH for independent object code	30
KMMM PASCAL for PET/CBM/C64	79
EARL for PET/CBM Disk-based ASSEMBLER	65
Super Graphics — BASIC Language Extensions	45
Fast machine language graphics routines for PET/CBM	
RAM/ROM for PET/CBM 4K $75 8K	$90
DISK ICU – Recovery System for PET/CBM	40

DISK SPECIALS

Scotch (3M) 5" ss/dd	10/ 2.20	50/ 2.00	100/ 1.95
Scotch (3M) 5" ds/dd	10/ 3.05	50/ 2.80	100/ 2.75
Scotch (3M) 8" ss/sd	10/ 2.30	50/ 2.10	100/ 2.06
Scotch (3M) 8" ss/dd	10/ 2.85	50/ 2.70	100/ 2.65

We stock VERBATIM DISKS

Write for Dealer and OEM prices.

Sentinal 5" ss/dd	10/ 1.90	50/ 1.85	100/ 1.80
Sentinal 5" ds/dd	10/ 2.55	50/ 2.50	100/ 2.45
Wabash 5" ss/sd	10/ 1.65	50/ 1.60	100/ 1.55
Wabash 5" ss/dd	10/ 1.95	50/ 1.90	100/ 1.85
Wabash 8" ss/sd	10/ 2.00	50/ 1.95	100/ 1.85

We stock MAXELL DISKS

Write for dealer and OEM prices.

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

CASSETTE TAPES—AGFA PE-611 PREMIUM

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

DATASHIELD BACKUP POWER SOURCE 265
Battery back up Uninterruptible Power Supply with surge and noise filtering. The answer to your power problems.

Zenith ZVM-121 Green Phosphor Monitor	98
BMC 12A 12" Green Monitor	85
VOTRAX Personal Speech System	280
VOTRAX Type-N-Talk	160
VOICE BOX Speech Synthesizer (Apple or Atari)	
CompuServe Subscription (5 hours free)	32
Brother HR-15 Daisy Wheel Printer	475
Prowriter Parallel Printer	379
Panasonic 1090 Printer with Correspondence Mode	365
USI CompuMOD 4 R F Modulator	39
Daisywriter 2000 with 48K buffer + cable	1150
Many printers available (Gemini-Star, Brother, OKI, etc.)	
We Stock AMDEK Monitors	
Amdek DXY-100 Plotter	590
A P Products	15% OFF
Watanabe Intelligent Plotter 990	6-pen 1290
BROOKS 6 Outlet Surge Suppressor/Noise Filter	54
We stock Electrohome Monitors	
Synertek SYM-1 Microcomputer	189

ALL BOOK and SOFTWARE PRICES DISCOUNTED

Panasonic TR-120M1P 12" Monitor (20 MHz)	149
Panasonic CT-160 Dual Mode Color Monitor	285

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

ZENITH data systems

Z29 Terminal (DEC and ADM compatible)	680
ZT-1 Intelligent Communications Terminal	369
ZT-10 Intel. Terminal with Serial Port	340
Z100 16-bit/8-bit Systems in stock	CALL

We stock entire Zenith line.

ATARI SPECIALS

WE STOCK ENTIRE LINE—write for prices.

Atari 1200	CALL	**QIX**	34
Voice Box	100	**Anchor Modem—Atari**	79
FROGGER	25	**Atari Graphics (Compute!)**	11
Thorn EMI Software		**First Book of Atari**	11
EduFun Software		**APX Software**	

215-822-7727
252 Bethlehem Pike
Colmar, PA 18915

A B Computers

WRITE FOR CATALOG. Add $1.50 per order for United Parcel. We pay balance of UPS surface shipping charges on all prepaid orders (add extra for mail, APO/FPO, air). Prices include cash discount. Regular prices slightly higher. Prices subject to change.

A Personal Look At A Personal Computer

by Richard C. Vile, Jr.

The author describes his experiences of owning an Apple II and argues that the Apple II should not be regarded as obsolete. Included is an entertaining game to challenge the reader's intellect.

Sieve and Ups 'N Downs
requires:
Apple II with Integer BASIC
Programs Included for CoCo, C64, and Atari

I bought my Apple on the spur of the moment after two years of planning. This situation is not as paradoxical as it sounds. I had been thinking in the abstract for at least two years of buying a computer. But I couldn't decide which computer was best for me. I was a teacher of computer science and as such I wanted a computer that would give me the maximum access to programming. In particular, I wanted to be able to use machine and assembly language easily. I also wanted graphics. Color was not a necessity, although it would be nice.

After looking at many ads, I considered the ALTAIR and went to the MITS Caravan — a travelling computer roadshow. The ALTAIR seemed to require too much effort on the hardware side, whereas my interest was strictly in programming. I reluctantly gave up on ALTAIR. I mulled over other systems such as Southwest Technical Products, I scrutinized Digital Group systems. I went to see an OSI Challenger, I dropped by the local Radio Shack outlet and played with a TRS-80. I veritably lusted over a system called the ECD Micromind, which seemed to have great graphics. But I never could quite bring myself to plunk my money down on the counter.

Then in 1979 my son had to have his tonsils out and I stayed home to be with him after his ordeal. I said to myself "Maybe Kurt would like to see a home computer in action to take his mind off his discomfort" (sure Dad — you bought it to heal my tonsils... right!!!). So I went to the local computer emporium and bought an Apple II with 48K, a single disk drive (which they didn't have in stock, but I nonetheless ordered) and a few other odds and ends.

Why did I finally buy the Apple after all my wavering? I can only characterize the decision as a cross between a hunch and an impulse. I think it was the color graphics that tipped the balance. I don't know for sure and now that it's done and nearly three years later, it doesn't really matter a whole lot. I do know I'm not sorry I bought a computer and I am glad it was an Apple!

What Good is an Apple Anyway?

There are hundreds of practical answers to the question just posed. However, I think that in many cases the real answer is just as much emotional as it is practical. An Apple is more than just a computer, or a high-priced toy, or even a tool for increased individual productivity — although it certainly is "all of the above." At least for me, the Apple is also a *prized possession.* It's like a fine instrument is to a musician, or a well-worn radial arm saw is to a home craftsman, or a carefully tended set of copper pots is to a cook. Yes, it is a tool, instrument, and utensil. But it is more than that — much more. Losing my Apple would be like losing a friend.

But enough of waxing sentimental. The bottom line is still, what good is it? What do you use it for? What can you tell me about it that I don't already know?

To answer, at least partially, I will list a few of the uses to which I have put my Apple.

Education: Self

I have learned about microprocessors (specifically the 6502) and microcomputers. I have learned a lot about programming. In fact, for the first two years that I owned the Apple, I got my principal programming education using it. Even though I worked for a computer company, I had meandered into the lower levels of management and had little opportunity to do much real programming at work.

Writing

I got started writing for MICRO, not surprisingly, because of my Apple. To begin, the Apple gave me a source of material. Later, it became the principal instrument for turning ideas into hard copy. I now use the Apple to produce all my articles.

Using the Apple as a writing instrument falls into the category of word processing. Although I use a fairly simple program in Apple Pascal, which I wrote myself, I still can be far more productive than I ever could using my reconditioned Selectric typewriter. If I wanted to invest a couple of thousand dollars for a daisy-wheel printer, I could even have a professional-quality system in my basement. Many people do!

Education: Others

The Apple is a great teaching tool. With the plethora of languages and software available for it, the Apple can do more today for a person just beginning to learn about computers than many much larger computers could do just 10 or 15 years ago. Please indulge me while I offer a personal story to illustrate.

I got involved with computers just after I joined the mathematics department at Eastern Michigan University in 1970. The first computer I ever used was the instructional computer at Eastern — an IBM 1130. The first language in which I learned to program was FORTRAN. Being a mathematician, the first program I ever wrote was a prime number calculating program using the Sieve of Eratosthenes.

To run a job on the 1130, you first keypunched your program onto cards and then submitted the resulting "deck" to the computer operator. The operator stacked all the card decks into a reader and ran the jobs sequentially — in "batches," as the terminology went.

The computer was housed in a large room, but you could watch its operations through lots of big windows. Each job printed out a log on the line printer. The first page of this log had your user ID (including your name or initials) printed in big block letters. That meant you could watch the paper feeding out of the back of the printer and see your job starting up. I still remember doing this with my sieve program. I decided to time the computer to see how long it took to do the calculations. I started timing after the header page of the job log came out of the printer and stopped timing when my output began being printed. The primes were first calculated then all printed at once at the end. It took the 1130 90 seconds to compute the primes less than 10,000. This counted the time to translate the program from FORTRAN into 1130 machine language.

After I had owned my Apple for a year or so, I remembered my timing experiment. I decided to compare the Apple to the 1130. The program in listing 1 shows the same Sieve algorithm I implemented in my first FORTRAN program but now written in Apple Integer BASIC. I ran this program on the Apple and timed it between my RUN command and when it started printing out its list of primes less than 10,000. The Apple took only about 77 seconds to do the job! With a little trickery, this can be reduced to 36 seconds! Listing 2 shows a modification to the INIT subroutine, which uses the Monitor MOVE routine to perform the array initialization. This shaves about 40 seconds off the program's execution time.

The timing experiment is certainly comparing Apple's to Orange's (or whatever), but it does illustrate some important facts:

- The amazing power of microcomputers
- Just how far computer technology has progressed in 10 years
- That the Apple of today is comparable to the mini-computer of 10 years ago.

The Apple can be used to teach just about any undergraduate computing class. With proper simulation, you can even use it to teach assembly language for the IBM 1130!

Entertainment

The Apple is also a wonderful, albeit expensive, toy. The range of graphics applications from fantastic arcade-style games to hi-res adventures available for the Apple boggles the mind. When my Apple is "cooling" off from a hot session of word processing, my son uses it for games.

Applications

The term "applications" is a vague one at best. Just about any program might be dubbed an application. Roughly speaking, an application is a program that can be put to *practical* use. Some examples are:

- Checkbook and Home Financial Management
- Word processing and text preparation
- Spelling checking
- Mailing list preparation
- General Database Management

Of course, I have conspicuously omitted from the above list the one Apple application that almost defines the term. That is VisiCalc. VisiCalc, or Visible Calculator, combines the numerical calculation abilities of a microprocessor with the randomly addressable display in a true tour-de-force of programming. In fact, unknown thousands of Apple's have been sold simply because VisiCalc was originally written for the Apple and for a long time was available only on the Apple. I won't say more about VisiCalc, since so much has already been said (including entire books on how to use it).

Personal Programming

You can't use a computer without first programming it. Some people buy their programs, others prefer to write their own. Programming a computer is a satisfying, entertaining, and sometimes compelling activity. Certain people have been known to forego all other activity in order to sit in front of a computer terminal for long periods of time. (Guinness Book of World Records, please note!)

I am one of those people who enjoy programming as an end in itself. Programming is a lot like writing — it is a form of self-expression. The first time a program works can be an intensely satisfying moment.

Onward and Upward with the Apple

The Apple II just may be the world's most popular computer right now. In the brief existence of the explosive personal computer industry, some significant fraction of a million Apple's has been sold. Meanwhile the marketplace has been flooded with competition: Radio Shack (TRS-80, Model III, the Color Computer, etc.), Commodore (PET, Super-PET, VIC-20, Commodore 64, etc.), Atari (400, 800), IBM PC, EXIDY Sorceror, and now the tidal wave of Japanese imports — EPSON, SONY, Panasonic, Casio, etc.

The question arises, now that Apple's are not unique, do we just put them in the corner and let them gather dust? Do we bid our Apple a fond farewell and opt for the shiny new 16-bitters? My answer is a resounding NO! While many of us may acquire a second computer that is more powerful than our Apple, that is no reason to throw the Apple out in the trash. It is still a tool whose versatility and effectiveness deserve continued exploitation for many years to come.

One way to make sure your Apple doesn't lose its "bite" is to continue reading and learning about it and computers in general. I hope to encourage that activity by my series of articles as a MICRO contributing editor.

You may contact the author at 3467 Yellowstone Dr., Ann Arbor, MI 48105.

(Listings begin on next page)

Listing 1

```
   1 DIM SIEVE(5000)
   5 INIT=1000:NEXT=500:SIFT=600
   6 WAIT=700
  20 GOSUB INIT
  30 PRIME=3: GOTO 50
  40 GOSUB NEXT
  50 IF PRIME*PRIME>=10000 THEN 65
  55 PRINT "SIFTING OUT MULTIPLES OF ";PRIME
  60 GOSUB SIFT: GOTO 40
  65 PRINT "DONE SIFTING - PRESS A KEY TO
       GET LIST": GOSUB WAIT
  66 PRINT "LIST OF PRIMES < 10,000": PRINT : PRINT
  67 PRINT "2";" "
  70 COUNT=1
  75 FOR I=1 TO 4999
  80 IF SIEVE(I)=0 THEN 100
  90 PRINT (2*I+1);" ";
  95 COUNT=COUNT+1: IF COUNT MOD 5=0 THEN PRINT
 100 NEXT I
 105 PRINT : PRINT
 110 PRINT "THERE ARE ";COUNT;" PRIMES"
 115 PRINT "LESS THAN TEN-THOUSAND"
 125 END
 500 REM  NEXT
 505 I=PRIME/2
 510 I=I+1: IF SIEVE(I)=0 THEN 510
 515 PRIME=2*I+1
 520 RETURN
 600 REM  SIFT
 610 FOR J=PRIME*PRIME TO 10000 STEP 2*PRIME
 615 SIEVE(J/2)=0
 620 NEXT J
 630 RETURN
 700 KEY= PEEK (-16384): IF KEY<128 THEN 700
 705 POKE -16364,0: RETURN
1000 PRINT "INITIALIZING THE SIEVE"
1010 FOR I=1 TO 5000
1015 SIEVE(I)=1
1020 IF I MOD 1000=0 THEN PRINT I
1025 NEXT I
1030 RETURN
```

Listing 2

```
   1 DIM SIEVE(5000)
   5 INIT=1000:NEXT=500:SIFT=600
   6 WAIT=700:MOVE=-468
  20 CALL -936: PRINT "QUICKER SIEVE": PRINT : PRINT
  25 GOSUB INIT
  30 PRIME=3: GOTO 50
  40 GOSUB NEXT
  50 IF PRIME*PRIME>=10000 THEN 65
  54 PRINT
  55 PRINT "SIFTING OUT MULTIPLES OF ";PRIME
  60 GOSUB SIFT: GOTO 40
  65 PRINT : PRINT "DONE SIFTING - HIT A KEY
       TO GET LIST": GOSUB WAIT
  66 PRINT "LIST OF PRIMES < 10,000": PRINT : PRINT
  67 PRINT "2";" "
  70 COUNT=1
  75 FOR I=1 TO 4999
  80 IF SIEVE(I)=0 THEN 100
  90 PRINT (2*I+1);" ";
  95 COUNT=COUNT+1: IF COUNT MOD 5=0 THEN PRINT
 100 NEXT I
 105 PRINT : PRINT
 110 PRINT "THERE ARE ";COUNT;" PRIMES"
 115 PRINT "LESS THAN TEN-THOUSAND"
 125 END
 500 REM  NEXT
 505 I=PRIME/2
 510 I=I+1: IF SIEVE(I)=0 THEN 510
 515 PRIME=2*I+1
 520 RETURN
 600 REM  SIFT
 610 FOR J=PRIME*PRIME TO 10000 STEP 2*PRIME
 615 SIEVE(J/2)=0
 617 PRINT ".";
 620 NEXT J
 630 RETURN
 700 KEY= PEEK (-16384): IF KEY<128 THEN 700
 705 POKE -16364,0: RETURN
1000 PRINT "INITIALIZING THE SIEVE"
1005 POKE 2056,0: POKE 2057,1
```

(continued)

Listing 2 *(continued)*

```
1010 POKE 60,8: POKE 61,8
1015 POKE 62,23: POKE 63,47
1020 POKE 66,10: POKE 67,8
1025 CALL MOVE
1049 RETURN
```

Listing 3

```
   5 DIM SLOT(8),ANS$(10)
  10 INTRO=2000:INIT=1900:GETNUM=1800
  11 VALID=1000:UPDAYTE=1100:CHECK=1200
  12 CLR=-16368:KBD=-16384:FLIP=1300
  15 GAMES=0
  20 FOR I=1 TO 8:SLOT(I)=1: NEXT I
  25 GAMES=GAMES+1
  30 GOSUB INTRO
  40 GOSUB INIT
  50 MOVES=0:DONE=0
 100 GOSUB GETNUM
 105 GOSUB VALID
 110 IF OK THEN GOSUB UPDAYTE
 112 IF NOT OK THEN PRINT "";:REM CTRL-G
 115 MOVES=MOVES+1
 120 GOSUB CHECK
 125 IF NOT DONE THEN 100
 200 REM PRINT WINNING MESSAGE
 201 REM =====================
 205 VTAB 18: TAB 1: PRINT "CONGRATULATIONS!
       YOU SOLVED THE PUZZLE"
 210 PRINT "IN ";MOVES;" MOVES"
 215 PRINT "TRY AGAIN? "
 220 INPUT ANS$
 230 IF ANS$="Y" OR ANS$="YES" OR
       ANS$="OK" THEN 20
 299 END
1000 REM CHECK VALIDITY OF MOVE
1001 REM FOR KEY ENTERED (1-8).
1002 REM ======================
1005 N=KEY- ASC("0"):OK=0
1010 IF N#1 THEN 1020
1015 OK=1: RETURN
```

Listing 3 *(continued)*

```
1020 IF N#2 THEN 1030
1025 OK=(SLOT(1)=1): RETURN
1030 IF SLOT(N-1)#1 THEN RETURN
1035 FOR I=1 TO N-2
1040 IF SLOT(I)=1 THEN RETURN
1045 NEXT I
1050 OK=1: RETURN
1100 REM UPDATE THE SLOT ARRAY AND
1101 REM THE PUZZLE DISPLAY.
1102 REM ========================
1105 GOSUB FLIP
1110 SLOT(N)=1-SLOT(N)
1149 RETURN
1200 REM CHECK IF THE PUZZLE HAS
1201 REM BEEN SOLVED.
1202 REM =======================
1205 CHK=0
1210 FOR I=1 TO 8:CHK=CHK+SLOT(I): NEXT I
1215 DONE=(CHK=0)
1220 RETURN
1300 REM FLIP A TAB ON THE DISPLAY.
1301 REM ========================
1305 IF SLOT(N)=0 THEN 1315
1310 NOW=9:NEW=13: GOTO 1320
1315 NOW=13:NEW=9
1320 VTAB NOW: TAB 12+2*(N-1): PRINT " ";
1325 VTAB NEW: TAB 12+2*(N-1): PRINT N;
1349 RETURN
1800 REM ROUTINE TO GET A KEY
1801 REM BETWEEN 1 AND 8
1802 REM ===================
1805 KEY= PEEK (KBD): IF KEY<128 THEN 1805
1810 POKE CLR,0
1815 IF KEY>= ASC("1") AND KEY<= ASC("8") THEN RETURN
1820 GOTO 1805
1900 REM INITIALIZE THE DISPLAY
1901 REM =====================
1905 TEXT : CALL -936
1910 VTAB 7: TAB 19: PRINT "UP"
1915 VTAB 9: TAB 12: PRINT "1 2 3 4 5 6 7 8"
```

(continued)

Listing 3 *(continued)*

```
1920 VTAB 11: TAB 12: PRINT "= = = = = = = ="
1925 VTAB 15: TAB 17: PRINT "DOWN"
1949 RETURN
2000 REM INTRODUCTION TO GAME
2001 REM ===================
2005 TEXT : CALL -936
2010 IF GAMES>1 THEN RETURN
2015 VTAB 2: TAB 1
2020 PRINT "WELCOME TO UPS 'N DOWNS!"
2025 PRINT "THIS IS A GAME OF COMBINATIONS IN WHICH"
2030 PRINT "THE AIM IS TO MOVE A ROW OF NUMBERED"
2035 PRINT "TABS FROM A POSITION IN WHICH THE TABS"
2040 PRINT "ARE ALL 'UP', TO A POSITION IN WHICH "
2045 PRINT "THE TABS ARE ALL 'DOWN'."
2050 PRINT "AT ANY GIVEN TIME DURING THE GAME, SOME"
2055 PRINT "BUT NOT ALL OF THE TABS ARE FREE TO "
2060 PRINT "MOVE.  THOSE WHICH ARE FREE MAY BE"
2065 PRINT "CHANGED FROM UP TO DOWN OR VICE-VERSA,"
2070 PRINT "BY TYPING THE NUMBER OF THE TAB. IF YOU"
2075 PRINT "TYPE THE NUMBER OF A TAB WHICH IS NOT"
2080 PRINT "FREE TO MOVE, THE GAME WILL SOUND A "
2085 PRINT "BEEP AND REQUIRE YOU TO SELECT SOME"
2090 PRINT "OTHER NUMBER.  CHANGING A TAB FROM ONE"
2095 PRINT "POSITION TO THE OPPOSITE POSITION ALSO"
2100 PRINT "CHANGES WHICH OF THE OTHER TABS ARE"
2105 PRINT "THEN FREE TO MOVE."
2110 POKE 50,63: PRINT : TAB 10: PRINT "HIT ANY KEY TO BEGIN";
2115 POKE 50,255
2120 POKE CLR,0
2125 IF PEEK (KBD)<128 THEN 2125
2130 POKE CLR,0
2149 RETURN
```

Screen Dump

```
        UP

1  34  678
= = = = = = = =
  2   5

        DOWN
```

Note: Listings 4, 5, and 6 are the Apple "Up 'N' Downs" program for the Commodore-64, CoCo and Atari respectively. The instructions for playing the game are contained in the subroutine at 2000 in the Apple version. They are not included in all the other versions.

Listing 4: For Commodore

```
5 REM PUZZLE
10 DIM P(8)
20 GOSUB 30000
100 GET N$:IF N$="" THEN 100
102 NUM=ASC(N$)-48:IF NUM<1 OR NUM>8 THEN 100
110 MOVE=MOVE+1
200 REM ---CHECK VALIDITY---
210 IF NUM=1 THEN 300
220 IF NOT(P(NUM-1)) THEN 400
230 IF NUM=2 THEN 300
250 FOR Q=NUM-2 TO 1 STEP -1:IF P(Q)=0 THEN NEXT Q:GOTO 300
260 GOTO 400
300 REM ---GOOD MOVE---
310 PRINT "■";LEFT$(H$,NUM*2+9);LEFT$(V$,P(NUM)*2+11);" ";
320 P(NUM)=NOT(P(NUM))
330 PRINT "■";LEFT$(H$,NUM*2+8);LEFT$(V$,P(NUM)*2+11);NUM;
340 FOR Q=1 TO 8:IF NOT(P(Q)) THEN NEXT Q:GOTO 500
350 GOTO 100
400 REM ---BAD MOVE---
410 POKE 54273,06:POKE 54272,75:POKE 54276,33:POKE 54277,24
420 POKE 54278,128:POKE 54296,15
430 FOR Q=1 TO 50:NEXT Q:POKE 54276,0
440 GOTO 100
500 REM ---DONE ROUTINE---
510 POKE 54272,75:POKE 54276,33:POKE 54277,24
520 POKE 54278,128:POKE 54296,15
530 FOR Q=0 TO 255:POKE 54273,Q:NEXT Q:POKE 54276,0
540 PRINT "■";LEFT$(V$,14);"    C O N G R A T U L A T I O N S !"
550 PRINT "■YOU DID IT IN■";MOVE;"■MOVES"
560 PRINT "        ■TRY AGAIN? ■(HIT ANY KEY)"
570 GET N$:IF N$="" THEN 570
580 RUN
30000 REM ---INIT---
30010 H$="■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■"
30015 V$="■■■■■■■■■■■■■■■■■■■■■■■■■"
30020 PRINT "■■";LEFT$(V$,9);
30030 PRINT ," 1 2 3 4 5 6 7 8"
30040 PRINT ,"----------------"
30050 POKE 53281,0
30060 FOR Q=1 TO 8:P(Q)=-1:NEXT Q
30090 RETURN
READY.
Z
```

Listing 5: For Color Computer

```
5 REM PUZZLE
10 DIM P(8)
20 GOSUB 30000
100 N$=INKEY$:IF N$="" THEN 100 ELSE NUM=ASC(N$)-48:
    IF NUM<1 OR NUM>8 THEN 100
110 MOVE=MOVE+1
200 REM ——CHECK VALIDITY——
210 IF NUM=1 THEN 300
220 IF NOT(P(NUM-1)) THEN 400
230 IF NUM=2 THEN 300
250 FOR Q=NUM-2 TO 1 STEP -1:IF P(Q)=0 THEN NEXT Q:GOTO 300
260 GOTO 400
300 REM ——GOOD MOVE——
310 PRINT @(P(NUM)*2+6)*32+NUM*2+5," ";
320 P(NUM)=NOT(P(NUM))
330 PRINT @(P(NUM)*2+6)*32+NUM*2+4,NUM;
340 FOR Q=1 TO 8:IF P(Q)=0 THEN NEXT Q:GOTO 500
350 GOTO 100
400 REM ——BAD MOVE——
410 SOUND 100,2:GOTO 100
500 REM ——DONE ROUTINE——
510 FOR Q=5 TO 200 STEP 5:SOUND Q,1:NEXT Q
520 PRINT @288," C O N G R A T U L A T I O N S"
530 PRINT "YOU DIO IT IN";MOVE;"MOVES"
540 PRINT "TRY AGAIN? (HIT ANY KEY)"
550 IF INKEY$="" THEN 550 ELSE RUN
30000 REM ——INIT ROUTINE——
30005 CLS
30010 PRINT @134," 1 2 3 4 5 6 7 8"
30020 PRINT @166,"————————————————"
30030 FOR Q=1 TO 8:P(Q)=-1:NEXT Q
30090 RETURN
```

Listing 6: For Atari

```
5 REM PUZZLE
10 DIM P(8)
20 GOSUB 30000
100 GET #1,NUM:NUM=NUM-48:IF NUM<1 OR NUM>8 THEN 100
110 MOVE=MOVE+1
200 REM ——CHECK VALIDITY——
210 IF NUM=1 THEN 300
220 IF  NOT (P(NUM-1)) THEN 400
230 IF NUM=2 THEN 300
250 FOR Q=NUM-2 TO 1 STEP -1:IF P(Q)=0 THEN NEXT Q:GOTO 300
260 GOTO 400
300 REM ——GOOD MOVE——
310 POSITION NUM*2,P(NUM)*2+3:PRINT #6;NUM;
320 P(NUM)= NOT (P(NUM))
330 POSITION NUM*2,P(NUM)*2+3:PRINT #6;" ";
340 FOR Q=1 TO 8:IF P(Q)=0 THEN NEXT Q:GOTO 500
350 GOTO 100
400 REM ——BAD MOVE——
410 SOUND 0,100,10,10:SOUND 1,95,10,10
412 FOR Q=1 TO 50:NEXT Q
420 SOUND 0,0,0,0:SOUND 1,0,0,0:
    SOUND 2,0,0,0:SOUND 3,0,0,0
430 GOTO 100
500 REM ——DONE ROUTINE——
510 FOR QQ=1 TO 3:FOR Q=200 TO 10 STEP -5:
    SOUND 0,Q,10,10:NEXT Q:NEXT QQ
512 SOUND 0,0,0,0
520 POSITION 0,7:PRINT #6;"ConGRatULatIOns !";
530 PRINT #6:PRINT #6;"you did it in ":
    PRINT #6;MOVE;" moves"
540 PRINT #6;"try again?":PRINT #6;
    "   (HIT ANY KEY)":POKE 764,255
550 IF PEEK(764)=255 THEN 550
560 POKE 764,255:RUN
30000 REM ——INIT——
30002 GRAPHICS 2+16
30010 FOR Q=1 TO 8:P(Q)=1:NEXT Q
30020 POSITION 0,3
30022 PRINT #6;" 1 2 3 4 5 6 7 8"
30024 PRINT #6;" ————————————————"
30030 OPEN #1,4,0,"K:"
30090 RETURN
```

MICRO

USING SIGNED ARITHMETIC ON THE 6502

by

Randall Hyde

A technique to overcome the problem of missing signed comparisons.

Almost eight years have passed since a group of Motorola employees joined MOS Technology to design the high-performance microprocessor that has been incorporated into the PET, Apple, Atari, and other microcomputer systems. Needless to say, the 6502 isn't quite state-of-the-art anymore and the newer 6502-based microcomputer owners may find the 6502's instruction set somewhat limited. True, the 6502's instruction set is lacking (especially when compared to today's high performance microprocessors), but no one can argue about the 65xx family's success in the home computer marketplace. Why did the 6502 become so popular in spite of its modest instruction set? Actually it became popular because of its modest instruction set.

The designers of the 65xx family came from Motorola after participating in the 6800 design effort. The number of basic instructions the 6800 supports is roughly twice the number of instructions supported by the 6502. Why would someone who just created an excellent microprocessor want to design a microchip with fewer instructions? The answer lies in technology. Around 1975, when the 6502 was designed, the technology wasn't anywhere near what it is today. Like today's parts, the performance and cost of a part is directly proportional to its size — particularly the cost. The 6502's designers wanted to create a chip that could be sold very inexpensively — a controller system. To reduce the cost of the 6800 they had to reduce the "die" size. Better technology and removal of several of the lesser-used instructions found on the 6800 helped. MOS Technology was able to introduce the 6501 (the forerunner to the 6502) for only $20 while the 6800 was selling for $80 and the 8080 was still selling for over $100.

The 6800 supports 16 branch instructions. The 6502's designers cut this down to eight instructions. Gone are the BRA (branch always), BSR (branch to subroutine), BGT (branch if greater than, signed), BHI (branch if greater than, unsigned), BGE (branch if greater or equal, signed), BLT (branch if less than, signed), BLE (branch if less or equal, signed), and BLS (branch if less or equal, unsigned). The BSR and BRA instructions are easily replaced with the JSR and JMP instructions (although JSR and JMP are not relocatable instructions). That leaves the BGT, BHI, BGE, BLT, BLE, and BLS instructions unimplemented on the 6502. The signed branches were removed from the 6502's instruction set because signed comparisons are rarely used in assembly language, especially in the small controller systems for which the 6502 was targeted. The missing unsigned branches are easily replaced with equivalent 6502 branches.

Unsigned Comparisons on the 6502

As many 6502 programmers are aware, the BCS and BCC instructions can be used to check for ">=" and "<", respectively. In fact many assemblers, like the LISA interactive assembler, let you enter BGE or BLT in place of BCS and BCC. For example, consider the following code:

```
LDA VAR1
CMP VAR2
BGE ISGTREQL
```

If the unsigned value contained in VAR1 is greater than or equal to the unsigned value contained in VAR2, then control is transferred to location ISGTREQL; otherwise the program continues execution at the next statement following the BGE instruction.

As long as you want to test for ">=" or "<" you're in great shape. But if you want to test for ">" or "<=" you're in trouble. The 6502's designers didn't include the BHI and BLS instructions in the 6502's instruction set. However, there is an easy way to circumvent this problem. If X > = Y, then it is also the case that Y < = X (think about it). Since you can test to see if X > = Y it stands to reason that you can test to see if Y <= X (after all, it's the same test). If you want to test for X < = Y all you have to do is check to see if Y > = X! Therefore, to see if

VAR1 is less than or equal to VAR2 you would use the code

```
LDA VAR2
CMP VAR1
BGE ISLE
```

and control will be transferred to location ISLE if VAR1 < = VAR2 and to the instruction after the BGE statement if VAR1 > VAR2.

To see if VAR1 > VAR2 apply this same reasoning to the BLT/BCC instruction. For example:

```
LDA VAR2
CMP VAR1
BLT ISGT
```

Control is transferred to location ISGT if VAR1 is greater than VAR2. The instruction after the BLT instruction is executed if VAR1 is less than or equal to VAR2.

Signed Arithmetic and the Existing Literature

Signed arithmetic isn't handled as easily. Fortunately, unsigned arithmetic is used better than 99% of the time. However, when that small 1% of the time occurs signed arithmetic can cause some real problems.

Worse than the fact that the 6502 doesn't support signed comparisons, almost all of the available literature doesn't discuss signed arithmetic, and those that do usually get it wrong. The original perpetrator of this problem is the *MOS Technology 6502 Programming Manual*. In this manual they have a table that looks something like the following:

Comparison	**N**	**Z**	**C**
A, X, or Y < Memory	1*	0	0
A, X, or Y = Memory	0	1	1
A, X, or Y >= Memory	0*	0	1

*Valid only for two's complement compare

This table implies that you can use the BMI and BPL instructions after a compare to check whether one signed value is less than, equal to, or greater than or equal to another signed operand. In reality, the N flag alone cannot be used for signed comparisons. The Rockwell R6500 programming manual uses a somewhat different table:

Comparison	**N**	**C**	**Z**	**V**
Accumulator < Memory	Either	Reset	Reset	Unchanged
Accumulator = Memory	Reset	Set	Set	Unchanged
Accumulator >= Memory	Either	Set	Reset	Unchanged

The only information concerning a signed comparison is a single cryptic sentence: "The compare instruction is designed to allow a signed comparison between two values, assuming one makes appropriate use of the Z and N and C flags." No discussion of how one makes appropriate use of the flag ensues.

In actuality, you cannot use the Z, N, and C flags to perform a signed comparison. You must use the N and V flags. Since the 6502 CMP instruction doesn't affect the V flag you cannot even use the CMP instruction to perform a signed comparison. The CMP instruction is simply a subtraction, and the 6502's SBC instruction *does* affect the V flag, so the SBC instruction can be used to perform signed comparisons.

The Two's Complement System

Most CPUs, including the 6502, use a notation known as the "two's complement" numbering system to represent signed numbers. This system (assuming an 8-bit-wide value) can represent values in the range −128 to −1 and 0 to +127.

The two's complement system uses the high order (H.O.) bit of a number to differentiate between positive and negative numbers. If the H.O. bit is clear, the number is considered to be positive and the low order (L.O.) bits contain the binary representation of the number. As long as the H.O. bit is zero, the two's complement form of a number is identical to the straight binary representation for that number.

If the H.O. bit is set, then the number is negative and the L.O. bits contain the value stored in the two's complement form. To obtain the two's complement form of a positive number you first invert all the bits and then add one to the inverted result. For example, to take the two's complement of one you would

1. Invert all the bits:
   %00000001 ——► %11111110
2. Add one to the inverted result:
   %11111110 + 1 ——► %11111111

Therefore %11111111 ($FF) is −1 in the two's complement numbering system.

The beautiful thing about the two's complement numbering system is that you can use the same addition and subtraction instructions used for unsigned arithmetic. For example, consider the addition "1 + (−1)". The expected result of zero is obtained using the 6502 ADC instruction, *if you ignore the carry flag.* For example:

```
  %11111111
+ %00000001
-----------
  %00000000 C=1
```

Subtraction works in a similar fashion. Subtracting %00000001 from %11111111 leaves you with %11111110, which is the two's complement form of −2.

There's only one problem with two's complement arithmetic. When using unsigned arithmetic the carry flag is used to detect an overflow or underflow when adding and subtracting numbers. Since the 6502 carry flag detects a carry out of the eighth bit, and we're interested in detecting an overflow by a carry from the seventh bit into the eighth bit (or underflow from the eighth bit into the seventh bit), we cannot use the carry flag to check for signed overflow. The 6502 V flag (overflow flag) detects a carry from the seventh to the eighth bit (or *vice versa*). After an addition or subtraction the V flag will be set if an overflow occurred; it will be clear if the arithmetic operation was completed successfully. Note that the V flag is always set on overflow and clear on no overflow regardless of the arithmetic operation being performed. This is in direct contrast to how the carry flag operates with the ADC and SBC instructions (the carry is clear after an ADC and set after a SBC instruction if no overflow/underflow occurred). Therefore, the BVS instruction can be used after an ADC or SBC instruction to see if signed overflow or underflow occurred. Likewise the BVC instruction can be used to branch to some location if overflow did not occur.

Two's Complement Comparisons

Although you can use the same instructions to add and subtract two signed values using the two's complement number system, on the 6502 you cannot use the CMP instruction to compare two signed values (as previously mentioned). Since the CMP instruction doesn't affect the V flag (which is necessary for signed comparisons) the SBC instruction must be used instead.

If X and Y are unsigned 8-bit values, the following code would be used to compare them:

```
LDA X
CMP Y
```

Contrast this to the code required if X and Y are signed values

```
SEC
LDA X
SBC Y
```

Simply using the SBC instruction in place of the CMP instruction is the easy part. The hard part is deciphering the condition code flags after the subtraction is performed.

Since two signed values are equal if, and only if, all their bits match, the BEQ and BNE instructions can be used to test for equality or inequality. This is identical to the test for unsigned numbers. In fact, if you are comparing two signed values to see if they are equal or not equal, you could use the CMP instruction and avoid having to use the SBC with the required SEC instruction.

To test the other inequality operations (greater than, greater than or equal, less than, and less than or equal) the SBC instruction must be used to compare the signed values. If you execute the code segment

```
SEC
LDA X
SBC Y
```

then the N and V flags will be set as follows:

```
X < = Y (N eor V) or Z = 1
X > Y (N eor V) or Z = 0
X < Y (N eor V) = 1
X > = Y (N eor V) = 0
```

Since there are no 6502 instructions to let you perform logical operations directly on the condition code flags, a series of BMI, BVS, BEQ, BPL, BVC, and BNE instructions must be used to determine whether or not a comparison is true. For example, if you want to see whether or not X > = Y you would use the code:

```
        SEC
        LDA     X
        SBC     Y
        BVC     TSTPL
        BPL     ISGE
        JMP     ISLT
;
TSTPL   BMI     ISGE
ISLT:
```

If the N and V flags are the same, then control will be transferred to the location specified by ISGE. If the N and V flags are different (N eor V = 1), then control will be transferred to the location immediately after the comparison (at the ISLT label). To check for less than (instead of greater than or equal) simply change the last two statements to

```
TSTPL   BPL     ISLT
ISGE:
```

and control will be transferred elsewhere if X < Y (to label ISLT), and control will drop through to ISGE if X is not less than Y (i.e., X > = Y).

To test for X > Y or X < = Y it is easier to compare Y to X and use the tests for greater than or equal, or less than (as described for unsigned values earlier), than attempt to test the Z, N, and V flags.

Sixteen-bit Operations

Multiprecision signed addition and subtraction is handled in a fashion identical to unsigned addition and subtraction, except you test the V flag when checking for overflow after operating on the H.O. byte.

Comparisons are only slightly more difficult; the tests for equality are identical to the tests for a 16-bit unsigned value. For example:

```
    LDA X
    CMP Y
    BNE NOTEQL
    LDA X+1
    CMP Y+1
    BNE NOTEQL
```

and:

```
    LDA X
    CMP Y
    BNE NOTEQL
    LDA X+1
    CMP Y+1
    BEQ ISEQL
NOTEQL:
```

The tests for greater than, greater than or equal, less than, and less than or equal aren't much more difficult than the equivalent 8-bit comparisons:

Test for X > = Y:

```
        LDA     X
        CMP     Y
        LDA     X+1
        SBC     Y+1
        BVC     TSTPL
        BPL     ISGE
        JMP     ISLT
;
TSTPL   BMI     ISGE
ISLT:
```

Test for X < Y:

```
        LDA     X
        CMP     Y
        LDA     X+1
        SBC     Y+1
        BVC     TSTPL
        BPL     ISGE
        JMP     ISLT
;
TSTPL   BPL     ISLT
ISGE:
```

Of course X > Y and X < = Y can be easily synthesized from these two code sequences.

Signed Input and Output

Once you can perform unsigned numeric I/O, signed I/O is trivial. Assuming you have the two routines ATOI and ITOA, which convert a character string to an integer value (ATOI, ASCII to Integer) and an integer to a character string (ITOA, Integer to ASCII), it is easy to convert these two routines to operate on signed values. Listing 1 is a subroutine that converts the two's complement integer stored in location VALUE to a character string, which is stored in STRING. Listing 2 does just the opposite; it converts the string stored in STRING to a two's complement binary value and stores the result into VALUE.

Complementing a Value

Often the need arises to *negate* a two's complement value. Either the positive version of a number must be converted to the negative version or *vice versa*. Most newcomers to assembly language follow the standard definition of a complemented number and invert all the bits and add one. For example:

```
LDA     X
XOR     #$FF
STA     X
LDA     X+1
XOR     #$FF
STA     X+1
CLC
LDA     X
ADC     #1
STA     X
LDA     X+1
ADC     #0
STA     X+1
```

Actually, there's a much simpler way to take the two's complement of a number — simply subtract it from zero. If you want to negate X you should use the code

```
SEC
LDA     #0
SBC     X
STA     X
LDA     #0
SBC     X+1
STA     X+1
```

This code performs the same function as the "invert and add" algorithm shown above.

To take the absolute value of a two's complement number you must check it to see if it is negative. If it isn't, leave the number unchanged. If it is, take the two's complement of the number to convert it to a positive number. The code to accomplish this is:

```
ABS   LDA   X+1
      BPL   0      ;Already positive,
                   no need to
                   negate.
;
      SEC
      LDA   #0
      SBC   X
      STA   X
      LDA   #0
      SBC   X
      STA   X

^0    RTS
```

Using Signed Arithmetic within Your Programs

Using signed arithmetic on the 6502 isn't the easiest task in the world. Most 6502 assembly language texts either avoid the discussion of signed arithmetic or present it incorrectly. Since signed arithmetic is rarely used this hasn't proved to be too much of a problem. Some programs I've seen that use signed arithmetic have severe problems in them. Others (like the Apple Pascal P-code interpreter) are kludged up in order to make them work. All these problems might not have occurred had MOS Technology, Synertek, and Rockwell documented the operation of the 6502 just a little better.

As previously mentioned, the 6502's designers removed the branches that let you easily perform signed comparisons in order to reduce the amount of silicon required on the 6502. Their justification was that the signed comparisons were rarely used and, when necessary, they could be emulated using existing instructions as this article has pointed out. Unfortunately, when signed comparisons must be made they are somewhat of a pain to perform. Therefore, if you can possibly get by without using signed arithmetic, by all means do so. On the other hand, when you need to perform signed arithmetic these routines are quite efficient and they do work. MICRO

Randall Hyde is vice president in charge of advanced research and development at Lazer MicroSystems, Inc., a Southern California software development firm. His text, "Using 6502 Assembly Language", is widely employed by Apple users everywhere. You may contact Mr. Hyde c/o Lazer MicroSystems, Inc., 1791 Capital, Unit G, Corona, CA 91720.

Listing 1

```
                         TTL "Listing one— ITOA Subroutine"
                 *
                 *
                 * Apple equates for the test
                 * program.
                 *
                 PUTC    EQU $FDED   Character output routine.
                 INDEX1  EPZ $80
                 *
                 *
                 *
                 * Signed output test program.
                 *
0800 A9 00               LDA #0
0802 85 80               STA INDEX1
0804 A6 80       LOOP    LDX INDEX1
0806 E0 08               CPX #LASTVAL
0808 B0 25               BGE ALLDONE
                 *
080A BD 30 08            LDA NUMBERS,X
080D 85 00               STA VALUE
080F E8                  INX
0810 BD 30 08            LDA NUMBERS,X
0813 85 01               STA VALUE+1
0815 E8                  INX
0816 86 80               STX INDEX1
                 *
                 * Convert to a string.
                 *
0818 20 38 08            JSR ITOA
                 *
                 * Output the string.
                 *
081B A2 00               LDX #0
081D B5 02       LOOP2   LDA STRING,X
081F F0 06               BEQ PRTDONE
0821 20 ED FD            JSR PUTC
0824 E8                  INX
0825 D0 F6               BNE LOOP2   Always taken
                 *
0827 A9 8D       PRTDONE LDA #$8D    Carriage return
0829 20 ED FD            JSR PUTC
082C 4C 04 08            JMP LOOP
                 *
082F 60          ALLDONE RTS
                 *
                 *
0830 64 00 C9    NUMBERS ADR 100,!-55,32000,!-2546
0833 FF 00 7D
0836 0E F6
                 LASTVAL EQU *-NUMBERS
                 *
                 *
                 *
                 VALUE   EPZ $0
                 STRING  EPZ $2
                 DIGIT   EPZ $A
                 LEADO   EPZ $B
                 *
                 * ITOA (Integer TO ASCII) converts
                 * the signed binary value stored
                 * in location VALUE to an ASCII string.
                 * The string is stored in ascending
                 * order starting at location STRING.
                 * The string is terminated with a
                 * zero byte.
                 *
                 *  Note: At least seven bytes must
                 *        be reserved for the character
                 *        string.
                 *
```

(continued)

Listing 1 *(continued)*

```
                *
                ITOA:
                *
                * Initialize the string index.
                *
0838 A0 00               LDY #0
                *
                * Check to see if the number is
                * negative.
                *
083A A5 01               LDA VALUE+1
083C 10 12               BPL NOTNEGTV
                *
                * Since VALUE is negative output
                * a "-" and take the absolute
                * value of VALUE.
                *
083E A9 AD               LDA #"-"
0840 85 02               STA STRING
0842 C8                  INY
                *
0843 38                  SEC
0844 A9 00               LDA #0
0846 E5 00               SBC VALUE
0848 85 00               STA VALUE
084A A9 00               LDA #0
084C E5 01               SBC VALUE+1
084E 85 01               STA VALUE+1
                *
                * Initialize a couple of locations
                * required by the system.
                *
0850 A2 04      NOTNEGTV LDX #4      Init digit counter
0852 85 0B               STA LEADO   Initialize LEADO to positive
                                     value
                *
0854 A9 B0      ITOA1    LDA #"0"    Initialize digit
0856 85 0A               STA DIGIT   counter for each loop.
                *
                * This loop divides VALUE by a power
                * of ten by performing a repeated subtraction.
                *
0858 38         ITOA2    SEC
0859 A5 00               LDA VALUE
085B FD 8E 08            SBC TBL10L,X  Subtract 10**X
085E 48                  PHA
085F A5 01               LDA VALUE+1
0861 FD 93 08            SBC TBL10H,X
0864 90 09               BLT ITOA3   Divison complete?
                *
0866 85 01               STA VALUE+1 Store remainder
0868 68                  PLA         back into value
0869 85 00               STA VALUE
086B E6 0A               INC DIGIT   Increment the quotient
086D D0 E9               BNE ITOA2   Always taken
                *
086F 68         ITOA3    PLA         Pop junk off of stack.
0870 A5 0A               LDA DIGIT   Get quotient
0872 E0 00               CPX #0      Is this the last digit?
0874 F0 0B               BEQ ITOA5   If so, add it to STRING
0876 C9 B0               CMP #"0"    If not, see if this is a
0878 F0 03               BEQ ITOA4   leading zero.
                *
                * If not a leading zero, set LEADO
                * to a negative value.
                *
087A 38                  SEC
087B 66 0B               ROR LEADO
                *
087D 24 0B      ITOA4    BIT LEADO   Is this a leading zero?
087F 10 04               BPL ITOA6
0881 99 02 00   ITOA5    STA STRING,Y If not, add to string
0884 C8                  INY
                *
0885 CA         ITOA6    DEX         Repeat for each power of 10
0886 10 CC               BPL ITOA1
                *
                *
0888 A9 00               LDA #0      Zero terminate the string
088A 99 02 00            STA STRING,Y
088D 60                  RTS
                *
                *
                *
                * Power of ten tables.
                *
088E 01 0A 64   TBL10L   BYT 1,10,100,1000,10000
0891 E8 10
0893 00 00 00   TBL10H   HBY 1,10,100,1000,10000
0896 03 27
                         END
```

Listing 2

```
                         TTL "Listing two— ATOI Subroutine"
                *
                *
                *
                VALUE    EPZ $0
                SIGN     EPZ VALUE+2
                STRING   EPZ SIGN+1
                *
                *
                * CR is a constant representing
                * the carriage return character.
                *
                CR       EQU $8D
                *
                *
                *
                *
                * ATOI converts the string stored
                * in location STRING to a signed
                * two's compliment integer.
                * The resulting integer value is
                * stored into the two locations
                * VALUE and VALUE+1.
                *
                *   Note: the numeric string can
                * have any number of leading blanks
                * and it must be terminated with
                * a space, carriage return, comma,
                * or zero byte.
                *
                *   Error codes returned in the
                * condition code flags:
                *
                *   N)  Set: Illegal initial character
                *            (a digit or "-" wasn't encountered
                *             at the beginning of the line or
                *             a digit did not immediately follow
                *             the minus sign).
                *
                *   V)  Set: Attempted to convert a value
                *            outside the range -32767..32767.
                *
                *   C)  Set: Numeric value was terminated
                *            with a character other than a
                *            space, comma, carriage return
                *            or zero byte.
                *
                *
                *  The N and V flags SHOULD be tested
                * after calling ATOI.  Testing the
                * C flag is optional if you don't mind
                * allowing other non-digit characters
                * to terminate the number.
                *
                *
                *   On return:
                *
                *      VALUE and VALUE+1 contain the
                *    converted integer value.
                *
                *      The X-register points at the
                *    delimiter for the current value.
                *
                *
0800 A2 00      ATOI     LDX #0      Init index into STRING
0802 86 02               STX SIGN    Assume a positive number
```

(continued)

Listing 2 *(continued)*

```
0804 86 00             STX VALUE   Init VALUE to zero
0806 86 01             STX VALUE+1
0808 20 6F 08          JSR BLKDEL  Skip leading blanks
               *
               * Check the first character to see
               * if it is a "-".  If so, set the
               * SIGN variable to a non-zero value.
               *
080B C9 AD             CMP #"-"
080D D0 05             BNE 0
               *
080F 85 02             STA SIGN    Non-zero for negative
0811 E8                INX
0812 B5 03             LDA STRING,X Get the next character
               *
0814 20 78 08          JSR TSTDEC  Make sure this is
0817 90 05             BCC ISDEC   a decimal digit.
               *
               * If the first character was illegal
               * return with the N flag set.
               *
0819 18                CLC
081A B8                CLV
081B A9 80             LDA #$80    Set N flag
081D 60                RTS
               *
               *
               *
               * If this number has all the
               * beginnings of a good decimal
               * value, convert it.
               *
081E B5 03     ISDEC   LDA STRING,X
0820 20 78 08          JSR TSTDEC  Is the current char
0823 B0 1E             BCS NOTDEC  a numeric character?
               *
               * If so, multiply VALUE by ten
               * and add in this digit.
               *
0825 20 7F 08          JSR MUL10
0828 70 13             BVS OVRFLW
               *
082A 29 0F             AND #$F     Convert ASCII to BINARY
082C 18                CLC
082D 65 00             ADC VALUE
082F 85 00             STA VALUE
0831 A5 01             LDA VALUE+1
0833 69 00             ADC #0
0835 85 01             STA VALUE+1
0837 70 04             BVS OVRFLW  Check for signed overflow
               *
0839 E8                INX         Set up for next character
083A 4C 1E 08          JMP ISDEC   and repeat.
               *
083D 18        OVRFLW  CLC
083E 2C 42 08          BIT SETOVFL Set V and clear N
0841 60                RTS
               *
0842 40        SETOVFL BYT $40
               *
               *
               * NOTDEC is branched to if a non-digit
               * was encountered on the line.
               * At this point the numeric conversion
               * is complete.  All that remains to
               * do is make sure that the number
               * is terminated with a space, comma,
               * carriage return, or zero byte.
               *
0843 C9 A0     NOTDEC  CMP #" "    Check space
0845 F0 12             BEQ DECOK
0847 C9 AC             CMP #","    Check comma
0849 F0 0E             BEQ DECOK
084B C9 8D             CMP #CR     Check carriage return
```

(continued)

Listing 2 *(continued)*

```
084D F0 0A              BEQ DECOK
084F C9 00              CMP #0      Check zero
0851 F0 06              BEQ DECOK
                *
                * If the program falls through to
                * this point the number is terminated
                * with an illegal delimiter.  The
                * probable cause is a typo.  Report
                * the problem by returning with the
                * carry flag set.
                *
0853 2C 58 08           BIT CLRNV   Clear N & V flags
0856 38                 SEC
0857 60                 RTS
                *
0858 00         CLRNV   BYT 0
                *
                *
                *
                * DECOK: This code is executed
                * if the number is correct in
                * every sense.
                *
0859 A5 02      DECOK   LDA SIGN    If non-zero then
085B F0 0D              BEQ 0       VALUE must be negated.
                *
085D 38                 SEC
085E A9 00              LDA #0
0860 E5 00              SBC VALUE
0862 85 00              STA VALUE
0864 A9 00              LDA #0
0866 E5 01              SBC VALUE+1
0868 85 01              STA VALUE+1
                *
086A 2C 58 08           BIT CLRNV   Clear N & V
086D 18                 CLC         No input error
086E 60                 RTS
                *
                *
                *
                *
                **********************************
                *                                *
                *                                *
                *    **********************      *
                *    *                    *      *
                *    * Utility Subroutines *     *
                *    *                    *      *
                *    **********************      *
                *                                *
                *                                *
                **********************************
                *
                *
                * BLKDEL- skips any leading blanks
                *         in the STRING buffer.
                *
086F CA         BLKDEL  DEX
0870 E8         BLKDLP  INX
0871 B5 03              LDA STRING,X
0873 C9 A0              CMP #" "
0875 F0 F9              BEQ BLKDLP
0877 60                 RTS
                *
                *
                *
                * TSTDEC- tests the character in
                *         the accumulator to see
                *         if it is a decimal digit
                *         or not.  If it is, the
                *         carry flag is returned
                *         clear.  If the character
                *         isn't a decimal value
```

(continued)

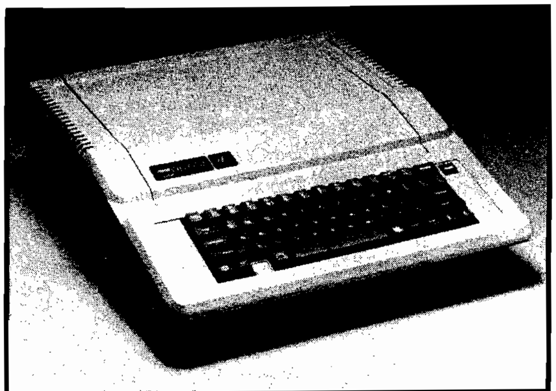
Listing 2 *(continued)*

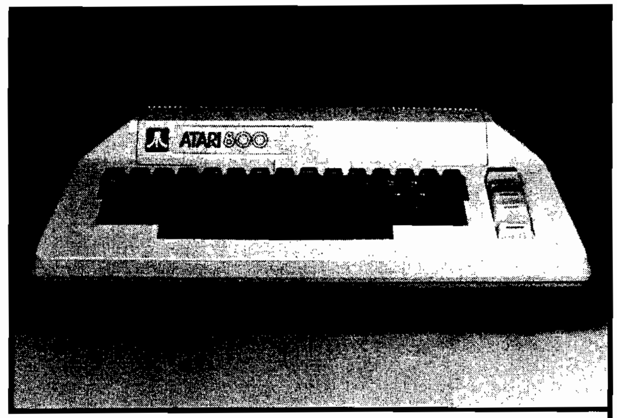
```
                *         then the carry flag is
                *         returned set.  The Acc
                *         is returned unchanged.
                *
0878 49 B0      TSTDEC   EOR #"0"    Map "0".."9" to 0..9
087A C9 0A               CMP #10     Carry set if Acc = 10
087C 49 B0               EOR #"0"    Restore Acc
087E 60                  RTS
                *
                *
                * MUL10 multiplies VALUE by 10.
                * If an arithmetic overflow occurs
                * return with the V flag set.
                *
087F 48         MUL10    PHA         Save Acc
0880 8A                  TXA         X-, and Y-register
0881 48                  PHA         values.
0882 98                  TYA
0883 48                  PHA
                *
0884 06 00               ASL VALUE   Multiply VALUE
0886 26 01               ROL VALUE+1  by two
0888 30 21               BMI OVERFLOW  Check for signed overflow
                *
088A A5 00               LDA VALUE
088C A4 01               LDY VALUE+1
                *
088E A2 02               LDX #2
0890 06 00      SHFTLP   ASL VALUE   Multiply VALUE
0892 26 01               ROL VALUE+1  by four to give
0894 30 15               BMI OVERFLOW  Check for signed overflow
0896 CA                  DEX         VALUE * 8.
0897 D0 F7               BNE SHFTLP
                *
                * Add it 2*VALUE to 8*VALUE to
                *         get 10*VALUE.
                *
0899 18                  CLC
089A 65 00               ADC VALUE
089C 85 00               STA VALUE
089E 98                  TYA
089F 65 01               ADC VALUE+1
08A1 85 01               STA VALUE+1
08A3 70 06               BVS OVERFLOW
                *
                * At this point things look good,
                * return to the calling procedure.
                *
08A5 68                  PLA
08A6 A8                  TAY
08A7 68                  PLA
08A8 AA                  TAX
08A9 68                  PLA
08AA 60                  RTS
                *
                *
08AB 68         OVERFLOW PLA
08AC A8                  TAY
08AD 68                  PLA
08AE AA                  TAX
08AF 68                  PLA
08B0 2C 42 08            BIT SETOVFL
08B3 60                  RTS
                         END
```

MICRO™

A Product Catalog for Apple and Atari

The final installment in our product catalog series.





Atari

Expansion Memory

16K/32K Expander, Mosaic Electronics, $74.95

32K RAM Board, Mosaic Electronics, $87.95

48K RAM Kit, Mosaic Electronics, $115

48K RAM Board, Intec Peripherals Corp., $240

48K RAM Board, Tara Computer Products, $199

48K RAM Board, Neotechnic Industries Inc., $129.95

48K Austin Board, Austin Franklin Associates, $114.95

48400 48K RAM Upgrade, Polly Products, $89.99

48K/52K RAM Board, Newell Industries, $159.95, $139.95 with 16K trade-in

64K RAM Select, Mosaic Electronics, $149

RAMDISK Memory System, Axlon Inc., 128K, $699

FASTCHIP, Newell Industries, ROM replacement, $39.95

Modem

MICROCONNECTION, The Microperipheral Corporation, Autodial/Auto-answer, $199.50 up

Smartmodem, Hayes Microcomputer Products, $279

Disk Drive

Percom RFD DD disk drives, Percom Data Company, $699 first-drive, $399 add-on

AT-88 SD disk drive, Percom Data Company, $488

Rana 1000 disk drive, Computer Creations Inc., $419

Micro Mainframe Floppy Disk Drive, Micro Mainframe, less than $450

FAST-CHIP, BiNARY Computer Software, Disk drive upgrade, $39.95

ATR8000 Disk Interface, SWP Inc., $499.95

Happy 810 Drive Enhancement, Happy Computing, hardware kit, $249.95

Hardware Cartridge

EPROM Board, Elcomp Publishing Inc., $29.95

EPROM Board Kit, Elcomp Publishing Inc., $14.95

EPROM Board, Radical Systems, $20

Accessories

ECHO Speech Synthesizer, Street Electronics, $369.95

Voice Box, The Alien Group, $169

Voice Box II, The Alien Group, $169

VersaWriter Drawing Tablet, Versa Computing Inc., $299

FULL-VIEW 80 Display Card, BIT 3 Computer Corporation, $179

Austin 80 Color Video Board, Austin Franklin Associates, $279.95

Series B Pro-Stick, Game-Tech, $39.99

Fingertip Controller, KY Enterprises, $19.95, kit $14.95

Joystick Replacement Insert, J.E. Koch & Co., $4.35 pair

Tara 400 Keyboard, Tara Computer Products, price n/a

JoyTyper 400, Microtronics, $129.95

EPROM Burner, Elcomp Publishing Inc., $179

EPROM Burner Kit, Elcomp Publishing Inc., $49

Cartridge Maker EPROM Burner, Radical Systems, $79

The Bytewriter PROM Burner, Convologic Inc., $149.95

Custom Printers and Interfaces

EPSON Printer Interface, Elcomp Publishing Inc., $19.95

RS-2332 Interface, Elcomp Publishing Inc., $19.95

Interface No. 1, Looking Glass Microproducts, $85

Operating System

Ramrod MMOS, Newell Industries, $159.95

M/L Debugger

MMG BASIC Debugger, MMG Micro Software, $34.95

ATMONA-2, Elcomp Publishing Inc., $49.95 cassette, $54 disk

BUG/65, Optimized Systems Software Inc., $34.95

M/L Monitor

ATMONA-1, Elcomp Publishing Inc.,

$19.95 cassette, $24.95 disk, $59 cartridge

Omnimon!, CDY Consulting, Resident monitor, $99.95

Assembler

ATMAS, Elcomp Publishing Inc., Macro-assembler, $89 disk, $129 cartridge

ATAS, Elcomp Publishing Inc., Cassette-based assembler without macros, $49.95

MAC/65, Optimized Systems Software Inc., $80

MAE Assembler, Eastern House, includes editor, $99.95

Edit 6502, LJK Enterprises Inc., includes M/L monitor, price n/a

Languages

BASIC Commander, MMG Micro Software, $34.95

FORTH, Elcomp Publishing Inc., $39.95, disk

Tiny C, Optimized Systems Software Inc., $99.95

BASIC A+, Optimized Systems Software Inc., $80

C/65, Optimized Systems Software Inc., C language compiler, $80

ABC, Monarch Data Systems, $69.95

The BASIC Compiler, Datasoft, $99.95

BASM, Computer Alliance, $99.95

Word Processor

Textwizard, Datasoft, $99.95

Letter Perfect, LJK Enterprises Inc., $149.95

Letter Perfect ROM, LJK Enterprises Inc., $249.95

AtariWriter, Atari Inc., less than $100

Alog Pagewriter, Alog Computing, $39.95

Bank Street Writer, Broderbund Software, $69.95

ATEXT-1, Elcomp Publishing Inc., $29.95 cassette, $34.95 disk, $69 cartridge

Data Base

File Manager 800, Synapse Software, $84

File Manager+, Synapse Software, $99.95

File-Fax, TMQ Software, $129.95

Data Perfect, LJK Enterprises Inc., $99.95

FILE-IT, Swifty Software Inc., $34.95

FILE-IT 2, Swifty Software Inc., $49.95

MMG Data Manager, MMG Micro Software, $49.95 disk

CCA Data Management System, CE Software, $99.50

Mailing List

Magic Mail, ABBS Software, $59.95

Mailing List, Datasoft, $24.95

Mailing List, Elcomp Publishing Inc., $19.95 cassette, $24.95 disk

Communications Package

Telelink, Atari Inc., $29.95

TeleTari, Don't Ask, $39.95

Chameleon, Atari Program Exchange, $17.95

Atari I/O Package, Mosaic Electronics, $18

Datalink, Swifty Software, $39.95

Downloader, Computer Age, $24.95

T.H.E. Smart Terminal, BiNARY Computer Software, $49.95

TSmart, The Microperipheral Corporation, $79.95

Business Package

Miles Payroll System, Miles Computing, $179.95

Invoice Writing for Small Business, Elcomp Publishing Inc., $29.95 cassette, $39.95 disk

Microinv, Compumax, Inventory control, $140

Inventory Control, Elcomp Publishing Inc., $19.95 cassette, $24.95 disk

A Financial Wizard 1.5, ON LINE Computer Centers, $59.95

General Ledger System, FCC Inc., $149.95

Budgetmaster, Sunrise Software, $29.95 cassette, $34.95 disk

Atari Addresses

Mosaic Electronics
P.O. Box 748
Oregon City, OR 97045

Intec Peripherals Corp.
906 E. Highland Ave.
San Bernardino, CA 92404

Tara Computer Products
3648 Southwestern Blvd., Dept. S.
Orchard Park, NY 14127

Neotechnic Industries, Inc.
P.O. Box 277
Redondo Beach, CA 90277

Optimized Systems Software, Inc.
10379 Lansdale Ave.
Cupertino, CA 95014

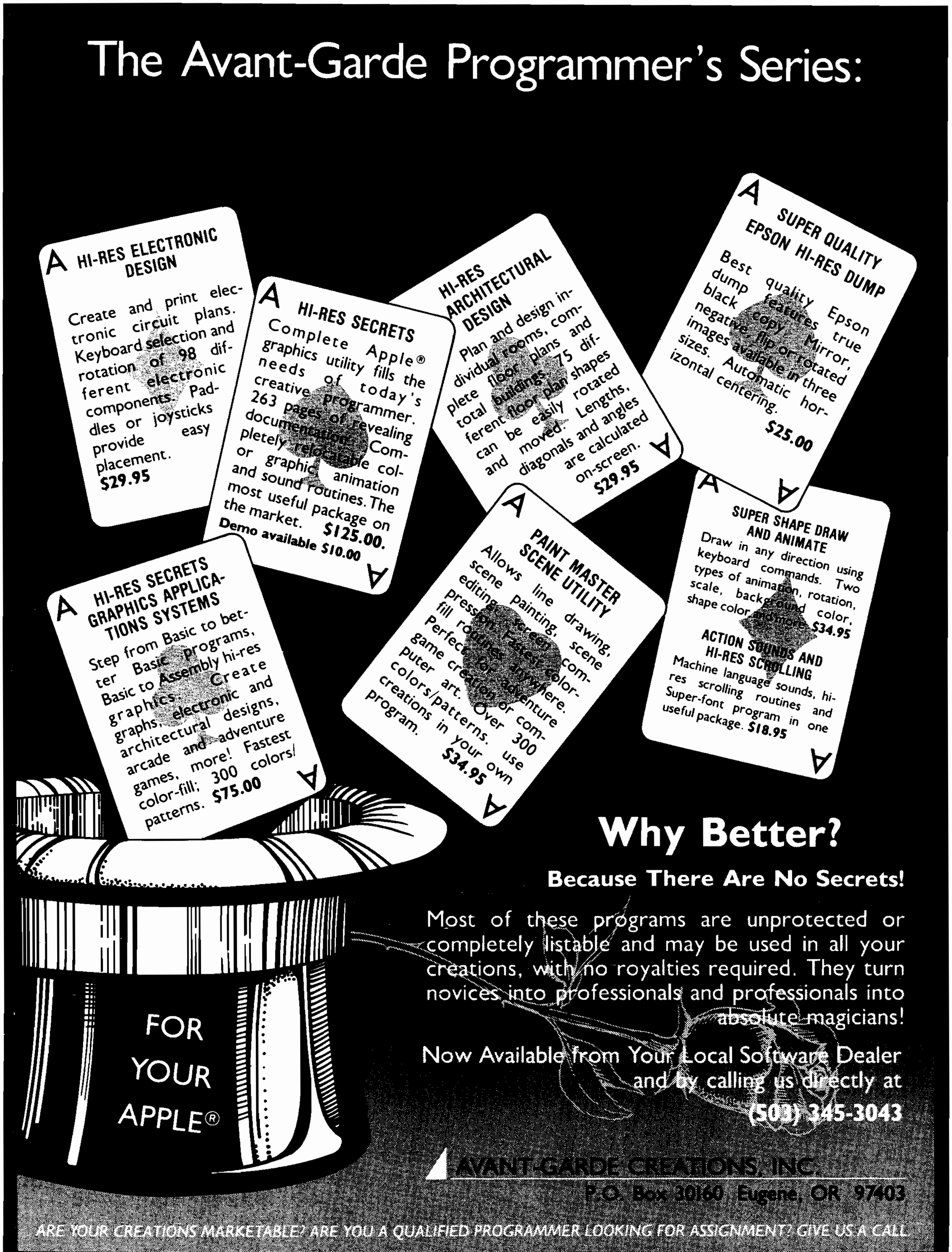
Miles Computing
7136 Haskell Ave. #204
Van Nuys, CA 91406

Austin Franklin Associates
43 Grove Street
Ayer, MA 01432

Polly Products
P.O. Box 8485
Riverside, CA 92515

Newell Industries
3340 Nottingham Ln.
Plano, TX 75074

Axlon, Inc.
170 N. Wolfe Rd.
Sunnyvale, CA 94086


The Avant-Garde Programmer's Series:
A HI-RES ELECTRONIC DESIGN
Create and print electronic circuit plans. Keyboard selection and rotation of 98 different electronic components. Paddles or joysticks provide easy placement.
$29.95
A HI-RES SECRETS
Complete Apple® graphics utility fills the needs of today's creative programmer. 263 pages of revealing documentation. Completely relocatable color graphic animation and sound routines. The most useful package on the market. $125.00.
Demo available $10.00
HI-RES ARCHITECTURAL DESIGN
Plan and design individual rooms, complete floor plans and total buildings. 75 different floor plan shapes can be easily rotated and moved. Lengths, diagonals and angles are calculated on-screen.
$29.95
A SUPER QUALITY EPSON HI-RES DUMP
Best quality Epson dump features true black copy. Mirror, negative, flip or rotated images available in three sizes. Automatic horizontal centering.
$25.00
A HI-RES SECRETS GRAPHICS APPLICATIONS SYSTEMS
Step from Basic to better Basic programs, Basic to Assembly hi-res graphics. Create graphs, electronic and architectural designs, arcade and adventure games, more! Fastest color-fill; 300 colors/patterns. $75.00
A PAINT MASTER SCENE UTILITY
Allows line drawing, scene painting, scene editing, screen compression. Fastest color-fill routines anywhere. Perfect for adventure game creation or computer art. Over 300 colors/patterns. use creations in your own program.
$34.95
A SUPER SHAPE DRAW AND ANIMATE
Draw in any direction using keyboard commands. Two types of animation, rotation, scale, background color, shape color and more. $34.95
ACTION SOUNDS AND HI-RES SCROLLING
Machine language sounds, hi-res scrolling routines and Super-font program in one useful package. $18.95
FOR YOUR APPLE®
Why Better?
Because There Are No Secrets!
Most of these programs are unprotected or completely listable and may be used in all your creations, with no royalties required. They turn novices into professionals and professionals into absolute magicians!
Now Available from Your Local Software Dealer and by calling us directly at
(503) 345-3043
AVANT-GARDE CREATIONS, INC.
P.O. Box 30160 Eugene, OR 97403
ARE YOUR CREATIONS MARKETABLE? ARE YOU A QUALIFIED PROGRAMMER LOOKING FOR ASSIGNMENT? GIVE US A CALL

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

AVAILABLE NOW! Call 800 227-9998**
For the name of your nearest dealer, detailed information or a catalog of products.

***California, Canada and Alaska, please call (415) 965-1735.*

In Europe: SUPERSOFT, Winchester House, Canning Road, Harrow Wealdstone, England HA3 7SJ, Tel. 01 861 1166

The Microperipheral Corporation
2643 151st Pl. N.E.
Redmond, WA 98052

Hayes Microcomputer Products, Inc.
5835 Peachtree Corners East
Norcross, GA 30092

Percom Data Company, Inc.
11220 Pagemill Rd.
Dallas, TX 75243

Computer Creations, Inc.
P.O. Box 292467
Dayton, OH 45429

Micro Mainframe
11325 Sunrise Gold Circle
Building E
Rancho Cordova, CA 95670

BiNARY Computer Software
3237 Woodward Ave.
Berkeley, MI 48072

SWP, Inc.
2500 E. Randol Mill Rd., Suite 125
Arlington, TX 76011

Happy Computing
P.O. Box 32331
San Jose, CA 95152

Elcomp Publishing, Inc.
53 Redrock Lane
Pomona, CA 91766

Radical Systems
2002 Colice Road, S.E.
Huntsville, AL 35801

Street Electronics Corporation
3152 East LaPalma Avenue, Suite D
Anaheim, CA 92806

The Alien Group
27 West 23rd Street,
New York, NY 10010

Versa Computing, Inc.
3541 Old Conejo Road, Suite 104
Newbury, CA 91320

BIT 3 Computer Corporation
8120 Penn Avenue South, Suite 548
Minneapolis, MN 55431

Game-Tech
283 Broadway
Arlington, MA 02174

KY Enterprises
195 Claremont, Suite 288
Long Beach, CA 90803

Microtronics, Inc.
P.O. Box 8894
Fort Collins, CO 80525

Convologic, Inc.
2800 West State Road 434, Suite 1276
Longwood, FL 32750

Looking Glass Microproducts
P.O. Box 5084
Loveland, CO 80537

MMG Micro Software
P.O. Box 131
Marlboro, NJ 07746

Optimized Systems Software, Inc.
10379 Lansdale Ave.
Cupertino, CA 95014

CDY Consulting
421 Hanbee
Richardson, TX 75080

Eastern House
3239 Linda Dr.
Winston-Salem, NC 27106

Monarch Data Systems
P.O. Box 207
Cochituate, MA 01778

Datasoft Inc.
19519 Business Center Drive
Northridge, CA 91324

Computer Alliance
21115 Devonshire Street
Suite 132
Chatsworth, CA 91311

LJK Enterprises, Inc.
P.O. Box 10827
St. Louis, MO 63129

Atari, Inc.
1272 Borregas Ave.
Sunnyvale, CA 94086

Alog Computing
1040 Veronica Springs Rd.
Santa Barbara, CA 93105

Broderbund Software
1938 Fourth St.
San Rafael, CA 94901

Synapse Software
820 Coventry Road
Kensington, CA 94707

TMQ Software
82 Fox Hill Drive
Buffalo Grove, IL 60090

Swifty Software, Inc.
P.O. Box 641
Melville, NY 11747

CE Software
238 Exchange St.
Chicopee, MA 01013

ABBS Software
P.O. Box 28
Laurel, MD 20707

Don't Ask Computer Software
2265 Westwood Blvd., Suite B-150
Los Angeles, CA 90064

Atari Program Exchange
P.O. Box 427
155 Moffett Park Drive
Sunnyvale, CA 94086

Computer Age
Silver Spring, MD

Miles Computing
7136 Haskell Ave. #204
Van Nuys, CA 91406

ON LINE Computer Centers of OKC
10944 North May
Oklahoma City, OK 73120

FCC, Inc.
4712 Chastant St.
Metairie, LA 70002

Sunrise Software
12800 Eastwood Blvd.
Cleveland, OH 44125

Apple

Apple Peripherals

Nearly 20,000 pieces of software and a tremendous amount of hardware add-ons are available for the Apple. Therefore we are providing only a resource of books that list software and hardware for the Apple II, Apple II+, and Apple IIe.

The best book, a must for all Apple owners:

The Apple II Blue Book
WIDL Video
5245 West Diversy
Chicago, IL 60639

The Software Catalog
Elsevier Science Publishing Co., Inc.
52 Vanderbilt Ave.
New York, NY 10017

Software Directory
PC Clearinghouse, Inc.
11781 Lee Jackson Highway
Fairfax, VA 22033

Directory of Educational Computing Resources
Classroom Computer News
Intentional Educations, Inc.
341 Mt. Auburn St.
Watertown, MA 02172

Directory of Educational Software
Sterling Swift Publishing Co.
1600 Fortview Rd.
Austin, TX 78704

VanLoves Apple Software Directory
Vital Information, Inc.
7899 Mastin Drive
Overland Park, KA 66204

Skarbek Software Directory
11990 Dorsett Rd.
St. Louis, MO 63043

The Addison-Wesley Book of Apple Computer Software
The Book Company
16720 Hawthorne Blvd.
Lawndale, CA 90260

International Microcomputer Software Directory
Imprint Software
420 South Howes St.
Fort Collins, CO 80521

MICRO

MACHINE LANGUAGE INPUT ROUTINES FOR COMMODORE COMPUTERS

The beginning machine-language programmer will often find that input/output routines are the most difficult to write. More often than not, the actual computation part of a program is straightforward; it's getting the required data to and from the computer that is the hard part. Output routines have been treated previously, so this article will concentrate on effective ways to input data to a Commodore computer. In particular, machine-language programming of input routines for the CBM-8032 is discussed. However, all of these routines are available on the other Commodore computers, including the VIC-20.

Perhaps the easiest way to input data is directly from the keyboard. We can do this one byte at a time with the GET A BYTE routine, located at $FFE4. (This is a kernal routine; it will be located at the same place on all Commodore computers.) The principle of operation is quite simple. Upon being called, this routine will determine if a key is depressed or not. If no key is depressed, the zero flag in the status register will be set. If, on the other hand, a key is depressed then the zero flag is cleared and the accumulator will contain the ASCII code for that key.

Generally, you will want to imbed this routine in a loop (just like the GET statement in BASIC), so that the keyboard will be continually checked until a key has been bound. Figure one shows an example of this.

While this routine is short and simple, it does have several drawbacks. First of all, when the routine is called, the cursor vanishes. In effect, the screen has been disabled. Even when a key is depressed there is no visual feedback since the character is not reflected to the screen. (You can reflect it to screen yourself, if you wish, by calling routine $FFD2, the OUTPUT A BYTE routine.) In addition, if you type a mistake, there is no chance to catch it and use the excellent screen editing features of the Commodore computers to correct it.

The INPUT A BYTE routine, at $FFCF does allow these features. When this routine is called, the cursor is present. The presence of the blinking cursor usually pacifies the neophyte; it gives a visual indication that the system hasn't crashed. With the cursor present, the user may enter the desired input information. If a mistake is made, the [delete], [insert], and cursor movement keys may be used to correct the error. When everything is right, the user can then hit [return] and the data will be input. As you can tell, this is much more "user friendly." So, unlike the GET A BYTE routine, the INPUT A BYTE routine actually inputs data from the screen, not the keyboard.

Figure two gives a simple example. When the subroutine is called, the X register is loaded with a zero. At this

The three machine-language methods presented here allow you to input data (both string and numeric) to a Commodore computer. These methods use ROM routines inherent to the computer, and consequently consume very little additional memory.

BY THOMAS HENRY

point the cursor will appear and all computing will stop until a string has been input. A carriage return, (hex $0D) indicates that it is time to move again. The main loop will now go into effect, taking one byte at a time from the input string and storing it in the buffer. When the carriage return is found at the end of the string, the loop concludes. In this example a zero byte is used to indicate the end of the string; it may be that your intended application won't need this.

There is nothing sacred about the buffer used in this example. You may store the input string anywhere in memory. Likewise there is no reason why the X register must be used as the index counter. If you need to use indirect addressing, for example, the Y register would be the one to use.

Thus far, the two input routines have been generalized in the sense that they will work with any character and don't require any interaction with BASIC. This makes them perfect for writing monitors, assemblers, disassemblers, and so on. However, even in machine-language programming, there are times when you will wish to interact with BASIC in a more intimate way. For instance, if you are writing a "wedge" for your system, you may need to input some parameters which BASIC would then use. A good example of this is a RENUMBER utility. The command may read RENUMBER 100,10, where RENUMBER is the command, 100 is the first line number of the new numbering scheme, and 10 is the increment between successive lines. In this case we need to input not only an alphabetic string (RENUMBER) but also some integer parameters (100, 10). BASIC will then take over and use these parameters (100, 10). BASIC will then take over and use these parameters to perform the RENUMBERing.

So how do we input data for BASIC to use? The key is the well known CHRGET and CHRGOT routines, located at $0070 and $0076, respectively. These routines, which are used constantly by your Commodore computer, check for numerics, alphabetics, spaces, colons, and null characters. Strictly speaking, when these routines are called by the computer during the execution of a program, they are not really input routines. However, when used in the immediate mode they do become input routines in the sense that they take input from the user and process it.

The CHRGET and CHRGOT routines have been covered countless times in the past. Instead of repeating this information, we will instead look at how these useful routines can be combined with another to form an integer inputting routine.

Our goal is to be able to input a decimal integer and have it accepted. This not a trivial matter. Remember, when we type in a decimal number, we are really entering an ASCII string, not a strict number. We need to convert this ASCII string to the proper binary integer form, and the routine at $B8F6 (in conjunction with the CHRGOT routine) will do this. Refer to figure three. This listing should be appended to the listing in figure two; the combined listing is then a complete integer input routine.

If we have executed the routine in figure two, we then enter figure three with the input buffer (at $0200) containing an ASCII representation of a decimal number. The CHRGOT pointer (at $77 and $78) is then set to point to the start of the buffer. Next the CHRGOT routine is called. This has the effect of getting the digits (in ASCII form), one by one and will stop when a zero byte is encountered.

Next the ACCEPT AN INTEGER routine is called, and this will convert the string to true binary form. The result is deposited in $11 (low byte) and $12 (high byte).

As mentioned before, there is nothing particularly special about the input buffer. You could just as easily point the CHRGOT pointer to any address in memory.

Of course this routine (as presented in figure two and three) is a bare-bones approach. No error detection has been built in. This is easy to implement, though. For example, suppose a user inputs the gibberish " + @[i8U" and calls the routine. What will happen? As it turns out, the CHRGOT routine looks for this and signals the ACCEPT AN INTEGER routine that what follows is not a decimal integer. The program will end with zero bytes being loaded into locations $11 and $12.

Another common cause of error is overflow. Only decimal integers between -1 and 64000 may be input. Any other entry will spur on a "?syntax error" message from the BASIC operating system. In both of these error conditions (entry of gibberish or number out of range) the system will not crash; in this sense the program is protected.

These three routines should simplify your own work in machine-language input programming. However, this is hardly the final word on the subject. I discovered these routines by trial and error. More input routines undoubtedly exist in your Commodore computer; why don't you let others know the results of your experimentation through the pages of MICRO!

GET A BYTE ROUTINE

```
                        GETBYT = $FFE4        ; ROUTINE TO GET A SINGLE BYTE
                        ;
                        ;
                                *=$5000
5000   20 E4 FF         START   JSR GETBYT    ; START OF GET LOOP
5003   C9 00                    CMP #$00      ; IS ZERO FLAG SET?
5005   F0 F9                    BEQ START     ; YES, SO NO KEY IS DEPRESSED.
5007                    ;
5007                    ;
5007                    ; AT THIS POINT, THE ACCUMULATOR NOW CONTAINS
5007                    ; THE ASCII EQUIVALENT OF THE KEY DEPRESSED.
5007                    ; YOU ARE FREE TO READ THIS VALUE, STORE IT
5007                    ; FOR LATER USE, COMPUTE WITH IT, ETC..
```

INPUT ROUTINE

Input Routines requires:
Any Commodore Computer

```
                        INPUT  = $FFCF
                        BUFFER = $0200
                        ;
                        ;
                                *=$5000
5000   A2 00                    LDX #$00      ;X REGISTER IS USED AS AN INDEX
5002   20 CF FF         LOOP    JSR INPUT     ;INPUT ONE BYTE FROM SCREEN
5005   C9 0D                    CMP #$0D      ;IS IT A CARRIAGE RETURN?
5007   F0 06                    BEQ END       ;YES, SO END EVERYTHING.
5009   9D 00 02                 STA BUFFER,X  ;NO, STORE THE BYTE
500C   E8                       INX           ;INCREMENT THE INDEX COUNTER
500D   D0 F3                    BNE LOOP      ;JMP (X IS NEVER ZERO NOW)
500F   A9 00            END     LDA #$00
5011   9D 00 02                 STA BUFFER,X  ;ZERO BYTE = END OF STRING
5014                    ;
5014                    ;
5014                    ; AT THIS POINT, THE BUFFER NOW CONTAINS THE
5014                    ; INPUT STRING.  THE STRING STARTS AT $0200
5014                    ; AND CONTINUES ON THROUGH THE MEMORY, WITH
5014                    ; A ZERO BYTE TERMINATING THE STRING.
5014                            .END
```

The GET A BYTE and INPUT A BYTE routines are kernal routines. This means that the call addresses are the same for all PET'S, CBM's, the VIC-20, and the Commodore 64. The CHRGET and CHRGOT addresses are the same for PET's with 2.0 ROM's and 4.0 ROM's. In 1.0 ROM's, the addresses are $C2 and $C8, respectively. In the VIC-20 and Commodore 64 these addresses are $73 and $79. For 1.0 ROM's the ACCEPT AN INTEGER routine is located at $C863 and the result is stored at $08 and $09. For 2.0 ROM's the routine address is $C873 and the result is stored the same way as presented in the article (i.e., the same as for 4.0 ROM's). The VIC-20 and Commodore 64 have the ACCEPT AN INTEGER routine at $C96B and the result is stored at $14 and $15.

INTEGER INPUT ROUTINE

```
                    CHRGOT = $0076
                    POINTR = $77
                    INTEGR = $B8F6
                    ;
                    ;
                    ; THIS ROUTINE MUST FOLLOW THE ROUTINE DESCRIBED
                    ; IN FIGURE TWO.  NOTE THAT THE ADDRESSES TAKE UP
                    ; WHERE THEY LEFT OFF IN THAT ROUTINE.  THE
                    ; ROUTINE IN FIGURE TWO PUTS THE ASCII STRING
                    ; EQUIVALENT OF THE INTEGER INTO THE INPUT BUFFER;
                    ; THIS ROUTINE CONVERTS THE STRING TO A BINARY INTEGER.
                    ; THE RESULT IS THEN DEPOSITED INTO $11 (LOW BYTE) AND
                    ; $12 (HIGH BYTE).
                    ;
                    ;
                         *=$5014
5014  A9 00              LDA #$00         ;SET THE CHRGOT POINTER
5016  85 77              STA POINTR       ;TO POINT TO THE
5018  A9 02              LDA #$02         ;INPUT BUFFER (AT $0200)
501A  85 78              STA POINTR+1
501C  20 76 00           JSR CHRGOT       ;FETCH THE DIGITS AND
501F  20 F6 B8           JSR INTEGR       ;CONVERT TO A BINARY INTEGER
5022                     .END
```

References

1. T. Henry, *"Machine Language Screen Utilities for the CBM-8032"*, MICRO, October 1982.
2. N. Hampshire, *THE PET REVEALED*, (Slough, Bershire, England: Commodore Business Machines (UK) Limited, 1980), pp. 78-82.
3. W. Kolbe, *"Define Your Own Function Key on PET"*, MICRO, October 1980, pp. 19-20.
4. B. Seiler, *"The Append Wedge"*, PET USER'S CLUB NEWSLETTER, June 1979, pp. 24-30.

Thomas Henry is a professional writer in the areas of electronic music, circuit design, and Commodore computers. He is currently completing a Master's degree in mathematics. You may contact him at Transonic Laboratories, 249 Norton Street, Mankato, MN 56001.

TEXT COMPRESSION AND ENCRYPTION

By Walter Luke Jr.

By compressing data that normally occupies three bytes into two, a memory savings of 30% or more can be achieved. The same technique saves space on cassette or disk and results in a code that is difficult to break.

I was writing a Madlib game a few years ago and received an OUT OF MEMORY ERROR two-thirds of the way through. Naturally I was distressed because it is virtually impossible to squeeze out 5K of extra space from a 16K program. I listed the program and couldn't believe I had used up anywhere near 16K of memory; I had other programs that occupied a larger volume of space and still had 5 or 6K left to spare.

The problem was real and my gross underestimate of the amount of space I needed occurred because of two related facts:

1. A program composed principally of BASIC statements does not occupy as much space as the size of the listing implies because keyboards are tokenized by the Editor and use up only one byte of memory regardless of their external appearance.
2. The text attending instructions or screen displays eats up space — and quickly! Each short story in the Madlib game filled the screen approximately one and one-half times and therefore consumed about 1.5K of memory since each character and space uses one byte of RAM.

To make matters worse, BASIC imposes an 8-byte overhead for each line of text retained:

4 bytes for the line number and line link.
1 byte for the end of line flag.
1 byte for the 'Data' or 'Print' keyword.
2 bytes for the quotes.

Any other arrays or intermediate variables used to manipulate the text are an additional overhead.

I needed a way to get around the one byte per character memory penalty. An 8-bit byte can contain 256 different characters, but 95% of the ones I needed were among the 26 letters of the alphabet and 10 decimal digits. The rest are punctuation.

The software presented exploits this observation and packs three characters of data into two bytes — a 33% savings. This is great for my needs and helped a lot before I had a disk and was unable to swap data rapidly. Since then I've used the method to compress disk data and to encrypt information too sensitive to leave in plain text format in a timeshare system.

The process is as follows:

1. Define the characters you require - alphanumeric, graphic, or a mixture. This forms your abbreviated charcter set. You can use as many as forty different characters. Assign these to the variable AL$. (Refer to listing 1 - Text Compression and Encryption.)
2. Find an area of memory that won't be bothered for a while and assign the address of the start of this region to BA.
3. Assign each line of text to be compressed to M$ and sick the Compression program on it. After you've processed the entire text assign the string ETX to M$ to flag the program that all text has been received and to store an end mark (three zeros) into memory.
4. After the compressed text is stored into RAM, use whatever utility you have available to perform a block save of the populated RAM contents. The variable ET points to the end of text (and endmark) plus one and can be used along with your original assignment of BA to define precisely the range of memory saved. Listing 1 contains an example that might be helpful.

The characters I will be using are the space, comma, period, decimal digits and the alphabet. I've arbitrarily selected location 8192 ($2000 Hex) as a convenient holding area. BA is initialized to this. I've embedded the text I want to compress into DATA statements at the end of the program. I also could have entered the text from the keyboard in response to INPUT prompts, or read in an external data file. Each line of text is read into M$ and stored in compressed form into RAM. When the blurb ETX is encounted the program recognizes that the end of text has already been processed and stores three consecutive zeros into memory immediately after the compressed text. The decoding software will need this when the text is regenerated. The text in the DATA statements, by the way, is an excerpt from the Madlib program I mentioned earlier.

I now save the region of memory from 8192 and 8773 (ET) onto disk. You do your equivalent. A CBM monitor sequence might look like this:

```
.S "COMPTEXT", 01. 2000, 2245
```

Let me digress and demonstrate what this extra trouble has accomplished. The text occupies 812 bytes in the DATA statements. (The DATA statements themselves occupy something like 216 bytes, but let's ignore this.) The compressed text is contained in 581 bytes and our savings comes to 28.45%. This figure will asymptotically approach 33.33% as the amount of text increases and average line length increases. (The 'return' character has to be injected less often.)

Now that the compressed text is safely packed away, we will now have to be able to regenerate it for future use. (Refer to listing 2 - Text Expansion and Decryption.) This is accomplished by:

1. Setting AL$ equal to the same character string used in the Compression program.
2. Setting BA to the starting address of the region of memory you will load the compressed text into when executing the program.

To regenerate the Madlib text just stored we take care of AL$ and BA as described. For a CBM machine the monitor command might look like:

```
.L "COMPTEXT", 01
```

Again, do your equivalent.

Now, run the program and view the output shown in figure 1.

Note that the parentheses and apostrophes are missing. This is because they don't exist in our subset of the ASCII code. When the encoding software encounters these characters it POKES a space into RAM instead.

Many texts could be regenerated by

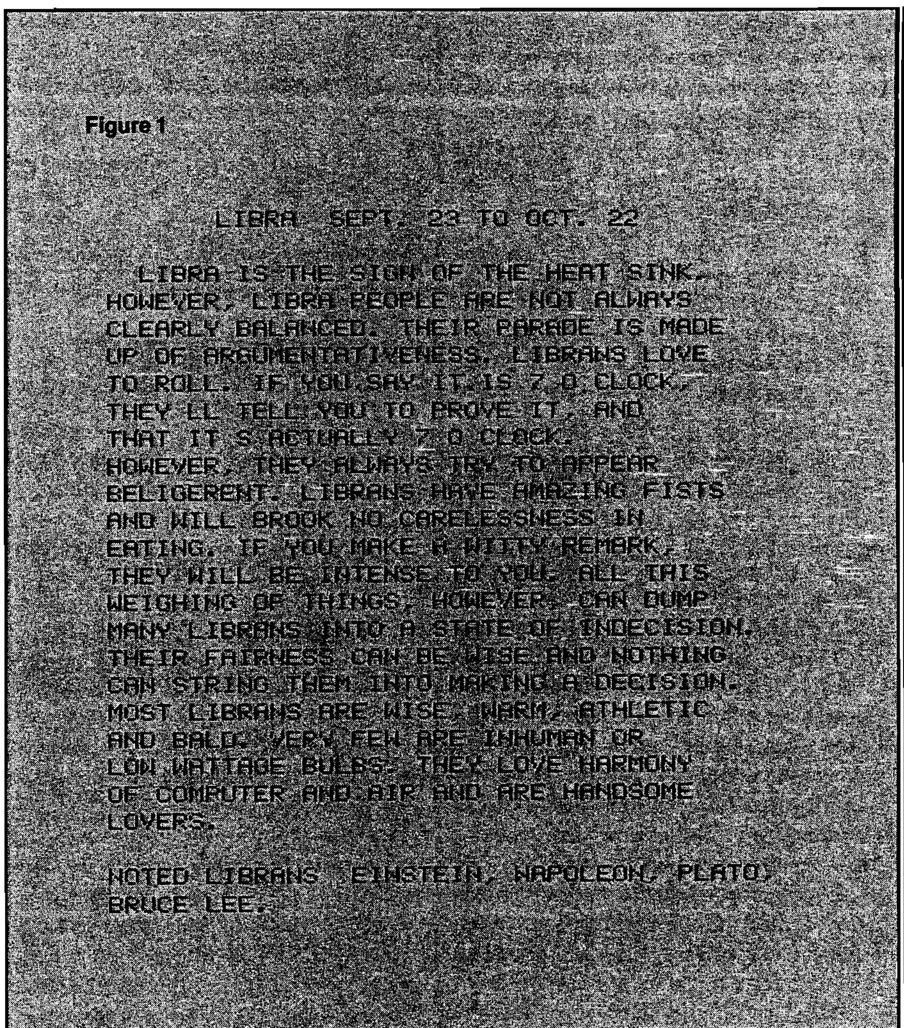


Figure 1

using a block of memory as a buffer for text and using the Text Expansion software as a subroutine to repetitively recreate successive texts placed into the buffer.

Another application of this software is not obvious. Let me pose two questions. If you viewed a dump of memory containing the compressed text, would you have any idea what it represented? It doesn't look anything like ASCII coded characters or compressed BASIC text. Moreover, if you had access to the software and looked at the variable S that is used to hold and form the two-byte sum in either program, you would gain no information. What if the string assigned to AL$ were called a key instead of a character set? It could be viewed in this light since the contents of memory or program variable S have no practical meaning if they can't be linked to a unique character string.

Therefore, the compressed text formed by listing 1 is meaningless garble unless it is processed by the software of listing 2 *and* unless the exact composition of AL$ is known for both programs.

Change line 1100 in listing 1 to read, AL$ = CR$ + " ,.EDCBA98765432 10ZYXWVUTSFGHIJKLMNOPQR". Run listing 1 and either save the encrypted text or immediately load listing 2 without turning the computer off. In either case, run listing 2 with the encrypted data in memory. The result is shown in figure 2.

A rather meaningless assortment of garbage, no? Only the spacing and punctuation has been preserved but we could have destroyed that by permuting these characters in AL$ also. Now change AL$ to match line 1100 in listing 1 and rerun the program. Magically, everything has seemed to sort itself out.

This process can be used to impose privacy on any desired text or software listing. It could also be used to secure data transmissions over communications links. In either case the information is secure against anybody not possessing *both* the software *and* the correct key. Rather surprising that one set of software can perform two apparently unrelated functions — Text Compression and Data Encryption! Now go ahead and be creative...

I've cleverly sidestepped any mention of the mathematical justification behind the software. It would lengthen this article two orders of magnitude to describe it lucidly. Let me partially

Figure 2

```
          30D9E  WA7V. ML V6 6CV. MM

   30D9E 0W VZA W0Y5 6X VZA ZAEV W052.
 Z6SATA9, 30D9E 7A673A E9A 56V E3SEQW
 C3AE93Q DE3E5CAB. VZA09 7E9EBA 0W 4EBA
 U7 6X E9YU4A5VEV0TA5AWW. 30D9E5W 36TA
 V6 9633. 0X Q6U WEQ 0V 0W H 6 C36C2,
 VZAQ 33 VA33 Q6U V6 796TA 0V, E5B
 VZEY 0V W ECVUE33Q H 6 C36C2.
 Z6SATA9, VZAQ E3SEQW V9Q V6 E77AE9
 DA30YA9A5V. 30D9E5W ZETA E4EP05Y X0WYW
 E5B S033 D9662 56 CE9A3AWW5AWW 05
 AEV05Y. 0X Q6U 4E2A E S0VVQ 9A4E92,
 VZAQ S033 DA 05VA5WA V6 Q6U. E33 VZ0W
 SA0YZ05Y 6X VZ05YW, Z6SATA9, CE5 BU47
 4E5Q 30D9E5W 05V6 E WVEVA 6X 05BAC0W065.
 VZA09 XE095AWW CE5 DA S0WA E5B 56VZ05Y
 CE5 WY905Y VZA4 05V6 4E205Y E BAC0W065.
 46WV 30D9E5W E9A S0WA, SE94, EVZ3AV0C
 E5B DE3B. TA9Q XAS E9A 05ZU4E5 69
 36S SEVVEYA DU3DW. VZAQ 36TA ZE9465Q
 6X C647UVA9 E5B E09 E5B E9A ZE5BW64A
 36TA9W.

 56VAB 30D9E5W  A05WVA05, 5E763A65, 73EV6,
 D9UCA 3AA.
```

Listing 1

```
100 REM
200 REM TEXT COMPRESSION & ENCRYPTION
300 REM WALTER LUKE JR.
400 REM 8/5/82
500 REM
600 REM THIS PROGRAM COMPRESSES TEXT PASSED TO IT IN 'M$' INTO MEMORY
700 REM LOCATIONS POINTED TO BY 'BA'
800 REM
900 PRINT "J"
1000 CR$=CHR$(13)
1100 AL$=CR$+" ,.ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789": REM CHARACTER SET
1200 LA=LEN(AL$): REM MAKE SURE 'AL$' CONTAINS NO MORE THAN 40 ELEMENTS
1300 BA=8192
1400 READ M$: IF M$="ETX" GOTO 1800: END OF TEXT MARKER FOUND
1500 LM=LEN(M$)
1600 GOSUB 2200
1700 GOTO 1400
1800 FOR I=0 TO 2:POKE BA+I,0:NEXT I:REM END OF COMPRESSED TEXT MARKER
1900 ET=BA+3: REM POINTER FOR END OF COMPRESSED TEXT IN MEMORY
2000 END
2100 REM
2200 REM BREAK LINE OF TEXT 'M$' INTO 3 LETTER GROUPS IN 'T$'
2300 REM
2400 FOR I=1 TO LM STEP 3
2500 IF LM-I > 1 THEN T$=MID$(M$,I,3): REM USUAL CASE
2600 IF LM-I = 1 THEN T$=RIGHT$(M$,2)+CR$: REM 2 LETTERS LEFT
2700 IF LM=I THEN T$=RIGHT$(M$,1)+" "+CR$: REM 1 LETTER LEFT
2800 REM
2900 REM 'T$' HAS A 3 LETTER GROUP
3000 REM
3100 GOSUB 3600: REM GENERATE INDEX VECTOR 'V%' FROM 'T$'
3200 IF LM-I =2 THEN T$="  "+CR$:GOSUB 3600
3300 NEXT I
```

(continued)

atone for this by talking about the programs a little. They were written on a CBM machine, but to my knowledge, the only machine-dependent function I sneaked in was 'Print "CLR" ' in both programs to clear the screen. Therefore, the software should be easily transportable to other machines. The programs are optimized for nothing in particular and could be speeded up and shortened spectacularly by using variables instead of constants, eliminating the Gosubs, De-'REM'arking, and using multiple statements on a line.

In listing 1 line 4400 V%(J) is given a value of 1, which is equivalent to a space if a character to be compressed is not contained in AL$. You might want this default to be another value. The test for S greater than 65535 in line 5300 is to insure that S will fit into two bytes. With the restriction that LEN(AL$) never exceed 40, this will never happen. Later on, if you blunder over this limit by changing AL$, this test may save you a lot of debugging.

Line 3200 of listing 2 checks for V%(J) = 1, which is the index for the Return character. This is not contradicting the previous paragraph, because the origins are different. If this test is passed, a Line Feed is supplied. If you are outputting to a device that automatically inserts a Line Feed when a Return is detected, you will get double spacing on the output. Delete this test if double spacing occurs.

Variables Used

- M$ - Holds line of text to be compressed
- T$ - Holds three characters from m$ at a time
- C$ - Holds one character from T$. Used for character to numeric equivalent conversion.
- AL$ - The key or abbreviated character set
- CR$ - Carriage Return. ASCII 13.
- BA - Memory pointer
- LA - Length of key or abbreviated character set
- LM - Length of line of text being compressed
- S - Running sum
- V% - Holds indices into AL$
- ET - Pointer to end of text plus one in memory

You may contact the author at R.D. 2, Maxian Rd., Box 1366, Binghamton, NY 13903.

Listing 1 *(continued)*

```
3400 RETURN
3500 REM
3600 REM CONVERT T$ TO A 3 COMPONENT VECTOR OF INDICES INTO AL$
3700 REM
3800 FOR J=1 TO 3
3900 C$=MID$(T$,J,1)
4000 PRINT C$;
4100 FOR K=1 TO LA
4200 IF MID$(AL$,K,1)=C$ THEN V%(J)=K-1:GOTO 4500
4300 NEXT K
4400 V%(J)=1: REM CHARACTER APPARENTLY DOESN'T EXIST IN OUR ALPHABET.
4500 NEXT J
4600 REM
4700 REM ENCODE THE 3 INDICES IN V% TO A TWO BYTE SUM
4800 REM
4900 S=0
5000 FOR J=1 TO 3
5100 S=S+V%(J)*LA↑(J-1)
5200 NEXT J
5300 IF S>65535 THEN STOP: REM POTENTIAL PROBLEM HERE
5400 REM
5500 REM SAVE 2 BYTE VALUES IN MEMORY LOCATIONS POINTED TO BY 'BA'.
5600 REM
5700 POKE BA, S/256: REM HI BYTE
5800 POKE BA+1, (S/256-INT(S/256))*256: REM LOW BYTE
5900 BA=BA+2
6000 RETURN
6100 REM
6200 REM ONE WAY TO CONVEY TEXT TO THE COMPRESSION PROGRAM
6300 REM IS VIA DATA STATEMENTS AS SHOWN BELOW...
6400 REM
6500 DATA "      LIBRA (SEPT. 23 TO OCT. 22)
6600 DATA " ": REM SINGLE SPACE FORCES A BLANK LINE
6700 DATA "  LIBRA IS THE SIGN OF THE HEAT SINK."
6800 DATA "HOWEVER, LIBRA PEOPLE ARE NOT ALWAYS"
6900 DATA "CLEARLY BALANCED. THEIR PARADE IS MADE"
7000 DATA "UP OF ARGUMENTATIVENESS. LIBRANS LOVE"
7100 DATA "TO ROLL. IF YOU SAY IT IS 7 O'CLOCK,"
7200 DATA "THEY'LL TELL YOU TO PROVE IT, AND"
7300 DATA "THAT IT'S ACTUALLY 7 O'CLOCK."
7400 DATA "HOWEVER, THEY ALWAYS TRY TO APPEAR"
7500 DATA "BELIGERENT. LIBRANS HAVE AMAZING FISTS"
7600 DATA "AND WILL BROOK NO CARELESSNESS IN"
7700 DATA "EATING. IF YOU MAKE A WITTY REMARK,"
7800 DATA "THEY WILL BE INTENSE TO YOU. ALL THIS"
7900 DATA "WEIGHING OF THINGS, HOWEVER, CAN DUMP"
8000 DATA "MANY LIBRANS INTO A STATE OF INDECISION."
8100 DATA "THEIR FAIRNESS CAN BE WISE AND NOTHING"
8200 DATA "CAN STRING THEM INTO MAKING A DECISION."
8300 DATA "MOST LIBRANS ARE WISE, WARM, ATHLETIC"
8400 DATA "AND BALD. VERY FEW ARE INHUMAN OR"
8500 DATA "LOW WATTAGE BULBS. THEY LOVE HARMONY"
8600 DATA "OF COMPUTER AND AIR AND ARE HANDSOME"
8700 DATA "LOVERS."
8800 DATA " "
8900 DATA "NOTED LIBRANS: EINSTEIN, NAPOLEON, PLATO,"
9000 DATA "BRUCE LEE."
9100 DATA "ETX"
9200 END
READY.
```

Listing 2

```
100 REM
200 REM TEXT EXPANSION & DECRYPTION
300 REM WALTER LUKE JR.
400 REM 8/5/82
500 REM
600 REM THIS PROGRAM SEGMENT WILL RECOVER AND PRINT COMPRESSED TEXT
700 REM USE THIS PROGRAM STAND-ALONE OR AS A SUBROUTINE IN YOUR LARGER PROGRAM
800 REM
900 PRINT "⊐"
1000 AL$=CHR$(13)+" ,.ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"
1100 LA=LEN(AL$)
1200 BA=8192
1300 REM
1400 REM RETRIEVE 2 BYTE INFO POINTED TO BY 'BA' AND REGENERATE THE 2 BYTE SUM
1500 REM
1600 FOR J=0 TO 2: REM TEST FOR END-OF-TEXT MARKER
1700 IF PEEK(BA+J)<>0 GOTO 2000: REM 3 ZEROS IN ROW SIGNIFIES THE END OF TEXT
1800 NEXT J
1900 END: REM GETTING HERE MEANS END OF COMPRESSED TEXT ENCOUNTERED
2000 S=256*PEEK(BA)+PEEK(BA+1)
2100 FOR J=1 TO 3
2200 S=S/LA
2300 V%(J)=(S-INT(S))*LA+1.0125
2400 S=INT(S)
2500 NEXT J
2600 BA=BA+2
2700 REM
2800 REM RECREATE ORIGINAL TEXT
2900 REM
3000 FOR J=1 TO 3
3100 PRINT MID$(AL$,V%(J),1);
3200 IF V%(J)=1 THEN PRINT CHR$(10);: REM 'LINE FEED' AFTER 'RETURN'
3300 NEXT J
3400 GOTO 1600
```

MICRO™

Using VIC and C 64 ROM Routines from BASIC

by Terry M. Peterson

Although we do most of our programming in BASIC, it is occasionally more efficient to use the computer's native language — machine-language. There are advantages in speed and memory usage, and most important, there are operations you can't even do in BASIC! However, you need a machine-language monitor and an assembler.

Fortunately, many common functions are performed in a set of routines contained in the KERNAL ROM. These routines are documented in both the *Commodore 64* and *VIC-20 Programmer's Reference Guides.* However, most of these require that you read from or write to the processor registers .A, .X, .Y, and .S. Time to get a machine-language monitor and

(Continued on page 98)

Table 1:

Bit number	Decimal value	Flag name	Flag meaning
B7	128	N	Negative result
B6	64	V	Overflow result
B5	16	—	Unused
B4	32	B	BRK encountered
B3	8	D	Decimal mode
B2	4	I	IRQ disable
B1	2	Z	Zero result
B0	1	C	Carry

Listing 1

```
100 REM SET CURSOR TO 5-TH ROW, 20-TH COL.

110 POKE 781,4:          REM Set .X to 5th row
120 POKE 782,19:         REM Set .Y to 20th col.
130 POKE 783,0:          REM Set .S for CARRY CLEAR
140 SYS 65520:           REM CALL 'PLOT'
```

assembler, right? Hold on! You may be able to put it off for now.

In the VIC and C-64 the 'SYS' BASIC statement allows you to call machine-language subroutines just as in earlier Commodore computers. However, SYS in the VIC and C-64 has been enhanced to allow you to specify the processor register contents when the subroutine is called. Also, it is possible to determine the register contents at the completion of the subroutine. BASIC does this by using four memory locations as pseudo-registers to pass the actual register contents back and forth. These locations are as follows:

Decimal Address	Register
780	.A Accumulator
781	.X X register
782	.Y Y register
783	.S Processor status

When the SYS statement is executed the contents of the four addresses listed above are loaded into the corresponding processor registers just before effectively performing a 'jump to subroutine' (JSR) to the address specified in the SYS statement. When the subroutine is finished the processor registers are saved in the same four locations before returning to BASIC. We may set the contents of the 6502 registers at the beginning of a SYSed subroutine by POKEing to the corresponding memory locations immediately prior to the SYS. Also, we may recover the values in the registers at the end of the subroutine by PEEKing those addresses.

To see exactly how this works, let's take the KERNAL's 'PLOT' subroutine (address: 65520) as an example. PLOT allows us to set or read the location of the cursor on the screen. We designate which function we want by setting the 'carry' flag of the processor status register appropriately: Carry 'set' means 'read current location into .X and .Y'; and carry 'clear' means 'move cursor to location specified by contents of .X and .Y'. (Yes, PLOT's use of 'set/clear' seems backwards to me too, but I didn't write the routine! Also 'backwards' is the use of 'x' and 'y': .X is used for the *row* and .Y for the *column*.) Now, in addition to setting or reading .X and .Y (by POKEing or PEEKing 781 and 782), we need to

Listing 2

```
200 REM READ CURRENT CURSOR POSITION

210 POKE 783,1:                           REM Set .S to CARRY SET
220 SYS 65520:                            REM CALL 'PLOT'
230 X=PEEK(781)+1: Y=PEEK(782)+1:         REM Get final .X & .Y vals.
240 REM X,Y CONTAIN ROW & COL. OF CRSR [(1,1) = HOME]
```

(*Editor's Note*: Because of BASIC's line-wrap feature, the values of X and Y may be considerably different than expected.)

Listing 3

```
THIS PROGRAM WILL SAVE TO
TAPE OR DISK A PART OF RAM.
IF THE SAVE IS TO TAPE, IT
WILL BE IN THE FORM OF AN
'ABSOLUTE' FILE THAT WILL
(RE)LOAD ONLY WHENCE IT
WAS SAVED.

1000 INPUT "DEVICE NUMBER #8[LEFT 3]";DV
1010 INPUT "FILE TO SAVE";F$: IF F$ = "" THEN 1010
1020 POKE187,PEEK(71): POKE188,PEEK(72): REM SNEAK LOC. OF F$
1030 FA = PEEK(187)+256*PEEK(188): REM CALC. POINTER TO F$
1040 POKE 183,PEEK(FA): REM SET FILENAME LENGTH
1060 POKE 187,PEEK(FA+1): POKE188,PEEK(FA+2): REM SET FN. POINTER
1070 :
1080 INPUT "START ADDRESS (HEX)";SA$
1090 N$ = SA$: GOSUB 2000: SL = BL: SH = BH
1095 :
1100 INPUT "END ADDRESS (HEX)";EA$
1110 N$ = EA$: GOSUB 2000: EL = BL: EH = BH
1120 POKE251,SL: POKE252,SH: REM SET STRT ADD. PTR
1130 POKE186,DV: POKE185,1: REM SET DEV. & S.A.
1140 POKE780,251: POKE781,EL: POKE782,EH: REM SET .A, .X, & .Y
1150 SYS65496: REM GO DO SAVE ($FFD8)
1160 END
1999 :
2000 REM CONVERT HEX TO 2 DEC. BYTES
2010 :
2020 N = 0
2030 FOR I=1 TO LEN(N$)
2040 :X = ASC(MID$(N$,I))-48
2050 :N = 16*N + X + 7*(X>9)
2060 NEXT
2100 BH = INT(N/256): BL = N-256*BH
2110 RETURN
```

determine which bit in the status register .S is the carry flag so we know what to POKE into location 783 before executing a 'SYS 65520'. Table 1 shows the processor status register bits.

In each case the flag is 'true' or 'set' if the corresponding bit is set, i.e., not zero. If we want to set the 'zero' flag we would POKE783,2 (B1 = 1); to set the 'carry', POKE783,1 (B0 = 1); to set both, POKE783,1 + 2. Now we're ready to use the plot routine (listing 1).

Note that since PLOT starts counting from zero for both rows and columns we set .X and .Y to one less than you might expect. Listing 2 demonstrates how to read the current cursor position.

As a less trivial example let's look at a BASIC program that performs the same function as the APPLE's BSAVE statement (or the 'S' monitor command, for the PET folks). This program uses the KERNAL 'SAVE' routine (65496) to copy any part of the computer's memory to tape or disk. (Actually, the tape save is restricted to memory addresses less than 32768; but, that's not a great hinderance in practice.) You might want to use this program to save a custom character set or a high-resolution screen for quick recall.

Line 1020 discovers where BASIC is keeping the value of the string F$. This is done by PEEKing at the zero page locations where BASIC's current-variable pointer is maintained. Note that the addresses of the PEEKs in line 1020 *must* be literals (i.e., ASCII digits), or the current-variable pointer will not be pointing to F$ anymore! Lines 1040-1060 then set the operating system filename pointers to use (the value of) F$ as the current filename. (We could have used the KERNAL routine 'SETNAM' for this step, but POKEing is more direct.) Line 1120 saves the start address in the free zero page area. Line 1130 sets the device primary and secondary addresses for the SAVE. (Here again, a KERNAL routine, 'SETLFS', could be used.) Finally, on 1140 the pseudo-registers are set; and SAVE is called on 1150.

MICRO

Terry Peterson is engaged in catalyst research at Chevron Research Company. He may be contacted at 8628 Edgehill Ct., El Cerrito, CA 94530.

Swap RAM or EPROM for Your ROM

by Ralph Tenny

8K byte EPROMs are expensive and they lack pin compatibility with most masked ROMs resident in personal computers. This article shows how to replace an existing 8K byte ROM with two relatively low cost EPROMs. Instructions are given for building such an adapter to replace the Extended BASIC ROM in the TRS-80C Color Computer.

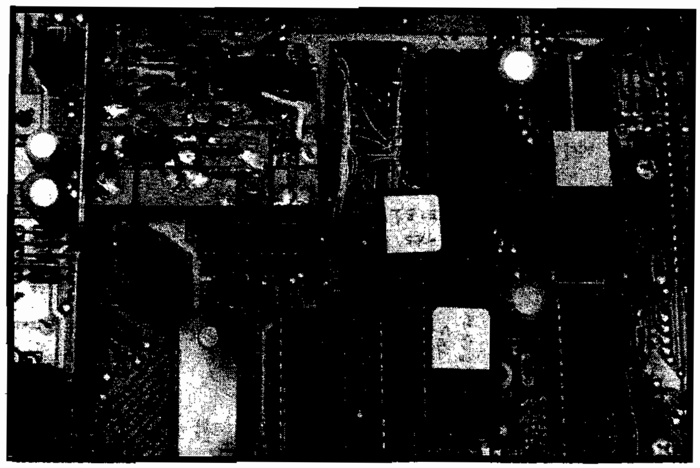


Photo 1. The maze in the extended BASIC socket is the plug part of the dual EPROM adapter; the two EPROMs zig-zag to the right and down beneath the BASIC socket.

Almost all personal computers have large blocks of memory set aside for system ROMS — operating system, BASIC, etc. These blocks of memory make the computer smart enough to perform many functions without you having to write any programs. When the computer revolution began, computer hobbyists had to write *every byte* of code that ran their computer — or else pay dearly for software support!

The other side of the coin is that our "appliance" computers — PET, Apple, VIC-20, TRS-80 Color Computer, Atari, etc., all boot up talking BASIC, and it is difficult to convince them to do otherwise. What we need is new auto start software if we want to dedicate the machine to some purpose other than a general-purpose home computer or a games machine. Although there are other ways to accomplish this, the most straightforward way is to substitute modified (or completely new) programs for the ROMs that now start the machines in BASIC.

Obviously, this is a detailed and difficult task, but it *can* be done. Part of the problem we have to solve is that many of these computers use masked ROMs, which hold 8K bytes of program. At present, the most commonly available EPROM is the 2716, which is a 2K byte EPROM.

Photo 2. Top side of the adapter board, showing the general shape of the perfboard with two sockets mounted.

However, one 2716 completely fills the socket where the current system ROM resides, and it only holds one-fourth as much program. The next choice is either the 2732 or 2532, both of which are 4K byte EPROMs; it takes two of those to equal the 8K ROM. The *next* jump is to the 2764 or 2564, but these EPROMs are 28-pin parts (too big to fit the socket) *and* they cost well over $20 each. This article describes one way around this problem — an adapter that fits two 4K EPROMs into the Color Computer to take the place of the Extended BASIC ROM. Although the details are for the Color Computer, the basic principle can be applied to any of the appliance computers if you understand their software and architecture well enough.

Let's set aside the notion of completely custom software a moment and examine our chances of partially modifying the Color Computer's software. If you study the memory map of Radio Shack's Color Computer (figure 1), you can see only two places where ROMs can be installed if you want to add your own software in ROM and preserve the I/O routines and machine initialization furnished by the BASIC ROM.

The most obvious place for your personal software is in the Extended BASIC socket addressed at $8000-$9FFF. The only other possible choice is the expansion port where the cartridge ROM fits; this port addresses at $C000-$FF00. If you wish to use Extended BASIC, only the expansion port is available. If you are using a commercial cartridge ROM that has been modified to defeat the auto-boot feature, you can use the Extended BASIC socket unless the cartridge ROM requires Extended BASIC. Either you must follow these constraints, or you must make substantial modifications to the software or to the hardware.

The fixture described here allows using two 4K byte EPROMs in the Extended BASIC socket, which normally holds an 8K byte ROM. If you want custom software totalling no more than 4K bytes, install your code in a 2532 EPROM and plug it in the Extended BASIC socket. If you have a larger program (perhaps you want to un-bug Radio Shack's BASIC), build the adapter described below and put your code into two 2532s. (It is possible to use 2732-Intel pinout-parts in the fixture, but the circuit schematic would have to be changed. The 2732 parts are not pin compatible with the Extended BASIC socket and will NOT work there! This limitation is imposed by the internal design of the 2732, which was meant to be used with the 8085 microprocessor.)

Photo 1 shows the adapter in place, but it blends into the background somewhat. The photo shows the upper right-hand area inside the RF shield of the Color Computer; the 40-pin IC in the upper right corner of the photo is the MC6883 Synchronous Address Multiplexer (SAM) chip. Moving left, you can see the BASIC ROM, and then a maze of wires feeding two 24-pin chips, one of which is turned 90 degrees from the other. This L-shaped part is the adapter I built to hold two EPROMs. The small board to the left is the CRT monitor circuit board (MICRO 54:19).

Photo 2 shows the top side of the fixture with only two sockets and a 24-pin component platform mounted to a piece of perfboard. Note that the perfboard fits closely between the pins of the component platform (JimPack Header Plug or equivalent), and the component platform is

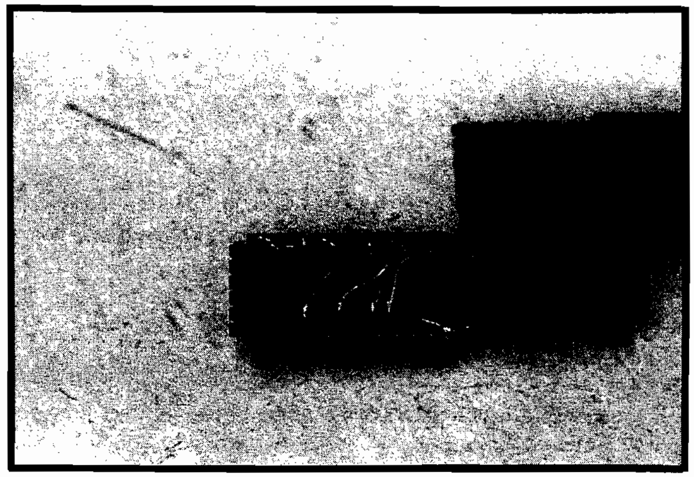


Photo 3. Here the decoder chip has been mounted and connected; power wires to the EPROMs have also been installed.

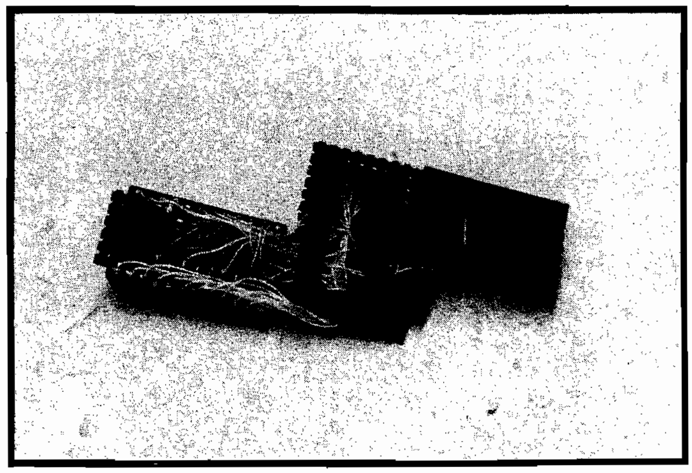


Photo 4. Top view of the completed module showing how wires are routed. Note that the center bridge of EPROM1's socket has been removed to ease the wire routing.

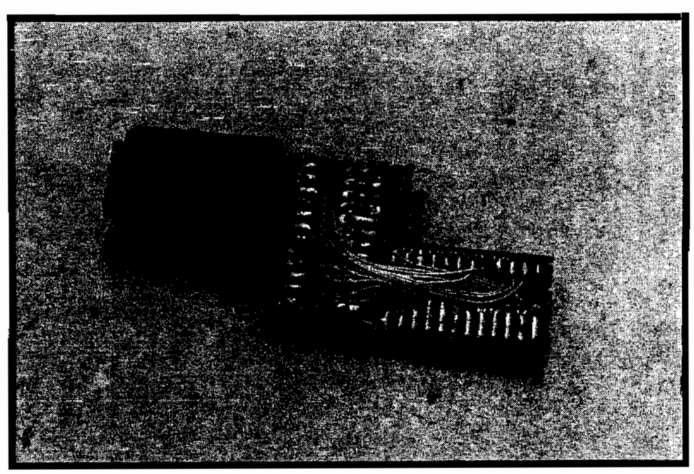


Photo 5. Bottom view of finished module. See text for commentary on wiring.

also glued to the perfboard for extra support. Photo 3 shows power wires (Vcc and Ground) and the decoder chip installed. Note that the decoder chip has been inverted and glued to the perfboard, then wired into the circuit. Photos 4 or 5 show all the wires installed; note that the center bridge of the rotated socket has been removed to ease the wiring.

Hints for building the module: Begin by cutting a piece of perfboard slightly larger than necessary to hold the two sockets and one plug. Cut and try until it will lay flat on top of the Extended BASIC socket and fit in between the surrounding components as shown in photo 1. Install the plug in the open socket, then slide the perfboard through the pins until the best fit is found. I used cyanacrolate glue ("super glue") to attach the plug; even so, always use an IC puller on the plug itself to remove the fixture; the perfboard will flex loose or break otherwise. Install the E-Z Circuit strips, trim them away from the edge of the perfboard, and bend the EPROM socket pins flat against the E-Z Circuit before soldering. Use low pro-

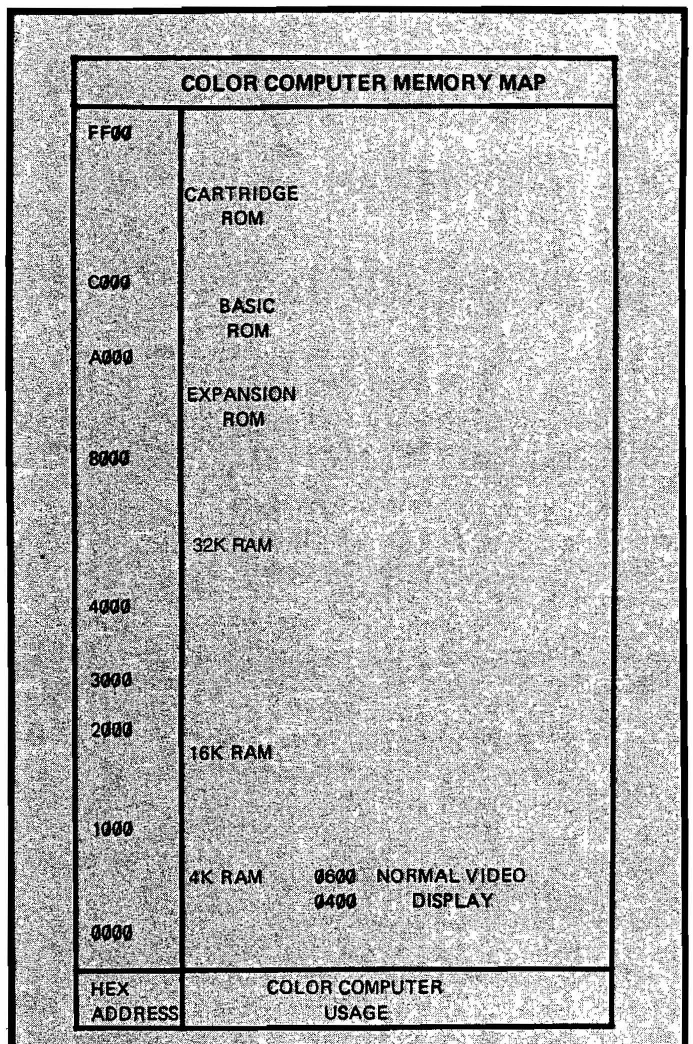


Figure 1: The memory map of the Color Computer reveals that only the Extended BASIC socket ($8000-$9FFF) or the Cartridge port ($C000-$FF00) can be used for custom software if the machine's BASIC ROM is used.

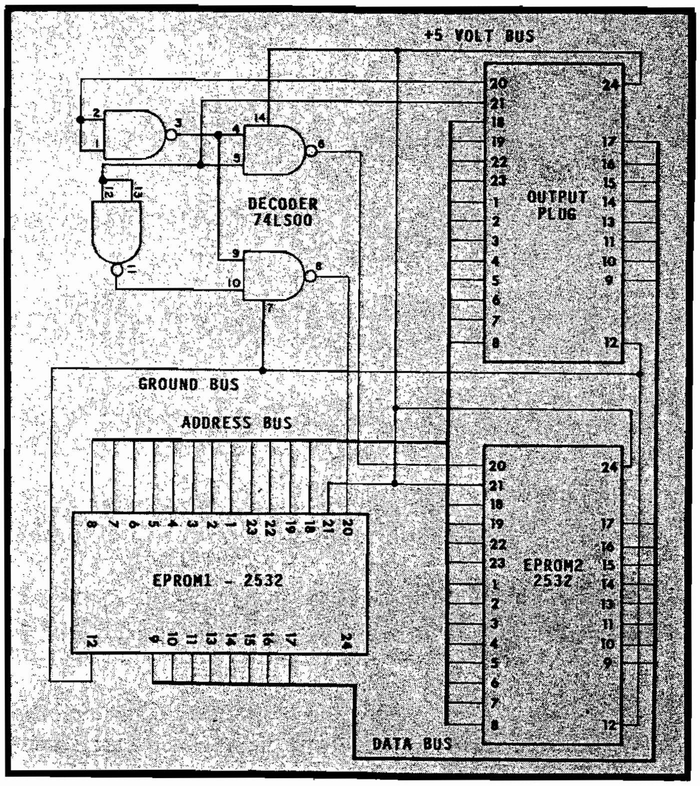


Figure 2: This schematic diagram shows how two 2532 EPROMs can be connected into the Color Computer memory map in place of the single 8K × 8 ROM, which holds Extended BASIC.

file sockets, and work carefully to minimize vertical height; it will all just barely fit under the RF shield! Refer to figure 2 for all the connections; take your time and be sure to avoid solder bridges. After the wiring is complete, check for proper continuity and shorts, then install your programmed EPROMs and run the computer. *(Note: Only the unique connections are detailed fully in figure 2; the rest are grouped into bundles and correspond pin-for-pin at each socket.)*

The BASIC ROM checks locations $8000-8001 for the code "45 58" (ASCII EX). If this check is successful, the computer begins executing the code starting at $8002. If your program begins with EX at $8000, the computer will run your program instead of BASIC. When the Extended BASIC ROM is in place, it checks $C000-C001 for "46 4B" (DK); if found execution begins at $C002. Thus, you can use Extended BASIC with your custom program, which plugs into the Color Computer expansion port.

If you wish to modify software on another computer using the concepts outlined here, you must be able to find the memory map for your computer, determine which EPROM has the proper timing for your computer, and create a wiring diagram for your computer like the one shown in figure 2.

Mr. Tenny is MICRO's *Interface Clinic* columnist. You may contact him at P.O. Box 545, Richardson, TX 75080.

MICRO

Displaying PET's Keyboard Matrix

Many commercial game programs use their own routines to read keys. This allows detection of more than one key at a time. Different keyboards and ROMs work differently. This program aids in program conversion and in writing your own keyboard scanning routines.

by Werner Kolbe

The Commodore computers of the PET/CBM series do not only differ in operating systems but also in versions of keyboard hardware. As a result, adapting programs written for one version to their own system is a primary concern of many Commodore enthusiasts. The following article will help them adapt machine-language programs to different keyboard implementations.

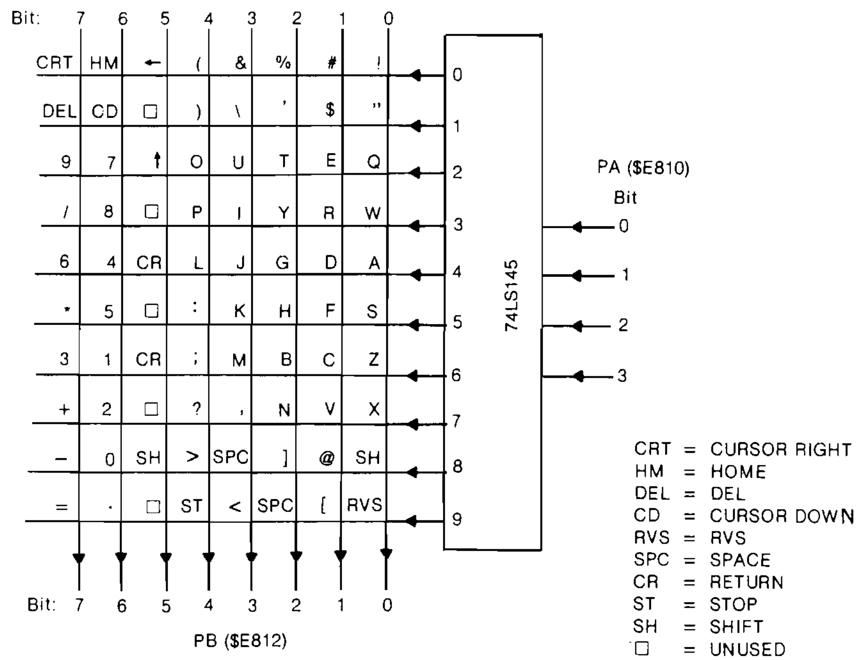
When you press a key on the keyboard you close an electrical contact that connects two wires. All the wires are organized in a rectangular matrix as shown in figure 1. The outputs of a four to ten multiplexer (in my PET it is a 74LS145) lead to ten horizontal lines. If a key is pressed down, one of these lines is connected to one of the eight vertical lines leading to the PB port of the PIA 6520.

To find out which key is pressed, the PA registered of the PIA is addressed under 59408 ($E810) and the number of the row is stored into this register. This line is then pulled from "high" or logical "1" to "low" or logical "0" level. If you want to detect whether or not the key M (on the non-business keyboard) is pressed, store a six into 59408, which will pull row six to low level. All the inputs of the PB input port have normally high level. But if in this example the "M" was pressed, you will get a "low" level on the vertical line three. Thus, if you address the PB port, you will get a binary 11110111 or a hexadecimal $F7 or decimal 247.

The operating system scans the keyboard every sixtieth second during its hardware interrupt cycle. It sequentially addresses the rows and tests the columns at the PB inputs. Therefore the BASIC programmer must not be concerned with contacts or rows and columns, he just uses his INPUT or

(Text continued on page 107)

Figure 1: **Sample Pet keyboard matrix**



Keyboard-Matrix
requires:
Any PET or CBM

Listing 1: **BASIC Program**

```
KEYBOARD  —  MATRIX

 90 FORI=890TO970:READQ:POKEI,Q:NEXT
 100 V=151:INPUT"ORIGINAL ROMS";R$
 105 IF ASC(R$)=ASC("Y")THENV=515
 110 PRINT"] BIT : 7  6  5  4  3  2  1  0 | PA
 120 PRINT"         |  |  |  |  |  |  |  |
 130 FORI=0TO9:PRINT" "I"> -                          - "I"█":NEXT
 140 PRINT"]         |  |  |  |  |  |  |  |
 150 PRINT"  BIT : 7  6  5  4  3  2  1  0 | PA ↑
 160 PRINT"██":SYS890
 170 PRINT"██  PEEK("V") ="PEEK(V)"█  ";
 180 GETA$:IFA$=""THEN160
 190 Q=ASC(A$)
 200 PRINTTAB(26)"ASC  ="Q"█  ";
 210 GOTO160
 220 DATA120,162,9,142,16,232,138,72,173,18,232,32,143,3,104,170,202,16
 221 DATA240,88,96,72,162,8,32,194,3,160,7,104,10,72,176,9,169,18
 222 DATA32,210,255,169,48,208,2,169,49,32,210,255,169,146,32,210,255,136
 223 DATA48,7,162,2,32,194,3,240,222,104,32,189,3,169,13,76,210,255
 224 DATA169,29,32,210,255,202,208,250,96
READY.
```

Listing 2:

Disassembly of machine-code portion

```
37B A2 09              LDX =09
37D 8E 10 E8           STX PORTA
380 8A                 TXA
381 48                 PHA
382 AD 12 E8           LDA PORTB
385 20 8F 03           JSR J1
388 68                 PLA
389 AA                 TAX
38A CA                 DEX
38B 10 F0              BPL 37D
38D 58                 CLI
38E 60                 RTS
38F 48           J1    PHA
390 A2 08              LDX =08
392 20 C2 03           JSR J3
395 A0 07              LDY =07
397 68                 PLA
398 0A                 ASLA
399 48                 PHA
39A B0 09              BCS L4
39C A9 12              LDA =12
39E 20 D2 FF           JSR J5
3A1 A9 30              LDA =30
3A3 D0 02              BNE L6
3A5 A9 31        L4    LDA =31
3A7 20 D2 FF     L6    JSR J5
3AA A9 92              LDA =92
3AC 20 D2 FF           JSR J5
3AF 88                 DEY
3B0 30 07              BMI L7
3B2 A2 02              LDX =02
3B4 20 C2 03           JSR J3
3B7 F0 DE              BEQ 397
3B9 68           L7    PLA
3BA 20 BD 03           JSR J9
3BD A9 0D        J9    LDA =0D
3BF 4C D2 FF           JMP J5
3C2 A9 1D        J3    LDA =1D
3C4 20 D2 FF           JSR J5
3C7 CA                 DEX
3C8 D0 FA              BNE 3C4
3CA 60                 RTS
```

GET statements and the system does the work for him. But if he tries to program more advanced games, he will discover a major drawback of the GET or, in machine language, the $FFE4 subroutine. It works only for one key pressed at the same time. If you try to control speed and direction at the same time, one or both will get priority and the other function will be disabled. That is the reason why many good programs use their own keyboard scanning routine by which this problem can be avoided. If you want to use such a program on a system with another keyboard, you will have to alter this routine.

I wrote the program "Keyboard-Matrix" to get a clear impression of the different functions and to be able to investigate the differences of the keyboard versions. The program is partly written in machine language to allow a fast response, but it is also possible to access the ports *via* PEEK and POKE from BASIC. In this case you have to disable PET's hardware interrupt before you interfere with the operating system's scanning routine. With a POKE 59411,60 the interrupt will be disabled, and with POKE 59411,61 it is restored. You must do that in a program because after the POKE 59411,60 your keyboard will be dead and you cannot enter anything else.

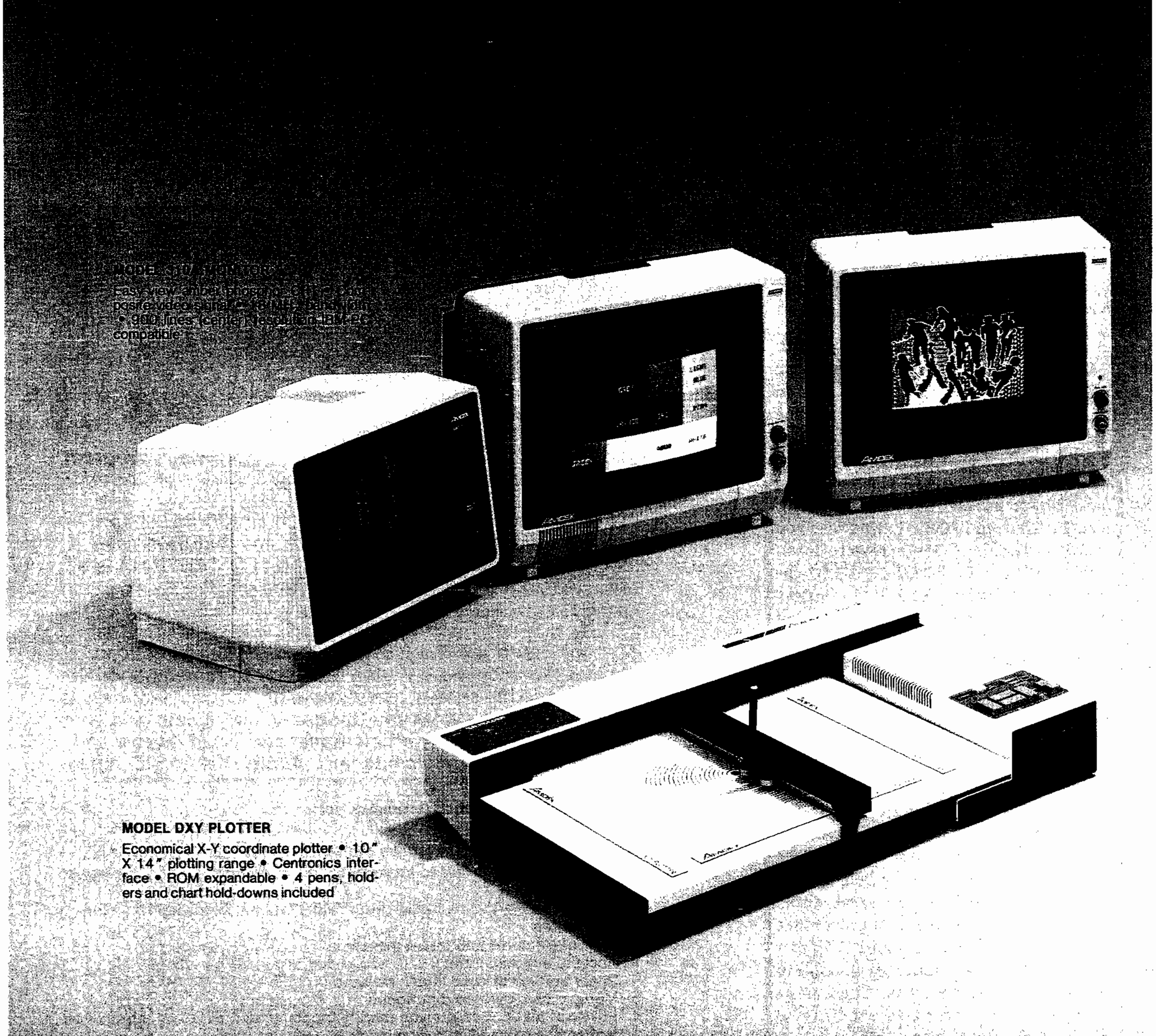
When running, the program Keyboard-Matrix will show you on the screen which row is connected to which column by the key you are pressing. It also works when several keys are pressed at the same time. Watch what happens if you press three keys, such as G, H, J in figure 1. In this case row 5 is also connected to column 3 over the three switches. There is no way to detect under this condition if the K is pressed or not.

Many programs also use the value that they PEEK under 151 (515 for the old ROMs), where the system puts a coded value of the pressed key. As the systems use different codes, the content of this location, together with the appropriate ASC value, is also displayed on the screen.

You may contact Mr. Kolbe at van der KamLaan 65, 2625 KN Delft, Netherlands.

MICRO

AMDEK...your guide to



YOUR COMPATIBILITY CHART

MORE COMPATIBILITY INTERFACES DUE SOON . . . CHECK FACTOR

COMPUTER	MONITORS							PLOTTER	AMDISK III	NOTES
	VIDEO-300	VIDEO-310	COLOR-I	COLOR-II	COLOR-IIA	COLOR-III	COLOR-IV			
IBM-PC	●	●	●	●	★	●		●	●	* Special Cabling Required
APPLE III	●			★	★	★		●		* Special Cabling or Converter Required
APPLE II	●		●	★	★	★		●		* DVM Board Required
ATARI 800	★		★							* Opt. Atari Cable Required
VIC-20			★							* Opt. VIC Cable Required
TRS-80	★									* Opt. TRS Cable Required
Osborne	★									* Opt. Interface Required
TI-99			★							* Opt. TI Cable Required
Commodore-64	★		★							* Opt. Commodore Cable Req.

WE DIDN'T MAKE IT CUTE,

Signed Binary Multiplication with the MC6809

By T. J. Wagner and G. J. Liponski

Timothy Stryker ("Signed Binary Multiplication is Unsigned," MICRO 56:76) observed that when two m-bit unsigned integers are multiplied, the least significant m-bits of the product is the correct signed product when the m-bit integers are treated as signed integers *and* their product is in the m-bit signed range. (The phrase *signed* integers in this note always means two's complement integers.) For example, to multiply two 8-bit signed integers, one could sign extend each to sixteen bits, perform an unsigned multiply with the 16-bit extensions, and take the least significant sixteen bits of this product for the signed 16-bit result.

In this note, we offer a different technique for signed multiplication, which is useful on a microprocessor that has an unsigned multiply instruction, such as the MC6809. This will also provide another comparison between the 6502 and the 6809.

The 6809 has a multiply instruction, MUL, which multiplies the unsigned 8-bit contents of accumulator A with the unsigned contents of accumulator B, putting the result in accumulator D (accumulator A concatenated with accumulator B). Because MUL is short and fast, it is more efficient to write multiple precision multiplication subroutines using MUL rather than implementing any of the standard algorithms. (Several such subroutines may be found in T.J. Wagner and G.J Lipovski's, *Fundamentals of Microcomputer Programming*, (MacMillan Publishing Co., Ltd., 1983).) It also makes sense to find ways of doing signed multiples that use MUL. We illustrate how the contents of D can be modified after MUL to carry out effectively a multiplication of the signed contents of A and B. Once you understand the technique, you can modify any unsigned multiplication routine to get the equivalent signed routine.

Figure 1

```
*       SUBROUTINE SGNMUL
*
*
* SGNMUL multiplies the signed con-
* tents of A times the signed contents
* of B, returning the correct signed pro-
* duct in D. Registers D and CC are
* changed. Only bit N in CC is set cor-
* rectly on return.
*
SGNMUL      PSHS      A,B
            MUL
            TST       1,S
            BPL       SGN1
            SUBA      ,S
SGN1        TST       ,S
            BPL       SGN2
            SUBA      1,S
SGN2        LEAS      2,S
            RTS
```

Suppose that M and N are 8-bit signed integers with two's complement representations $a_7,\ldots,a_0$ and $b_7,\ldots,b_0$, respectively. If M is in accumulator A and N is in accumulator B, then the MUL instruction multiplies

$$(M + a_7*2^8)*(N + b_7*2^8) \qquad (1)$$

putting the result in accumulator D. For example,

$$M = -a_7*2^7 + a_6*2^6 + \ldots + a_0*2^0$$

so that

$$M + a_7*2^8 = a_7*2^7 + a_6*2^6 + \ldots + a_0*2^0$$

is the *unsigned* integer in accumulator A and, similarly, $N + 6_7*2^8$ is the unsigned integer in accumulator B. Since (1) equals

$$M*N + a_7*N*2^8 + b_7*M*2^8 + a_7*b_7*2^{16} \qquad (1)$$

we see that modifying the contents of accumulator D to get M*N requires subtracting the two middle terms of (2) from D if they are non-zero. (The last term of (2), if non-zero, does not appear in D and can be ignored.) The subroutine SGNMUL, shown in figure 1, makes this adjustment in D where we note that to subtract 2^8*N or 2^8*M from D, we need only subtract N or M from the accumulator A. The instruction

```
BSR      SGNMUL
```

then is like an instruction that multiplies the signed contents of accumulator A times the signed contents of accumulator B, putting the result in D.

A comparison of the 6809 subroutine SIGMUL with the 6502 subroutine of listing 1 of Stryker indicates a substantial improvement in both length and speed. Multiple precision unsigned multiply subroutines for the 6809 can be easily modified by this technique to get efficient multiple precision signed multiply subroutines. We emphasize that this technique is most useful when used with microprocessors with an unsigned multiply instruction.

You may contact the authors at The University of Texas at Austin, College of Engineering, Austin, TX 78712.

MICRO

MICRO™

Interface Clinic

by Ralph Tenny

The circuits presented in previous columns have not had stringent power supply requirements, so batteries have been one option to power all designs presented. Our future projects will be much more dependent upon good power supply performance for proper operation than previous circuits. Therefore discussion of power supply techniques is in order.

My prime concern in the discussion to follow will be two power supply characteristics: *regulation* and *impedance.* Voltage regulation of a power supply is expressed as a percentage: (voltage change)/(output voltage) × 100%. Power supply impedance is defined as (voltage change)/(current change). We will compute examples below, but both these power supply parameters are computed after applying a load to a power supply and recording the changes in output voltage.

My main reason for discussing power supplies is that some of you will either want to save cash outlay by building your own, or learn by doing (the best way to learn!). For you tinkerers, I hope to provide guidelines to help insure successful project development.

Let's consider alternatives to the power supplies mentioned previously. One prime source for experimenter power supplies is the AC adapters now readily available. AC adapters are entirely adequate as primary voltage sources, but there are certain considerations that will dictate the performance of circuits they power.

Two kinds of adapters are available; the simplest type outputs only an AC voltage, while the second kind provides a DC voltage, with or without a filter capacitor on the output. Figure 1 shows a typical unregulated power supply that can be built using a variety of parts. The dashed line encloses the circuit diagram of an AC-output adapter, while the solid line encloses the circuit of a DC-output unit. If a filter capacitor is included in a DC-output adapter, it

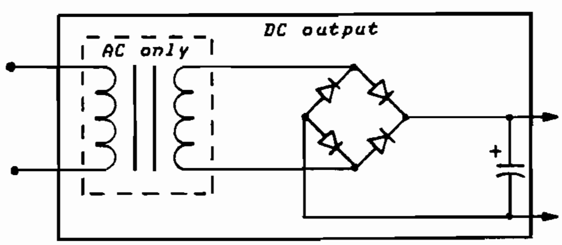


Figure 1: Typical schematics for AC adapters, showing the difference between AC and DC output types.

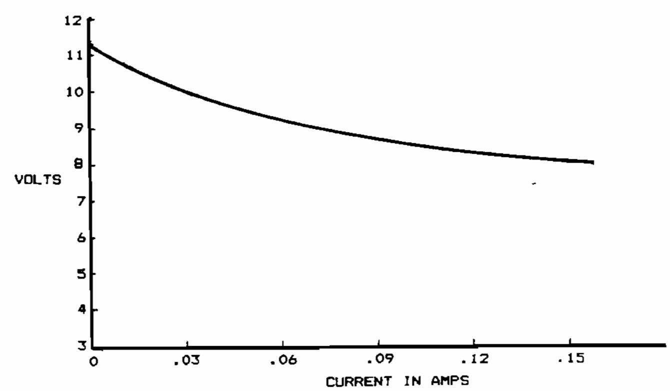


Figure 2: Voltage vs. output current plot of a typical DC-output AC adapter.

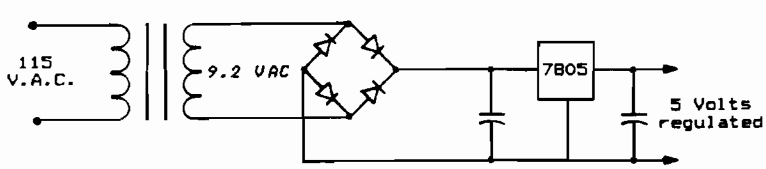
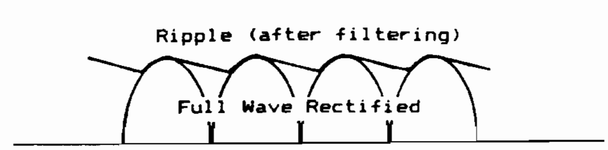


Figure 3: Regulated DC output from a DC-output AC adapter.

Figure 4: Full wave rectified DC (lower trace) can be filtered by adding a capacitor; the capacitor charges on the DC peaks and discharges between peaks.



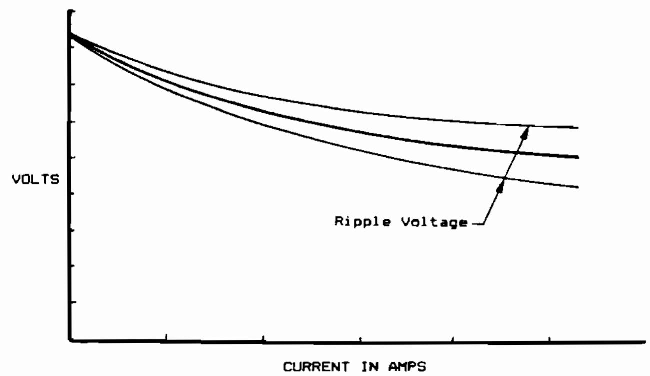
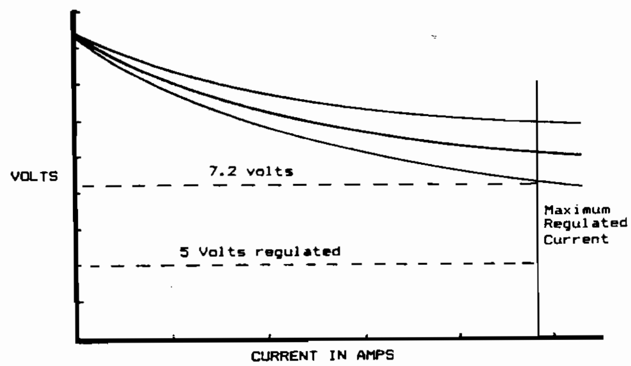


Figure 5: A repeat of Figure 1, showing the addition of ripple voltage excursions.

Figure 6: By allowing 2.2 volts "headroom" for a three-terminal regulator, it is possible to determine graphically the maximum regulated current.



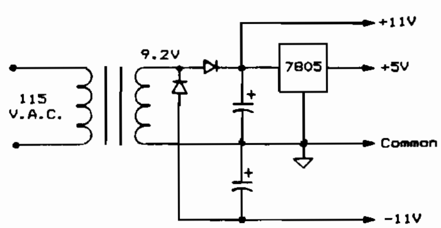


Figure 7: An AC-output adapter gives added flexibility in designing custom power supplies for special projects.

probably will be too small to give adequate filtering for our needs.

Previously I said that AC adapters are adequate as a *primary* voltage source. Let's see why I put a qualifier on that: figure 2 shows the voltage regulation curve for a typical DC-output AC adapter, which has a *rating* stamped on the case. This rating says "8V DC 160 ma."; the curve shows that, with 160 ma. (1 ma. = .001 Ampere) load, the output is about 8 volts. However, note that the voltage soars dramatically as the current load is reduced to zero. The normal voltage most of our experiments will need is 5 volts, +/−5% (between 4.75 and 5.25 volts), and this device will always exceed that limit unless it is severely overloaded.

Let's stop and figure the regulation and impedance of this "typical" power supply. From the performance curve, we can see that the no load output voltage is 11.4 volts; with the 150 ma. load the output is only 8.1 volts. To compute the regulation: (11.4 − 8.1)/11.4 = 29% (very poor). The impedance is 3.3v./.15A = 22 ohms. For our purpose, we need to put a 5-volt regulator on the output of the AC adapter as shown in figure 3. Now, when we measure the voltage change between no load and 150 ma. load, we find only 22 millivolts change. The new regulation is 0.022/5 volts, or .44% regulation; the impedance is 0.022/.15 = .15 ohms. That is much better and is about what we should expect to use for digital circuits.

Another important consideration for using any transformer and rectifier system for powering electronic circuits is ripple voltage. Figure 4 shows the voltage waveforms that result from full-wave rectification, before and after adding a filter capacitor to the circuit. The capacitor can be made very large, so that the ripple becomes small, but this increases the cost of the project. If we allow, for example, 1 volt of ripple, then figure 5 shows how the ripple

(Continued on page 115)

voltage affects the voltage output. Note that the ripple gets larger in proportion to the load current, and that this ripple (which really is an AC signal imposed on the DC voltage) is approximately centered on the voltaged that would be read by an ordinary voltmeter. The three-terminal regulator shown in figure 3 requires a minimum of 2.2 volts excess voltage in order for it to regulate properly. Since the regulator is very fast, this excess voltage should be measured at the bottom of the ripple peaks. This is illustrated in figure 6, where two straight lines corresponding to +5 volts and +7.2 volts (2.2 volts excess) have been added to the graph. At the point where the 7-volt line intersects the ripple voltage peaks, the regulator will stop working properly. This represents the maximum *regulated* current it is possible to deliver to a load with the transformer and filter capacitor shown.

In many cases, it is preferable to use AC-output AC adapters for a project; one major advantage is that these adapters are often available with higher output current; in fact, one with 2.5 Amperes output is available from Jameco. Another advantage is that with DC-output units, only the single voltage is easily available. Figure 7 shows a power supply that gives both +11 volts and −11 volts unregulated, and +5 volts regulated. This power supply circuit can be used for a small computer if the heavy-duty Jameco transformer is used. The two unregulated voltages can be used for RS-232 output drivers, while the regulated voltage can be used to power the main computer circuits. So, by making careful choices of transformers and other components, a broad range of voltages can be generated.

One final topic: power supply wiring practice. You should *always* use a heavy wire (or wide copper strip on a PC board) for the power supply common connection. Also, all high-current voltage wires should be as large as possible. If a wire or trace has to go over about 3", you should connect small filter (decoupling) capacitors between the power wire and ground at several places along the length of the wire. Finally, be sure that you solder all power supply connections very carefully, to minimize the chance of high impedance connections, which can prevent digital circuits from working properly.

Let's add another supplier to the onces previously mentioned, not because I feel we should spread our money around, but rather to have a broader ranger of parts available than Radio Shack has. I recommend you write to Jameco Electronics and get a catalog. Their address is 1355 Shoreway Rd., Belmont CA 94002. This company has a broad range of integrated circuits, along with many other parts useful in the projects we will see in future sessions.

Please forward questions and suggestions for discussion topics to Mr. Tenny at P.O. Box 545, Richardson, TX 75080.

MICRO

computer mail order

TeleVideo

TERMINALS

910	$559.00
912	$689.00
920	$739.00
925	$719.00
950	$929.00
970	CALL

COMPUTERS

800A	$1099.00
802	$2699.00
803	$1949.00
802H	$4695.00
806/20	$4999.00
816/40	$9199.00
1602	$3399.00
1603	CALL

MODEMS

HAYES

Smart	$219.00
Smart 1200 (1200 Baud)	$519.00
Chronograph	$199.00
Micromodem 100	$309.00
Micromodem II	$279.00
Micromodem II (with term)	$299.00
Smart Com II	$99.00
Smart 1200B	$469.00

NOVATION

J-Cat	$119.00
Cat	$144.00
D-Cat	$159.00
103 Smart Cat	$189.00
Apple Cat II	$279.00
103/212 Smart Cat	$439.00
212 Apple Cat II	$609.00
Apple Cat II 212 Upgrade	$309.00

ANCHOR

Mark I (RS-232)	$79.00
Mark II (Atari)	$79.00
Mark III (T..I. 99)	$109.00
Mark IV (CBM-PET)	$125.00
Mark V (Osborne)	$95.00
Mark VI (IBM-PC)	$179.00
Mark VII (Auto Ans/Auto Dial)	$119.00
Mark VIII	$269.00
TRS-80 Color Computer	$99.00
9 Volt Power Supply	$9.00

EAGLE

IIE-1	$1369.00
IIE-2	$1649.00
IIE-3	$2399.00
IIE-4	$3199.00
PC-E	$1579.00
PC-1	$2399.00
PC-2	$2799.00
PC-XL	$3599.00
1620	$3599.00
1630	$5499.00
1640	$6499.00
Cyma Software	CALL

MONITORS

AMDEK

300G	$149.00
300A	$159.00
310A	$169.00
Color I	$279.00
Color I plus	$299.00
Color II	$399.00
Color III	$349.00
Color IV	$999.00

USI

Pi 1, 9" G	$99.00
Pi 2, 12" G	$119.00
Pi 3, 12" A	$159.00
Pi 4, 9" A	$139.00
1400 Color	$299.00

ZENITH

ZVM 121	$95.00
ZT1 Terminal	$369.00

BMC

12" Green	$85.00
9191 13" Color	$299.00

TAXAN

12 N Green	$129.00
12 A Amber	$139.00

PANASONIC

TR 120 Hi-res. Green	$159.00
CT 160 Dual Mode Color	$299.00

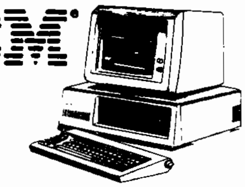
NEC

JB 1260	$119.00
JB 1201	$149.00
JC 1212	$299.00
JC 12-202	$299.00
JC 1203	$469.00

GORILLA

12" Green	$89.00

IBM



NEC 3550 PRINTER... $1799

PERCOM/TANDOM DRIVE

5¼" 320K Floppy	$279.00
10 Meg Hard	$1495.00

AMDEK

310A Amber Monitor	$169.00
DXY 100 Plotter	$599.00
Color II	$399.00

AST

Combo Plus	CALL
Mega Plus	CALL
Mega Pack	CALL
I/O Plus	CALL

PROFESSIONAL SOFTWARE

PC Plus Word Processing	$319.00

MICRO PRO

Word Star/Mail Merge	$319.00
InfoStar	$299.00
Spell Star	$159.00
CallStar	$159.00

MICROSTUF

Crosstalk	$139.00

MICROSOFT

Multiplan	$199.00

ASHTON-TATE

D-Base II	$419.00

IUS

EasyWriter II	$209.00
EasySpeller	$129.00
EasyFiler	$129.00

CONTINENTAL SOFTWARE

1st Class Mail/Form Letter	$89.00
The Home Accountant Plus	$109.00

SYNAPSE

File Manager	$119.00

LOTUS

123	$369.00

PFS

	APPLE	IBM
File	$89.00	$99.00
Report	$89.00	$89.00
Graph	$89.00	$99.00
Write	n/a	$99.00

KRAFT

IBM Joystick	$55.00
IBM Paddles	$39.00

QUAD RAM .... CALL

SANYO

SANYO

MBC-555	$795.00
MBC-1000	$1599.00
MBC-160 Drive	$539.00
PR5500 Letter Quality Printer	$699.00

PRINTERS

EPSON

MX80 FT, MX100	CALL
RX80	CALL
FX80, FX100	CALL

OKIDATA

82, 83, 84	CALL
92, 93	CALL

STAR

Stx 80	$169.00
Gemini 10	$299.00
Gemini 15	$479.00
Serial Board	$75.00

SMITH CORONA

TP-1	$469.00
TP-2	CALL
Tractor Feed	$129.00

C.ITOH

Gorilla	$209.00
Prowriter 8510P	$379.00
Prowriter 1550P	$689.00
Starwriter F10-40P	$1149.00
Printmaster F10-55P	$1569.00
Tractor Feed	$109.00

DIASYWRITER

2000 Letter Quality	$1149.00
2500...."NEW"	CALL
Tractor Feed	$109.00

DIABLO

620	$949.00
630	$1769.00

IDS

Call for ALL Configurations on IDS PRISM PRINTERS.

NEC

8023	$399.00
7710/7730	$2149.00
3510/3530	$1549.00

HEWLETT PACKARD

HP41CV... $209.00

HP 75........... $749.00

HP 41C	$146.00
HP 10C	$52.00
HP 11C	$69.00
HP 12C	$92.00
HP 15C	$92.00
HP 16C	$92.00

For HP41/41CV

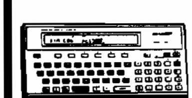
HPIL Module	$99.00
HPIL Cassette or Printer	$359.00
Card Reader	$144.00
Extended Functions Module	$64.00
Time Module	$64.00

TI-40 COMPACT COMPUTER $219

TIMEX SINCLAIR 1000 $39.95

16K Memory	$44.95
2040 Printer	$99.95
Vu-Calc	$17.95
Check Book Manager	$13.95
Organizer	$14.95
Budgeter	$13.95
Stock Option	$14.95
Loan & Mortage Amortizer	$12.95
Mindware Printer	$99.00

SHARP



PC-1500A POCKET COMPUTER $169.00

PC-1250A........ $89.00

125 Printer/Micro Cassette	$129.00
150 Printer/Plotter/Cassette	$172.00
151 4K RAM	$55.00
152 Cassette Recorder	$62.00
155 8K RAM	$94.00
158 8K RAM Battery	$129.00
161 16K RAM	$139.00

CABLES & CONNECTIONS

PRINTER CABLES

Atari to Parallel	$29.00
Atari to Serial	$29.00
Apple to Parallel	$69.00
Apple to Parallel/Graphics	$99.00
Apple to Serial	$89.00
IBM to Parallel	$35.00
IBM to Serial	$29.00
Parallel to Parallel	$29.00
Serial to Serial	$29.00
Grappler Plus	$129.00
PKASO	$139.00
Atari to Modem Cable	$29.00
CBM 64 to IEEE Board	$79.00
Apple 80 Column Card	$159.00
CBM Pet to Parallel	$99.00
CBM Pet to Serial	$89.00

READY FORMS

1or2" Address Labels (Tract. Feed)	$9.95
15" Report Paper (Tract Feed)	$24.95
8½" Blnk Wht Paper (Tract. Feed)	$19.95

computer mail order west

ORDERS ONLY

VISA

800-648-3311

In NV call (702)588-5654 Order Status Number: 588-5654

Dept. 915, P.O. Box 6689, Stateline, NV 89449

No risk, no deposit on C.O.D. orders. Pre-paid orders receive free shipping within the UPS Continental United States with no waiting period for certified checks or money orders. Add 3%(minumum $3.00) shipping and handling on all C.O.D. orders and credit card orders. Larger shipments may require charges. NV and PA residents add sales tax. All items subject to availability and price change. NOTE: We stock manufacturer's and third party software for most all computers on the market. Call today for our new catalog.

computer mail order

FRANKLIN

ACE 1000 Color Computer
ACE 1100 Drive & Cover for ACE 1000
ACE 1200 Computer with Disk Drive
ACE PRO PACK:
ACE 1000, Disk Drive,
80 Column Card, ACECalc &
ACEWriter II

MICRO-SCI
Apple & Franklin

A2	$249.00
A40	$349.00
A70	$459.00
C2 Controller	$79.00
C47 Controller	$89.00

RANA

Elite I (Apple/Franklin)	$279.00
Elite II (Apple/Franklin)	CALL
Elite III (Apple/Franklin)	CALL

VISICORP
FOR APPLE, IBM & FRANKLIN

Visidex	$189.00
Visifile	$189.00
Visiplot	$159.00
Visiterm	$89.00
Visitrend/Plot	$229.00
VisiSchedule	$229.00
Desktop Plan	$189.00
Visicalc (Apple, CBM, IBM)	$179.00

Visicorp prices for IBM may vary slightly.

LJK

Letter Perfect Apple	$109.00
Data Perfect Apple	$75.00

AXLON

Apple/Franklin 128K Ram	$299.00
Apple/Franklin Ram Disk	$729.00

MPC

Bubdisk (128K Non Volitare)	$649.00

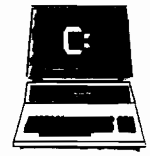
JOYSTICKS
WICO

Joystick	$21.95
Famous Red Ball	$23.95
Power Grip	$21.95
Three-Way Deluxe	$22.95
Atari/VIC Trackball	$34.95
Apple Trackball	$59.95

KRAFT

Atari Single Fire	$13.00
Atari Switch Hitter	$16.00
Apple Joystick	$44.00
Apple Paddles	$39.00

commodore



CBM 8032 $599

CBM64 .... $229
VIC 20 .... $99
CALL ON
Executive 64 Portable

1520 Color Printer/Plotter	$169.00
1525 80 Column Printer	$219.00
1526	$319.00
1530 Datasette	$69.00
1541 Single Disk Drive	$249.00
1600 VIC Modem	$59.00
1650 AD/AA Modem	$89.00
1701 14" Color Monitor	$249.00
Pet 64	$569.00
Pet 4032	$599.00
CBM 8032	$599.00
Super Pet	$999.00
B128-80	$769.00
BX256-80	$969.00
2031	$299.00
4040	$699.00
8050	$949.00
8250	$1199.00
9060	$1999.00
9090	$2199.00
4023	$379.00
6400	$1399.00
64K Upgrade	$269.00
Spell Master	$149.00
Z-Ram	$549.00
Silicon Office	$749.00
The Manager	$209.00
Soft Rom	$129.00
Jinsam	CALL

PROFESSIONAL SOFTWARE

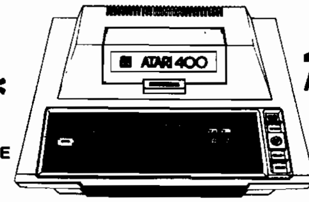
Word Pro 2 Plus	$159.00
Word Pro 3 Plus	$189.00
Word Pro 4 Plus	$279.00
Word Pro 5 Plus	$279.00
InfoPro	$179.00
Administrator	$399.00
Power	$79.00
Word Pro 64 Plus	$65.00

CAROCO

Light Pen	$32.00
Cassette Interface	$29.00
Parallel Printer Interface	$69.00
3 Slot Expans. Interface (20)	$32.00
6 Slot Expans. Interface (20)	$79.00

ATARI HOME COMPUTERS
ATARI 400

$29*



* Your FINAL PRICE after REBATE when purchased with package as shown.

ATARI 400 & 410 Recorder Package
YOU PAY $179.00
You Get ATARI REBATE $50.00
YOUR TOTAL COST $129.00
Retail Value of ATARI 410 $100.00
Net Cost of ATARI 400 $29.00
CALL FOR PRICING ON
THE ENTIRE ATARI XL LINE!

1010 Program Recorder	$74.00
1020 40 Col. Printer/Plotter	$249.00
1025 80 Col. Printer	$449.00
1027 Letter Quality Printer	$299.00
1050 Disk Drive	$379.00
850 Interface	$169.00
1030 Direct Connect Modem	CALL
CX30 Paddles	$12.00
CX40 Joystick	$8.00
CX42 Remote Joystick	CALL
CX77 Touch Tablet	$69.00
CX80 Trak Ball	$49.00
CX85 Keypad	$105.00
CX418 Home Manager	$69.00
CX488 Communicator II	$229.00
KX7098 Atari Accountant	$209.00
KX7101 Entertainer	$69.00
KX7102 Arcade Champ	$75.00

ALIEN

Atari Voice Box	$119.00
Apple Voice Box	$149.00

MEMORY

Axlon 32K Ram	$59.00
Axlon 48K Ram	$99.00
Axlon 128K Ram	$299.00
Intec 32K Board	$59.00
Intec 48K Board	$85.00
Intec 64K Board (400 only)	$99.00
Intec Real Time Clock	$39.00

INNOVATIVE CONCEPTS

Flip-n-File 10	$4.00
Flip-n-File 50	$19.00
Flip-n-File 400/800 Cart	$19.00

PERCOM
DISK DRIVES FOR ATARI

AT88-S1	$369.00
AT88-A1	$299.00
AT88-S2	$569.00
AT88-S1 PD	$469.00
AT88-S2 PD	$669.00
AT88-DDA	$139.00
AT44-S1	$579.00
AT44-S2	$969.00

TEXAS INSTRUMENTS DRIVE

TX 99-S1	$349.00

RANA

1000 Atari Disk Drive	$319.00

FLOPPY DISKS
MAXELL

MD-1	$32.00
MD-2	$44.00
FD-1(8")	$40.00
FD-2(8" DD)	$50.00

ELEPHANT

5¼" SS SD	$18.95
5¼" SS DD	$24.95
5¼" DS DD	$29.95

VERBATUM

5¼" SS DD	$26.00
5¼" DS DD	$36.00

HEAD

Disk Head Cleaner	$14.95

C.M.O. TOP 80

APPLE/FRANKLIN

1. Choplifter	$27.00
2. Bank Street Writer	$55.00
3. PFS: File	$89.00
4. Visicalc	$179.00
5. Home Accountant	$55.00
6. Zaxxon	$29.00
7. Most Amazing Thing	$28.00
8. Visifile	$189.00
9. Fathoms 40	$19.00
10. Deadline	$35.00
11. PFS: Report	$89.00
12. Zork III	$29.00
13. Frogger	$24.00
14. Facemaker	$24.00
15. Snooper Troops #1	$32.00
16. Delta Drawing	$35.00
17. Castle Wolfenstine	$24.00
18. Wayout	$29.00
19. Canyon Climber	$19.00
20. Bandits	$26.00

CBM 64

1. Word Pro 64	$65.00
2. Jumpman	$29.00
3. Gorf (20/64)	$14.95
4. Microspec Data Base 64	$89.00
5. Logo 64	$39.00
6. Microspec Gen. Ledger 64	$79.00
7. Zork III	$29.00
8. Frogger (64)	$23.00
9. Quick Brown Fox (20/64)	$49.00
10. Shamus	$29.00
11. Deadline	$29.00
12. Assembler 64	$14.95
13. Zork II	$29.00
14. 3-D Man	$14.00
15. Protector	$32.00
16. Starcross	$29.00
17. Easy Mail 64	$14.95
18. Grave Robber	$11.00
19. Wall Street	$19.00
20. Trash Man	$32.00

ATARI

1. Donkey Kong	$39.00
2. Zaxxon	$29.00
3. E.T. Phone Home	$39.00
4. Miner 2049er	$35.00
5. Dig Dug	$33.00
6. Choplifter	$29.00
7. Donkey Kong, Jr	$39.00
8. Canyon Climber	$25.00
9. Snooper Troops #2	$34.00
10. Word Wizard	$59.00
11. Picnic Paranoia	$34.00
12. Jumpman	$29.00
13. Shamus	$34.00
14. Letter Perfect	$109.00
15. File Manager 800	$69.00
16. Preppie	$24.00
17. Astro Chase	$25.00
18. Blade/Black Hole	$27.00
19. Pac Man	$33.00
20. Baja Buggies	$25.00
21. Crush, Crumble & Chomp	$24.00
22. Wayout	$27.00
23. Zork II	$29.00
24. Visicalc	$159.00
25. Atari Writer	$79.00
26. Three Little Pigs	$25.00
27. Upper Reaches of Apshai	$16.00
28. Starbowl Football	$24.95
29. Dreibs	$26.00
30. Protector	$34.00
31. Frogger	$25.00
32. Lunar Leeper	$24.00
33. Wizard of Wor	$34.00
34. Kindercomp	$21.00
35. Moon Shuttle	$22.00
36. Home Accountant	$55.00
37. Temple of Apshai	$29.00
38. Spell Wizard	$39.00
39. Nautilus	$26.00
40. O'Riley's Mine	$22.00

computer mail order east

ORDERS ONLY

800-233-8950

MasterCard

In PA call (717)327-9575 Order Status Number: 327-9576

Dept. 915, 477 E. 3rd St., Williamsport, PA 17701

INTERNATIONAL ORDERS: All shipments outside the Continental United States must be pre-paid by certified check only. Include 3% (minumum $3.00) shipping and handling. **EDUCATIONAL DISCOUNTS:** Additional discounts are available from both Computer Mail Order locations to qualified Educational Institutions. **APO & FPO:** Add minumum $5.00 shipping and handling.

New Publications

Microcomputers, A Parent's Guide, by Kenneth P. Goldberg and Robert D. Sherwood. John Wiley & Sons, Inc., 605 Third Ave., New York, NY 10158, 1983, 196 pages, paperback.
0-471-87278-4 $8.95

Swift's Educational Software Directory, Apple II Edition. Sterling Swift, 1701 South I-35, Austin, TX 78744, 1983, 478 pages, paperback, wire-o-bound.
0-88408-270-9 $18.95 plus $1.75 S/H

The Reader's Guide to Microcomputer Books, by Michael Nicita and Ronald Petrusha. Golden-Lee Book, 1000 Dean St., Brooklyn, NH 11238, 1983, 409 pages, paperback.
0-912331-00-3 $9.95

Computer BASICS, An Introduction to the Computer for Young People, by Hal Hellman. Prentice-Hall, Englewood Cliffs, NJ 07632, 1983, 48 pages, hardcover.
0-13-164574-9 $8.95

The Complete Handbook of Personal Computer Communications, by Alfred Glossbrenner. St. Martins Press, 175 Fifth, New York, NY 10010, 1983, 325 pages, paperback.
0-312-15718-5 $14.95

Mastering the VIC-20, by A. J. Jones, E. A. Coley, and D. G. J. Cole. John Wiley & Sons, Inc., 605 Third Ave, New York, NY 10158, 1983, 177 pages, paperback.
0-471-88892-3 $14.95

Commodore 64 Programmer's Reference Guide. Commodore Business Machines, Inc., and Howard W. Sams & Co., Inc., 4300 W. 62nd St., Indianapolis, IN 46268, 486 pages, paperback, wire-o-bound.
0-672-22056-3 $19.95

Atari BASIC, by Richard Haskell. Prentice-Hall, Inc., Englewood Cliffs, NJ 07632, 1983, 173 pages, paperback.
0-13-049791-6 $13.95

TRS-80 Extended Color BASIC, by Richard Haskell. Prentice-Hall, Inc., Englewood Cliffs, NJ 07632, 1983, 170 pages, paperback.
0-13-931246-3 $12.95

40 Easy Steps to Programming in BASIC & Logo, by James L. Poirot and R. Clark Adams. Sterling Swift, 7901 South IH-35, Austin, TX 78744, 1983, 64 pages, paperback.
0-88408-275-X $3.95

MICRO™

MICRO™

Hardware Catalog

The UltraTerm
for the Apple II+, Apple III, and Apple IIe

Videx introduces a new video display card that gives extraordinary powers to the Apple II+, Apple II, Apple IIe, and Franklin computers. The **UltraTerm** sets new standards for versatility. It delivers a high-quality 8 x 12 dot-character dot matrix with stable, flicker-free display that guarantees easy readability. Nine software-selectable video display formats allow as many as 4096 characters to be displayed.

Software-selectable display attributes include bright/dim, standard/alternate character sets, and normal/inverse. Interlace mode (512 scan lines) requires a monitor with high persistence phospher such as the Apple Monitor III or Amdek 300 Monitor.
$379.00

Videx, Inc.
897 N.W. Grant
Corvallis, OR 97330
(503)758-0521

Low-Cost Data Collection and Entry

The **1100A** optical card reader is designed to meet the needs of the burgeoning microcomputer and minicomputer markets. The desktop unit provides means of data entry. The RS-232C interface allows interaction with virtually any computer, either directly or remotely through a modem.

The 1100A is designed to meet the needs of education with applications including test scoring, attendance reporting, and administration.

The 1100A can read any combination of pencil marks, punched holes and printed marks. The unit can read soft lead pencils too, negating outmoded IBM-type magnetic lead pencils.

Mountain Computer, Inc.
300 El Pueblo Rd.
Scotts Valley, CA 95066
(408)438-6650

New Briefcase Computer
for the Commodore 64

Commdore Business Machines, Inc., has introduced a portable computer designed for the traveling businessman. Designated the **Executive 64**, the new portable has 64K RAM, a full upper/lower-case low-profile detachable keyboard, built-in 6-inch color monitor, and a built-in single floppy disk drive with 170K capacity. A second drive is optional. The new unit weighs 27.6 pounds and is briefcase size - 5" x 14-½" x 14-½".

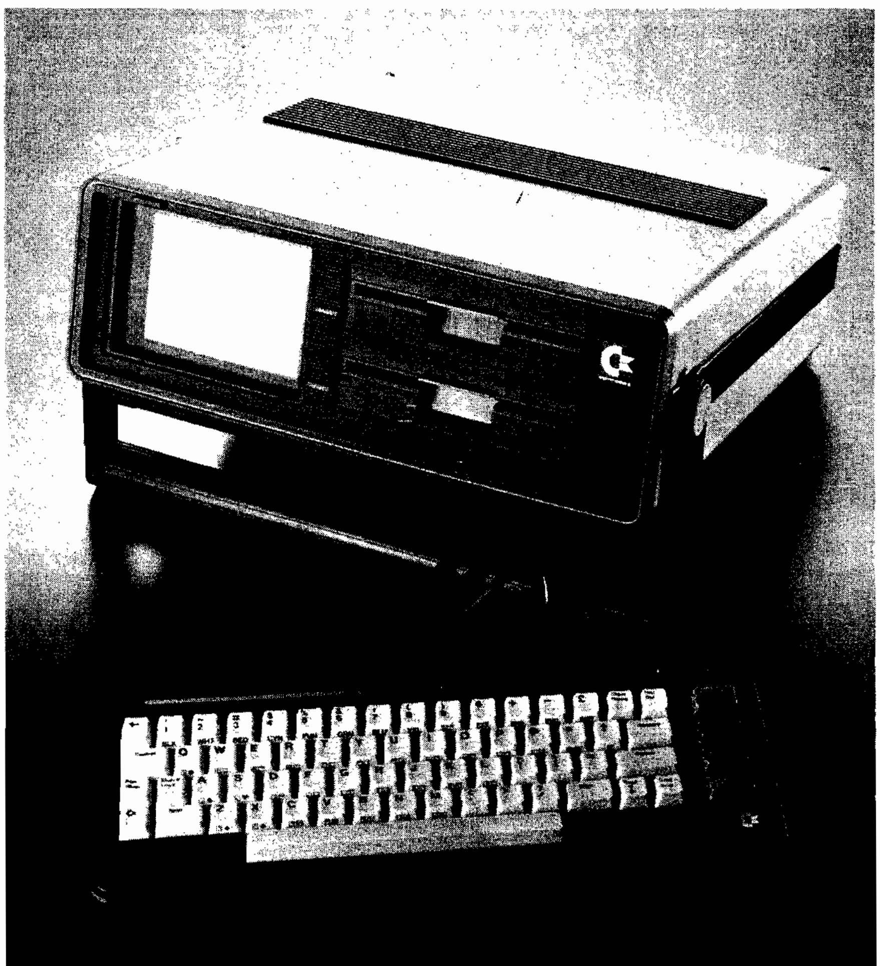
The Executive 64 is fully compatible with VIC-20 and C64 peripherals, including the VIC modem for telecommunications. External ports allow full-sized monitor and graphic printer hook-ups.

Using a Z-80 cartridge, the Executive 64 can accommodate a CP/M operating system or, with a PET Emulator, the system can use much of the available PET software. Moreover, the Executive 64 can use the large number of game cartridges available to the 64 family of computers and has full music and sound capabilities.

Resident in the unit's ROM is BASIC V2. Other high-level programming languages include Pascal, LOGO, COMAL, Assembler, and PILOT. Additionally, the Executive 64's 6510 central processor is 6502-program compatible.
$995.00

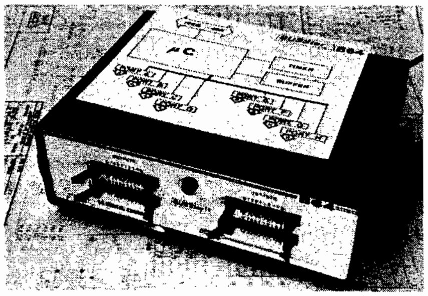
Commodore
Computer System Division
1200 Wilson Drive
West Chester, PA 19380;
(215)431-9100

Commodore Business Machines' Executive 64 System



IEEE-488 64 Digital Channel Output Module

for the Apple, IBM, and Commodore



The BUSSter—an IEEE-488 64 Digital Channel Output Module

Connecticut microComputer announces the **BUSSter B64**, which works with any computer that has an IEEE-488 interface (either built-in or added on), including computers manufactured by Apple, IBM, Commodore, Osborne, Hewlett-Packard, and Tektronix. The B64 is a new 64 digital line output module that is a self contained IEEE 488 (GPIB) bus-compatible device.

The BUSSter accepts commands and data from any host computer through its IEEE port and activates 1 to 64 digital TTL level lines. A BUSSter module economically increases a computer's interfacing capability while reducing its workload. The BUSSter B64 Digital Output Module is easily programmed through BASIC commands from the controlling computer.
$495.00

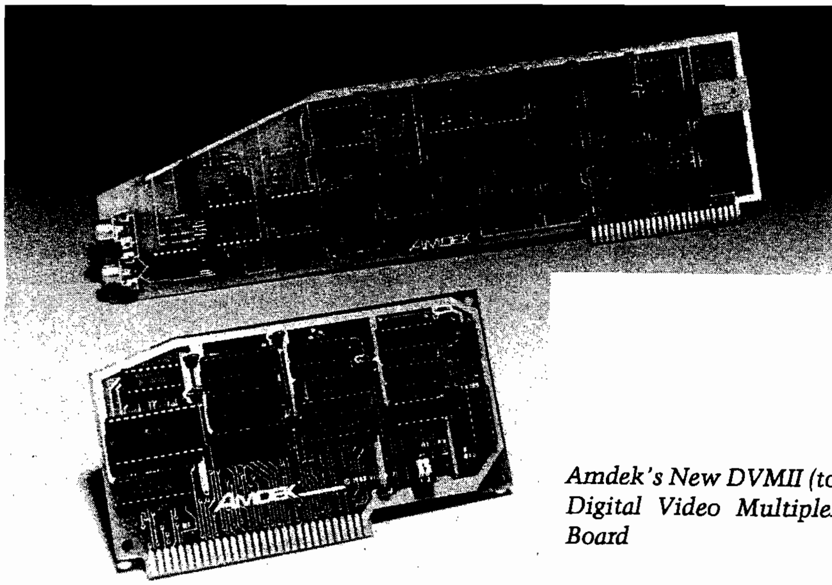
Connecticut microComputer
36 Del Mar Drive
Brookfield, CT 06804
(203)775-4595

Spectrum Control Center
for the Color Computer

Spectrum Projects announces their newest product, the **SPECTRUM CONTROL CENTER.** The SCC eliminates having to grope around behind your computer, fumbling with cables and plugs, by extending all the jacks to an interface box, which can be mounted anywhere. Features of the SPECTRUM CONTROL CENTER are a switch-selected Printer/Modem port, a cassette switch for those people who don't own CTR-80 and CCR-81 cassette recorders, and a LED power indicator to tell you whether your computer is on or off.
$99.95 plus $3.00 S/H.

SPECTRUM PROJECTS
93-15 86th Drive
Woodhaven, NY 11421
Voice (212)441-2807
B.B.S. (212)441-3755

(continued)



Amdek's New DVMII (top), Digital Video Multiplexor Board

Digital Video Multiplexor Supplies RGB Output for analog or Digital Monitors
for the Apple II and Apple II+

A new Digital Video Multiplexor Board, **DVM-II**, which plugs into any expansion slot of an Apple II or Apple II+ computer, is designed to supply RGB output for analog or digital monitors. The new board features 15 low-resolution colors (16 colors with an analog monitor), two additional high-resolution modes, all white, and three-color with pure white, and it is color-gun selectable for all green or other colored text. Boards with analog monitors a bits 1-4 selectable for all green or other colored text.

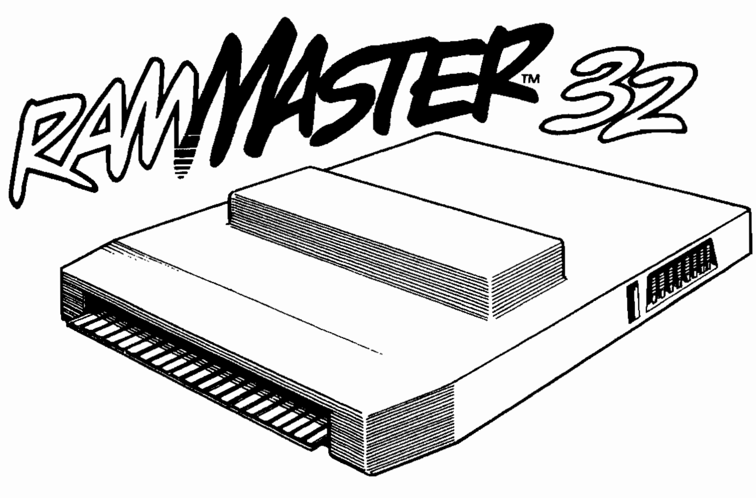
The DVM-II is expansion-slot independent and, with a simple plug-in installation, can supply the computer with RGB output. The board uses Apple NTSC Video output. It provides 80-column capabilities in high-resolution colors with the use of any RS-170 output, 80-column card. The DVM-II includes two connector cables and one video monitor connector cable to allow the board to be adapted to the monitor. Depending on the type of monitor, the adapter must be preset to allow the appropriate composite Csync signal output. The functioning of the DVM-II is controlled by software switches that are slot dependent.
$199.00

Amdek Corporation
2201 Lively Blvd.
Elk Grove Village, IL 60007
(312)446-5248

UNDER-STAND
for the Apple II, II+, and IIe

Verba Gloria announces **UNDER-STAND**, the latest product in their line of clear, acrylic aids for the Apple II, Apple II+, and Apple IIe. The UNDER-STAND is a space-saving monitor stand constructed of ultra-strong 3/8" clear acrylic. It has less flex and allows for better convective cooling than other stands, and it can be easily modified by the maker to accept an Apple attached cooling fan. The UNDER-STAND holds one or two drives, plus paddles or joystick on the center shelf, with the strength to hold a weighty monitor on top. The Apple can be slid out from underneath for easy access to peripheral cards.
$71.95

Verba Gloria
802 Twelfth Ave.
Menomonie, WI 54751



The RAMMASTER 32 for the VIC-20

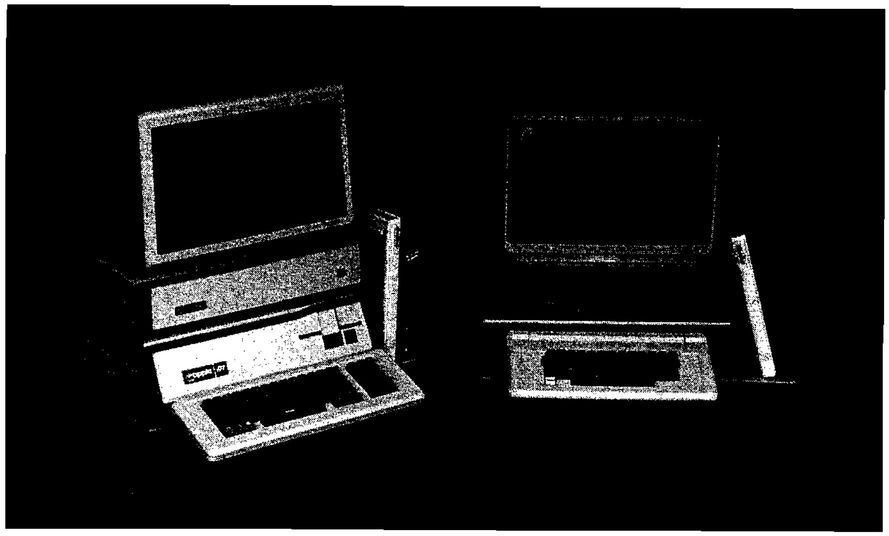
RAMMASTER 32
for the VIC-20

Mosaic Electronics introduces the **RAMMASTER 32.** This is a full service memory device that features a built-in expansion port, pause switch, write protect switch, a relocatable memory block, and a disabler switch so cartridges can be removed without turning off the computer. RAMMASTER 32 will expand the VIC-20 up to 37K.
$150.00

Mosaic Electronics
P.O. Box 708,
Oregon City, OR 97045
(800)547-2807 or 665-9574

Hardware Catalog *(continued)*

Pro-Tech Security
for the Apple II and III



Pro-Tech Security for Your Apple II or Apple III

Now you can protect your complete Apple II and Apple III systems from tampering and theft with two security designs from Segull Enterprises. **Pro-Tech II** secures the Apple II and cover, up to three disk drives, any type of monitor or TV, and is compatible with the popular external fans as well as securing them. **Pro-Tech III** secures the Apple III and cover, up to three disk drives or a Profile hard disk, and any size monitor or TV.

The Pro-Tech Locking Stands feature a rear-locking system that combines total security with ease of use. Simply slide in your Apple and disk drives, lock it, and you're done. The locking stands are made of 16-gauge steel and are color coordinated to Apple computers. For extra convenience, multiple units keyed the same are available.

Monitors are secured to the Pro-Tech Locking Stand by a 22" steel cable.
$155.00 for the Apple II
$165.00 for the Apple III
$99.00 for printers

Segull Enterprises
88 West Britannia Street
P.O. Box 869
Taunton, MA 02780
(617)823-9684 ext. 331

MICRO

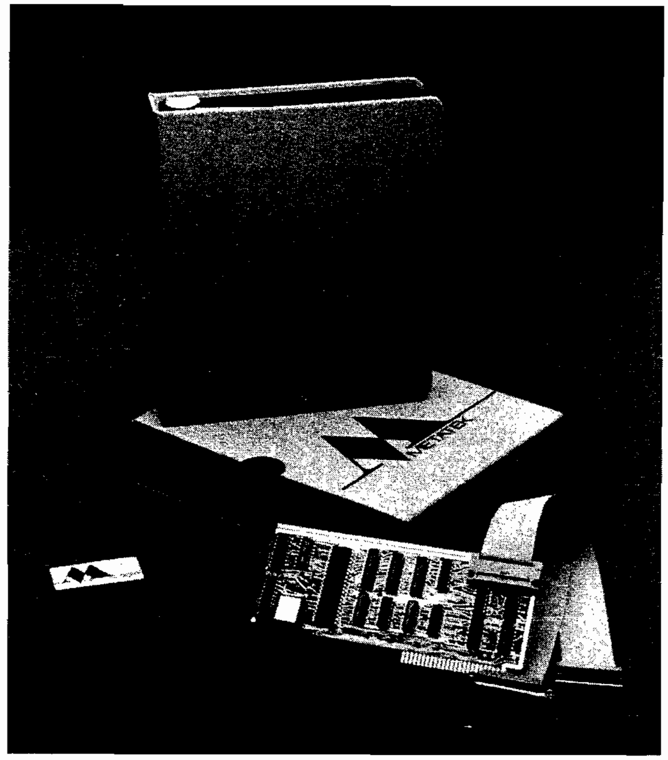
MICRO™

Software Catalog

Metatek Introduces Low-Cost Datascope

Metatek, Inc. announces **Metascope**, a low-cost data-line monitor designed to operate on the *Apple II*. The product consists of a printed circuit board, documentation, and all software necessary to turn an Apple II computer into a fully programmable data-line monitor. The unit is capable of displaying and storing data in asynchronous, byte-oriented synchronous, or bit-oriented synchronous (SDLC, HDLC) modes at speeds to 19.2K bits per second. Other features include the ability to start data recording based on a trigger pattern match, storage of data on diskette, and a programmable host emulation mode that allows the Apple II to act as a sophisticated communications controller capable of generating polling sequences with reply. Additionally, Metascope has a built-in capability to generate synchronous clock signals in host emulation mode thereby eliminating the need for costly modem emulators.

The retail price is **$895.00** with delivery two weeks ARO. The product is available from Metatek, Inc., 12525 Hummingbird St. NW, PO Box 33129, Minneapolis, MN 55433; (612)755-9587.



Class Scheduling Program

CMA Micro Computer announces a new version of its popular **Class Scheduling Program** for the *48K Apple with Applesoft*. The advanced version offers new editing procedures for editing groups of courses in the master schedule. The Class Scheduling Program allows schools with up to 2,400 students and up to 999 courses and sections of courses to analyze the master schedules and prepare individual student schedules. The system allows for the automatic entry of required courses and the fast entry of any optional request and alternates.

The system will schedule individual students and report anyone not scheduled in a requested course. Non-scheduled students can be given new requests and rescheduled until all students have been scheduled. Schedules can be printed for all students and rosters prepared for all courses offered. Adds and drops can be easily handled with forced scheduling or by rescheduling of individual students using the program's automatic scheduling elements.

The system is designed to work with the firm's Grading programs and Attendance bookkeeping system. The system requires two disk drives and an 80-column printer.

Additional information and demonstration versions are available from CMA dealers or directly from CMA Micro Computer, 55722 Santa Fe Trail, Yucca Valley, CA 92284; (619)365-9718.

Airplane Simulator

AIRSIM-3 Airplane Simulator for the *Apple II, Apple II+, or Apple IIe with 48K* is an aerobatic flight simulator with ground scenery and all the instrumentations required to practice instrument flight. Users can set up their own approach problems, complete with runway scenery and Nav-Aid locations. Pilots will find AIRSIM-3 useful for instrument-flight practice. Nonpilots will find AIRSIM-3 to be a simple enjoyable flight-like experience.

Price is **$40.00** and includes diskette and manual. Contact Ted Kurtz, Mind Systems Corporation, P.O. Box 506, Northampton, MA 01061; (413)586-6463.

Doing the School Yearbook with Your Apple

Single SOURCE Solution announces an Apple Computer educational product called **Yearbook.** This tutorial program is written to emphasize the basic principles and common technical practices necessary to the publishing and production of yearbooks. Yearbook delves into the world of layout, editing, and vocabulary. Designed for the novice, no computer skills are necessary to use the program. Applesoft high-resolution graphics are used extensively in this series of programs.

Price is **$99.95.** For more information contact Single SOURCE Solution, 2637 Pleasant Hill Road, Pleasant Hill, CA 94523.

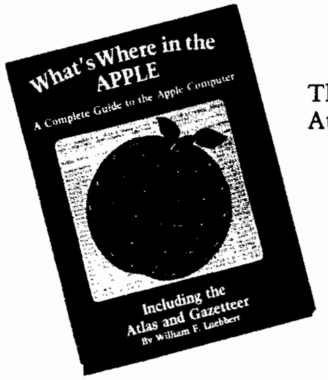
(Continued on page 129)

PRODUCTS from MICRO

Revised edition now available! All new text added to the original atlas and gazetter...

"What's Where in the Apple"

A Complete Guide to the Apple Computer.



This revised edition of the famous Apple Atlas will:

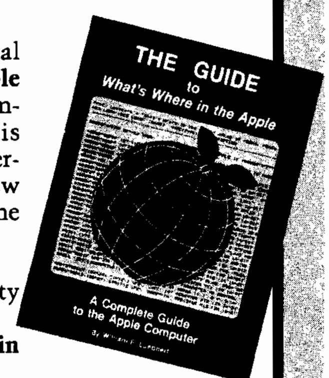
- provide you with more information on the Apple's memory than is available elsewhere
- guide you with a numerical Atlas and an alphabetical Gazette — to over 2000 memory locations of PEEKs, POKEs and CALLs
- give you names and locations of various Monitor, DOS, Integer BASIC and Applesoft routines and tells you what they're used for
- enable you to move easily between BASIC and machine language
- guide you through the inner workings and hidden mechanisms of the Apple

The Atlas and the All New Guide are now available in one 256-page Wire-O-Bound book for only $24.95.

All Apple users will find this book helpful in understanding their machine and essential for mastering it.

Retail price $24.95

If you own the original **What's Where in the Apple** purchase **The Guide** to complete your edition. This 128-page Wire-O-Bound version contains all the new material to be used with the memory map and atlas.



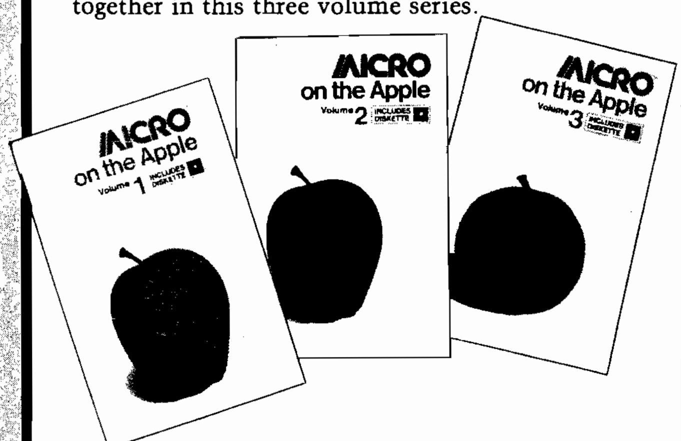
Don't miss this opportunity to complete your edition of **What's Where in the Apple.**

Order your copy of The Guide now.

Retail price $9.95

Get the most out of your Apple with the "MICRO on the APPLE" series...

This series of books for the APPLE computer is the best source of Apple specific programming aids, utilities, enhancements, games and much more. MICRO has taken the best APPLE material ever published in MICRO magazine and brought it together in this three volume series.



Volume 1 allows you to —
Round and format numbers accurately, get lower-case letters and punctuation into Applesoft strings or play the hit game "Spelunker"

Volume 2 lets you —
Speed up machine-language programming using five powerful machine-language aids, add additional editing and I/O features, or play the intriguing game "GalactiCube"

Volume 3 gives you more —
Machine-language aids, I/O enhancements, graphics and games

Each volume comes complete with diskette to save you the time of typing hundreds of lines of code.

If you use to an APPLE computer you should own these books.

Retail price $24.95

FOR EASY ORDERING
CALL TOLL FREE
1-800-345-8112
(IN PA 1-800-662-2444)
MasterCard and VISA Accepted

AND now the "MICRO on the APPLE" series is available as a boxed set.

All three voumes are packaged in an attractive box with the 110 programs on one diskette.

The set is available at a **20% savings** off the individual price.

Retail price $59.95

Order your copy of the

Best of MICRO

before they are sold out!

If you're looking for excellent software at a very low price these books are for you!

Here are some of the things that you will receive:

Volume 1 — SOLD OUT!

Volume 2 contains 55 articles and programs that appeared in MICRO magazine from October 1978 through May 1979. These programs are for the Apple, AIM 65, KIM-1, PET, OSI, and SYM-1. This 224-page book is paper bound 8½ x 11.
Retail price $8.00

Volume 3 contains 88 programs and articles from June 1979 through May 1980 issues of MICRO magazine. These programs are for use on Apple, AIM 65, KIM-1, PET, OSI, and SYM-1 computers. This 320-page book is 8½ x 11 and is paperbound.
Retail price $10.00

Learn how to master VIC BASIC programming with MICRO's newest book...

"Mastering Your VIC-20"

Now you can do more with your VIC-20. This new book and the 8 projects and 20 programs that it contains can teach you how to master VIC BASIC programming. Each chapter concentrates on a particular aspect of VIC BASIC...and each program is accompanied by discovery-oriented, tutorial text — clear directions that will quickly have you writing programs, modifying them and adding features all on your own. And to help you master your VIC-20 even faster, all 8 programs are already keyed in on the accompanying cassette.

Some of the programs you'll find in this book are:

- MICRO Calc — a minature spread sheet program that makes complex, repetitive calculations a breeze.
- Master — a guessing game that teaches programming with random numbers and flags.
- VIC Clock — to teach you ON...GOSUB functions and character graphics.

Order your copy of "Mastering Your VIC-20" by using the order form below.
Retail price $23.95

Each program is worth the price of the book itself.

ATTENTION OSI USERS

Micro has just released a new book, "MICRO on the OSI." This comprehensive book provides you with methods to improve the use of your Ohio Scientific Computer ...includes machine-language enhancements and BASIC aids. You can learn how to TRACE BASIC programs, DEBUG machine language programs and improve your OS65-D operating system.

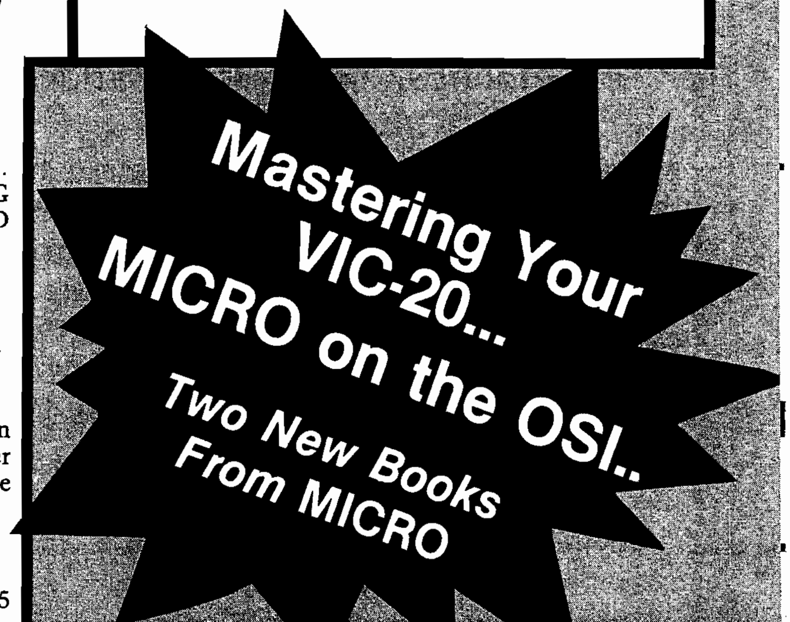
You'll also find:

- Hardware modifications for enhanced/reversed video
- Programs for control code and upper/lower-case entry

Plus... a What's Where in the OSI C1/C2

This new book will also help you to improve your own BASIC programs with PRINT AT, DELETE, AUTO-number and FUNCTION INPUT. No OSI owner can afford to be without this book.

You'll receive 25 programs in this 192 page book.
Order your copy today!
Retail price $19.95



USE THE POSTAGE-PAID CARD TO ORDER YOUR BOOKS FROM MICRO....

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 20, AMHERST, NH

POSTAGE WILL BE PAID BY ADDRESSEE

MICRO™

P.O. Box 6502
Amherst, NH 03031

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 20, AMHERST, NH

POSTAGE WILL BE PAID BY ADDRESSEE

MICRO™

P.O. Box 6502
Amherst, NH 03031

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 20, AMHERST, NH

POSTAGE WILL BE PAID BY ADDRESSEE

MICRO™

P.O. Box 6502
Amherst, NH 03031

Computer Grading Package for the Classroom Teacher

Grade Manager, a powerful and highly flexible computer grading package for the classroom teacher, is now available for the *48K Apple II* with either one or two disk drives and a printer. The package allows teachers to record and print 50 scores per student per quarter and to compute letter grades for up to 1088 students in as many as 15 separate classes. Teachers can choose from several variations of two grading methods — percentage grading and curve grading — making the package compatible with a wide variety of grading methods currently in use.

First-time users are guided step-by-step through the process by a comprehensive support manual that also explains the differences in the grading options. The package uses a data diskette as well as the program diskette.

For more information contact Minnesota Educational Computing Consortium, 2520 Broadway Drive, St. Paul, Minnesota 55113; (612)638-0602.

Attention Special Education Teachers!

The World of Counting, written for the *Apple II+ or Apple IIe* computer (with one disk drive), is an award winning program just introduced by Educomp Enterprises. This program is specifically designed to assist in the teaching of learning-disabled students. It utilizes colorful graphics, sound effects, and music to teach the beginning principles of counting. The World of Counting is also effective with any child in the 3- to 7-year-old mental-age group. This makes it a valuable addition to any program library for preschoolers.

Designed to be more than just a drill and practice program, it uses extensive repetition and reinforcement to introduce the numbers involved, review them, then test student's comprehension. Simple instructions are provided. There is no need of previous computer experience to run this program, which is completely self-contained and can be operated by the student without supervision once it has been started by the teacher.

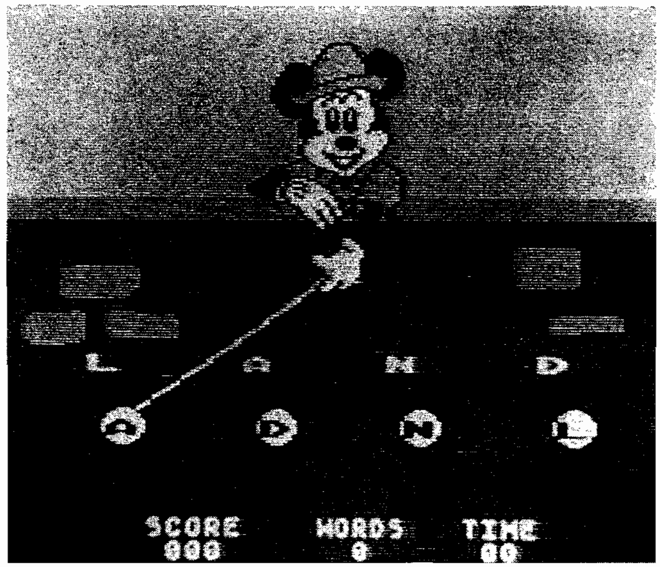
Price is **$24.95.** Contact Educomp Enterprises, 191 North 650 East, Bountiful, UT 84010.

A Neurologic Patient Simulator

Encephalon, for the *Apple II, Apple II+, Apple IIe, Franklin,* or compatible computers with 48K ROM, Applesoft, and DOS 3.3, is a neurologic Patient Simulator that uses high-resolution graphics. It allows medical students to practice neurologic examination and diagnosis on simulations constructed from findings of actual or hypothetical patients. The program allows a choice of patients, presents the history, and allows interactive simulated examination. It provides hi-res graphics, sound, and color.

Price is **$39.00** and includes disk and manuals. Available from Andent, Inc., 1000 North Ave., Waukegan, IL 60085.

Walt Disney Ships Its First Personal Computer Software Program to Atari



An actual scene from "Mickey and the Great Outdoors."

Disney made history when Mickey Mouse appeared in the first synchronized sound cartoon ever. Now Disney's making history again as Mickey enters the computer age in the first Disney microcomputer software program, **Mickey in the Great Outdoors,** designed exclusively for *Atari* home computers. The program comes on either a cassette to run with 16K of RAM or disk for 32K of RAM, and it provides literally hours of playing and learning enjoyment.

Mickey in the Great Outdoors fills the personal computer screen with high-quality animation and top-notch, full-color graphics. It also features original music and sound effects. Mickey appears in various situations called "learning adventures" in which the player actually controls his actions by using the joystick. The goal is to move Mickey along on his adventures.

This package offers two distinct learning adventures. The first, "Mickey Goes Hiking," develops and reinforces grammar and spelling skills by requiring the player to finish incomplete sentences and create words out of scrambled letters in order to move Mickey along on his adventure. To guide Mickey through his second adventure, "Mickey Goes Exploring," the player must finish incomplete equations and complete number sequences in their proper order, thus developing and reinforcing the basic math skills of equation solving and number sequencing for ages 7 to 10 years.

Contact Walt Disney Telecommunications and non-Theatrical Co., 500 South Buena Vista St., Burbank, CA 91521; (213)840-1111.

(Continued on next page)

Starter Kits from SKU

All the accessories *Atari and Commodore VIC-20* personal computer owners need are available in a new, comprehensive starter kit from SKU. The package contains two blank data cassettes from Maxell, an Intro PerfectData™ video display cleaning kit, a Discwasher cassette head cleaning kit, a Pointmaster ™ joystick, a multiple-plug outlet and $90 in rebate coupons from accessory and software vendors.

The kit is sold through mass retailers and computer specialty stores for **$44.99.** A kit is also available for Apple II and IBM PC owners for **$66.99.**

KINDER KONCEPTS

Midwest Software has announced KINDER KONCEPTS, a series of programs for the *Apple II+ and all Commodore computers except the VIC-20.* The programs deal with reading readiness, basic math concepts, perception, pattern recognition, letters, numbers, colors, and shapes. The series was developed with the cooperation of fourteen local kindergarten teachers and each program was written to fill one or more of the teachers' planned objectives for the year.

Midwest will custom build disks for customers of the Commodore versions who like to mix and match any number of programs from the available selection.

The disk versions will be menu driven and all programs follow a similar design. Ten problems are presented to the child. If the correct response is given on the first try, the child is rewarded with a smiling face and a little tune is heard. A frowning face follows an incorrect response and the correct answer is given on the third try. Each program has a built-in graph so the teacher or parent can monitor progress at a glance. All programs operate with a single keystroke and reading is kept to an absolute minimum.

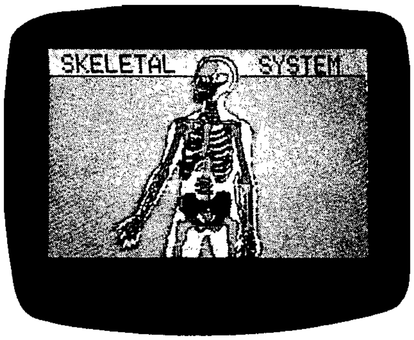
Price is **$7.95** each for the Commodore cassettes. For all ten programs on disk for Commodore or Apple II+ the price is **$69.50.** Contact Midwest Software, Box 214, Farmington, MI 48024; (313)477-0897.

MICRO

FOR COMPLETE GRAPHICS:

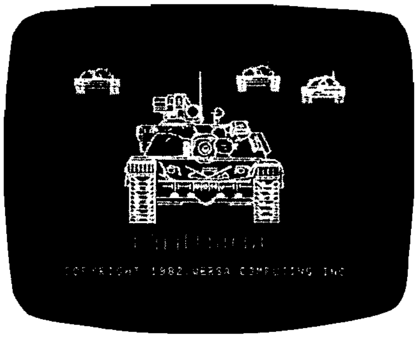
VersaWriter

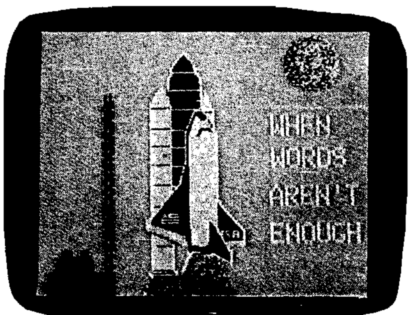
EDUCATION



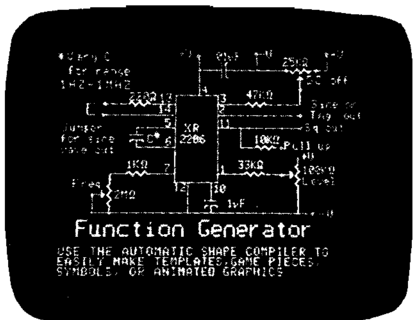
ARTIST

GAME PROGRAMMER





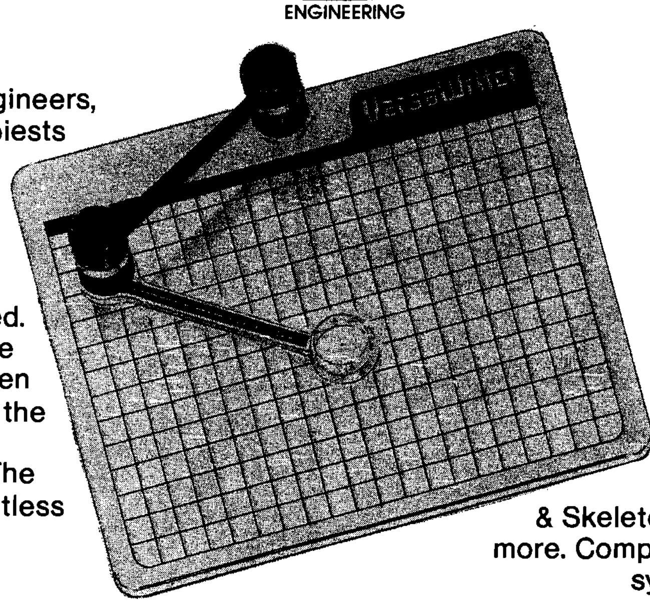
HOBBIEST



ENGINEERING

CHILDREN

Teachers, artists, engineers, programmers & hobbiests find VersaWriter an easy to use tool for creating micro computer graphics. No programming experience is required. Pictures can be made by simply tracing. Even children can explore the exciting world of computer graphics. The VersaWriter is as limitless as your imagination.



VersaWriter contains complete software for drawing with color, brushes & dots. Add text or fill in over 100 colors. Create your own shapes and place anywhere on the screen. Use Area/Distance, Move Picture, Electronic Drawing & Skeleton programs plus much more. Complete hardware/software system for Apple II/II+/IIe - **$299.00**

VERSA COMPUTING, INC.

Versa Computing Products are available at your local computer products store.

VersaWriter is also available with software designed for Atari & IBM PC.

Distributed by:

Computerland Corp.
Hayward, Calif.

Softsel Computer Products
Inglewood, Calif.

Pete & Pam Computers
Lancashire, England

Micron Distributing
Toronto, Canada

Program Spektrum
Bromma, Sweden

Micro Products Sales Group
Lynn, Mass.

Educational Media
Washington, Penn.

ESD Laboratories
Tokyo, Japan

Blue Ridge Computers
Capetown, South Africa

3541 Old Conejo Road, Suite 104 • Newbury Park, CA 91320 • (805) 498-1956

Next Month in MICRO

October Features
Programming Techniques!

- Select your programs from a directory by number
- Plot nearly any function on your computer screen
- Speed up array operations in Microsoft BASIC
- Learn how to implement linear search

Starting in October each computer will have its OWN section!

We've sectionalized MICRO for your convenience. You'll find a general section — something for everyone, plus separate sections for Commodore, Atari, Apple, and CoCo.

MICRO™

Reviews in Brief

Product Name: **General Chemistry**
Equip. req'd: Apple II
Price: $340.00 for seven disks
Manufacturer: COMPress
Van Nostrand Reinhold Co. Inc.
286 Congress St.
Boston, MA 02210

Description: *General Chemistry* is a series of seven disks that cover beginning chemistry including: the elements; inorganic nomenclature; chemical formulas and equations; atomic, formula, and molecular weights; percent composition and empirical formulas; Chemaze (a chemistry game); and ideal gases. The programs supply a lucid explanation of the basic yet sometimes confusing topics of general chemistry. It would be excellent for introductory chemistry courses as well as intermediate chemistry students.

Pluses: It is like a pithy version of a textbook since it deletes all of the trivial pieces of information that can easily confuse the student. The illustrations and sound effects keep the students from being bored. These programs could be beneficial to any chemistry student since it gives the reasoning behind every step in solving a problem.

Minuses: While clearly defining alkalis, metals, and gases, the program omits the definition of a non-metal.

Documentation: While skimpy, the programs are self teaching and require little knowledge beforehand regarding operation.

Skill level required: No particular knowledge necessary.

Reviewer: Rick Sohn

Product Name: **Master Grades**
Equip. req'd: PET, 32K with CBM disk drive
Price: $39.50
Manufacturer: Midwest Software
Box 214
Farmington, MI 48024

Description: *Master Grades* is a grades-maintenance program for classroom teachers written in compiled BASIC.

Pluses: The program is user friendly and useful. Setting up class lists is slow, but weekly updating goes quickly. It provides six print options to cover most grading needs, including progress letters for parents, and it allows the teacher to set both points needed for grades, and relative weight to give each score. The program is well protected against user errors and allows editing of names and grades. BASIC source code is available for a few dollars more.

Minuses: Since all data is held in memory at once, the total number of students per teacher is limited to 200.

Skill level required: Most teachers should be able to use this program.

Reviewer: Jim Strasma

Manufacturer's addenda: The current version of *Master Grades* keeps track of attendance as well. BASIC source code is included on the disk. Users may modify the BASIC and Midwest will recompile the user revision for a nominal fee. Works well with an ASCII printer but CBM printer gives more attractive hardcopy. 16K version available; also for Apple II, 3.3 DOS. Coming soon, CBM 64 and TRS level II versions.)

Product Name: **Bumble Games**
Equip. req'd: Apple 48K, 3.3 DOS
Price: $60.00
Manufacturer: The Learning Co.
4370 Alpine Road
Portola Valley, CA 94025

Description: *Bumble Games* is an educational game program for children that teaches the concepts of coordinate graphing using positive numbers. This is accomplished through six hierarchical games, which begin with a number line activity, progress to naming space coordinates, and finish using true coordinate graphing activities.

Pluses: Concepts are taught in a step-by-step approach using appealing graphics and pleasing sound effects. The sound effects can be turned on and off from the main menu. The last activity, *Bumble Dots,* has an option that allows construction of user-made coordinate graphing pictures.

Minuses: The user may tire of the similarity among three of the six activities.

Documentation: A well-organized pamphlet accompanies the program. In addition to this, the program has an instruction section for each activity.

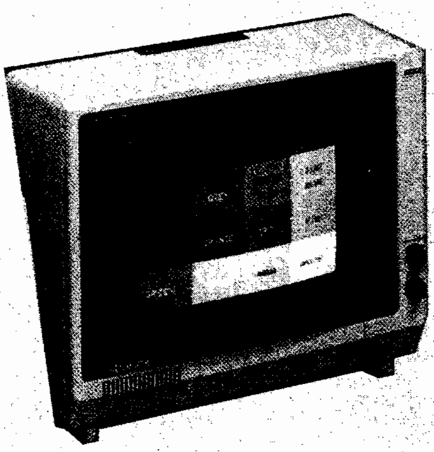
Skill level required: Children ages 4-10; no computer literacy necessary. Younger children will need assistance learning the activities.

Reviewer: Larry Ross

(Continued on page 136)

ENTER THE MICRO™ GRAPHICS

Grand Prize:



AMDEK COLOR-I MONITOR

Plus: A One Year Subscription to MICRO Magazine

4 First Prizes: AMDEK B&W Monitor

Plus: One Year Subscription to MICRO

---

4 Second Prizes: Box of Disks from Leading Edge

Plus: One Year Subscription to MICRO

---

4 Third Prizes: 1 Year Subscription to MICRO

Prizes Awarded One in Each Category: Atari, Apple, Commodore, TRS-80C

CONTEST

Use The Form Below To Enter Your Graphic Today

CONTEST RULES

1. No purchase necessary. Official Entry Forms are available in MICRO Magazine, or may be obtained, free of charge, from retail stores selling MICRO Magazine, or from MICRO Magazine directly by sending a self-addressed, stamped envelope to: MICRO Graphic Contest, P.O. Box 6502, Amherst, NH 03031. Exact copies of the Official Entry Form are also acceptable.

2. All entries must be on machine-readable format and, if possible, should be accompanied by a photograph or 35 mm slide of the monitor screen taken while the graphic is being displayed. Submitted material will not be returned unless a Self-Addressed, Stamped Envelope is provided. MICRO assumes no responsibility for lost slides or damaged material.

3. All entries must be postmarked no later than December 15, 1983.

4. Contestants may enter as many times as they choose, however each graphic submitted must be an original work and must be accompanied by its own entry form. Each graphic should be titled.

5. All prizes will be awarded. In the event contest judges are deadlocked on the winner of any prize, the prize will be awarded to the contestant whose entry has the earliest postmark. Substitutions are not allowed. All decisions of the judges are final.

6. By submitting a graphic to be judged, the contestant swears and affirms that the graphic is an original work created by the contestant and assigns all rights to reproduce the graphic, for no additional consideration, to The Computerist, Inc., and its divisions: Micro Ink, and MICRO Magazine. The contestant is liable for any and all litigation, court costs, and attorneys' fees resulting from his or her submission of plagiarized or stolen graphic programs.

7. Winners will be announced in the March 1984 issue of MICRO Magazine. A list of all winners may be obtained after March 1, 1984 by mailing your request and a self-addressed stamped envelope to: MICRO Graphic Contest, P.O. Box 6502, Amherst, NH 03031.

8. Employees of The Computerist, Inc., Micro Ink, and MICRO Magazine, as well as MICRO Magazine's columnists and contributing editors are ineligible.

9. The MICRO Graphic Contest is a contest of skill, talent, and programming ability and in no way constitutes a game of chance or lottery. Void where prohibited by law.

OFFICIAL ENTRY FORM

MICRO Graphic Contest Official Entry Form

Name ______________________

Street ______________________

City ____________ State ______ Zip ____

Phone ______________________

Age ______________________

System ______________________

Graphics Package ______________________

Title of Graphic ______________________

I have read the MICRO Graphic Contest rules and understand and have followed all regulations. ______________________

Signature

Complete and mail with your graphic to: MICRO Graphic Contest P.O. Box 6502, Amherst, NH 03031

Reviews in Brief *(continued)*

Product Names:	**Watchwords & Wordisk Maker**
Equip. req'd:	Apple II+ or Apple IIe, 48K, drive, (printer optional)
Price:	Watchwords $59.95; Workdisk Maker $29.95; (25% discount to schools)
Manufacturer:	Micromedia Software 276 Oakland Street Wellesley, MA 02181 (617) 237-5630

Description: *Watchwords* is a drill and practice program for spelling words that is set up as a climb-the-pyramid game. It uses multiple choice between alternate spellings of each word. There are nine levels, each having a list of up to 100 words. If you get enough correct answers you move up a level. The program comes with a standard set of words. *Wordisk Maker* allows you to make your own sets of words, up to 900 words per disk. Several pre-made Wordisks are also available for different skill levels.

Pluses: *Watchwords:* the teacher can set time limit, correct or incorrect spelling, sound or no sound, percentage required to climb, and choose number of words or type in words. Words appear in very large upper-case letters on the screen. It prints out student records and is extremely easy to use.

Minuses: It is unfortunate to package these programs separately — you really need both. *(Editor's note:* According to John Whitman of Micro Media Software, *Wordisk* is packaged separately because it can be used with several other software packages as well.) The present version does not allow blanks or hyphens in words. In *Wordisk Maker*, it is extremely awkward to view a wordisk list of more than 10 words on screen — you need a printer.

Documentation: Both excellent — easy to read and very complete; they even have indexes! The instructions on screen are also easy to follow.

Skill level required: *Watchwords:* age 6-adult, depends on level of words; *Wordisk Maker:* adult.

Reviewer: Mary Gasiorowski

Product Name:	**File-Fax**
Equip. req'd:	Apple II or Apple II+, minimum of one disk drive
Price:	$149.00
Manufacturer:	TMQ Software, Inc. 82 Fox Hill Drive Buffalo Grove, Illinois 60090 (312) 520-4440

Description: *File-Fax* is a new, easy-to-learn, data-base management system. Filing and sorting of data, keyed according to user-defined criteria, is a particularly attractive feature. On disk HELP screens make this package one of the user friendliest I've seen.

Pluses: This manufacturer has clearly devoted attention to eliminating problems and areas of awkwardness found in other DBM systems. Search and sort features are flexible to the extent that any string within any field on any disk drive can be located. The system is expandable to use up to eight disk drives.

Minuses: I felt the command formats were needlessly non-standard. This increases the learning time required to use the product without constantly referring to the manual.

Documentation: Quite good. The printing is easy on the eyes and the packaging is attractive. There is a thorough tutorial with vocabulary structured for the novice.

Skill level required: The least experienced of users should have no trouble working through the tutorial and then applying what he has learned to his own DBM needs.

Reviewer: Chris Williams

Product Name:	**Insta-Load**
Equip. req'd:	Apple II+, Apple IIe, or compatible, 48K RAM, and DOS 3.3
Price:	$24.95
Manufacturer:	Eden II Computing P.O. Box 959 Pebble Beach, CA 93953

Description: The central utility of this set of five is "FASTRACK," which saves Applesoft and binary programs to disk in a special format, enabling faster loading. It co-exits with standard files on the same disk. The other utilities are used to delete such files, create extremely fast-booting masters (seven to boot-up, including loading Integer BASIC into an optional RAM card), find and mark bad disk sectors, and map disk contents.

Pluses: Fast loading is accomplished without DOS modification. Speed increase is greatest with large files and binary files (binary: 2.7 × for 40 sectors, 3.5 × for 80 sectors; Applesoft: 2.3 × for 40 sectors, 2.7 × for 80 sectors). All utilities are copyable and modifiable.

Minuses: After FASTRACKing a file it cannot be copied with FID or DELETEd in the normal way; if Applesoft, RENAMEing it requires extra effort; from 1 to 255 extra bytes could be loaded with it from disk, requiring care not to overwrite any reserved memory just above program end (a rare occurence, of concern only to intermediate to advanced programmers); and other slight inconveniences.

Documentation: The 27-page manual is clear and complete, fully describing use of the utilities and detailing all restrictions.

Skill level required: Ordinary knowledge of BASIC and DOS.

Reviewer: Jon R. Voskull

(Continued on next page)

Reviews in Brief *(continued)*

Product Name:	**Discover BASIC: Problem Solving with the Apple II Computer**
Equip. req'd:	Apple II+, 32K, DOS 3.3 (printer recommended)
Price:	Teacher's Guide and Material — $74.95; Student Workbook — $5.95; Additional Demonstrations Disk — $9.95
Manufacturer:	Sterling Swift Publishing Company 7901 South IH-35 Austin, TX 78744

Author: Rick Thomas

Description: *Discover BASIC* is an extensive hands-on introduction to BASIC programming set up for a classroom. It covers major introductory topics such as PRINT, LET, INPUT, IF...THEN, GOTO, FOR...NEXT, READ/DATA, RND, low-resolution graphics, and DIM. Each topic is taught with demos, exercises, programming problems, summary, supplementary reading, and test questions. The program is based on learning by discovery.

Pluses: A well-written package designed for the classroom to teach programming. Materials provided are quite helpful for the teacher.

Minuses: Does not take much advantage of using the computer to teach the subject, though there are exercises to work out at the keyboard.

Documentation: Both the teacher's and student's manuals are easy to read and well written. The teacher's manual includes written unit objectives, supplementary activities, listings of programs, answers to the student manual, as well as a disk of demonstrations and a disk of program solution.

Skill level required: Grade 8 to adult (some formal reasoning skills needed).

Reviewer: Mary Gasiorowski

Product Name:	**Disk Library**
Equip. req'd:	Apple II
Price:	$39.95
Manufacturer:	Modular Media 11060 Paradela Street Miami, FL 33156

Description: *Disk Library* is an organizational utility for Apple disk files that creates library text files containing information on disk-based programs. Each program entry can include the program name, a user-assigned volume number (different from DOS's), the file type (Integer, Applesoft, etc.), and a user-defined program type. In addition, each library file includes a name for the disk and the number of free sectors on it. The program can handle up to 1,200 entries in each library file.

(Continued on page 140)

Reviews in Brief *(continued)*

Pluses: All types of DOS may be read by *Disk Library*, making it possible to catalog all of your Apple disks. Each library file may be sorted by name, volume, file type, or program type, with two keystrokes. Users may also generate neatly formatted hard copies of their library files.

Minuses: No operational problems encountered.

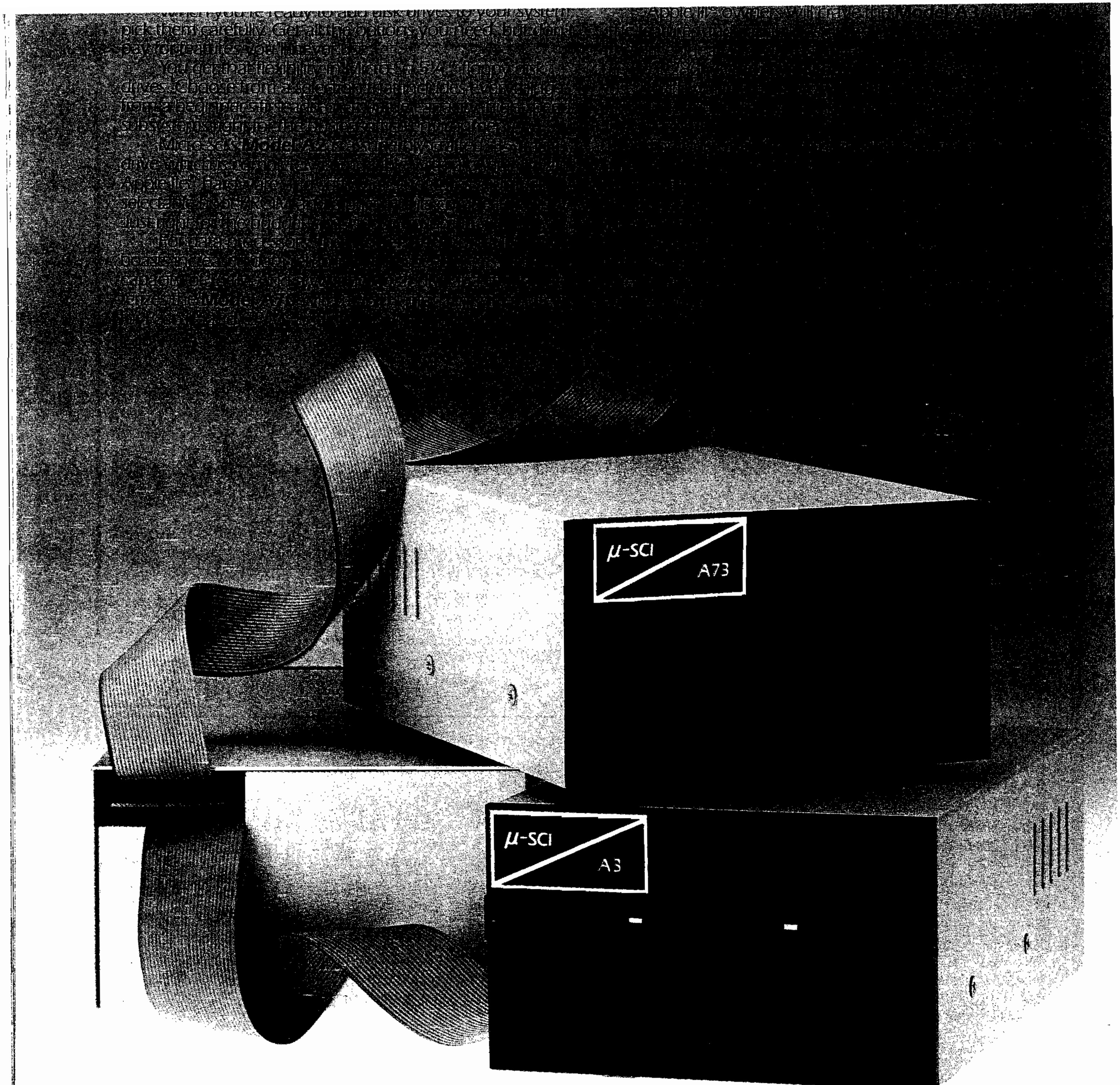
Documentation: Over 100 pages describe the operation of *Disk Library*, but a beginner can learn how to use it by booting the disk and experimenting. It handles user error very well.

Skill level required: This utility is easy to use.

Reviewer: John Hedderman

Editor's Note: In the July issue of MICRO (62:138) we published a review called "Color Diskette Repair." Computerware has informed us that the correct name is "Disk Utilities with Repair" and the price is $24.95, not $31.95 as stated.

MICRO


µ-SCI
A73
µ-SCI
A3

Advertiser's Index

MICRO INK is not responsible for claims made by its advertisers. Any complaint should be submitted directly to the advertiser. Please also send written notification to MICRO.

National Advertising Representatives

Home Office:
Bob Mackintosh, Advertising Manager
Sheila McDonough, Advertising Representative
10 Northern Boulevard
P.O. Box 6502
Amherst, NH 03031

West Coast:
The R.W. Walker Co., Inc.
Gordon Carnie
2716 Ocean Park Boulevard, Suite 1010,
Santa Monica, California 90405 (213) 450-9001

serving: Washington, Oregon, Idaho, Montana, Wyoming, Colorado, New Mexico, Arizona, Utah, Nevada, California, Alaska, and Hawaii (also British Columbia and Alberta, Canada)

Mid-West Territory:
Thomas Knorr & Associates
Thomas H. Knorr, Jr.
333 N. Michigan Avenue, Suite 403
Chicago, Illinois 60601 (312) 726-2633

serving: Ohio, Oklahoma, Arkansas, Texas, North Dakota, South Dakota, Nebraska, Kansas, Missouri, Indiana, Illinois, Iowa, Michigan, Wisconsin, and Minnesota.

NEWS...NEWS...NEWS...NEWS...NEWS...NEWS.

• • The First Annual **International Apple, IBM-PC Fair** will be held Saturday and Sunday, **September 24 and 25, 1983.** The event is co-sponsored by the Big Apple User's Group, The New York IBM Personal Computer User's Group, and The United Nations International School. The two-day show will be at the United Nations International School located at 23rd Street and East River Drive (next to Waterside) from 10:00 a.m. to 5:30 p.m.

IBM and Apple personal computer hardware and software will be on display and for sale. Seminars featuring games, computer languages, business applications, computer graphics, stock market analysis, scientific applications and data base management will be held. A Swap Room featuring software and hardware will also be open. A **children's workshop** run by children and only for children will be a highlight of the fair.

For more information contact Big Apple Users Group, P.O. Box 490, Bowling Green Station, New York, NY 10274.

• • **Computer Showcase Expo** has announced its schedule of **computer shows for the Fall.** The following shows will be held **September 22-25, 1983: New York** (New York Coliseum), **Detroit** (Cobo Hall), **Atlanta** (Atlanta Apparel Mart). The **San Francisco** show (Brooks Hall) will be held **September 29-October 2.** Contact The Interface Group, Inc., 300 First Ave., Needham, MA 02194.

• •The **Fourth Annual Conference on Classroom Applications of Computers** will be held **October 7-8, 1983,** at Independence High School, **San Jose, CA.** This conference, designed **exclusively for educators,** offers field trips, workshops, seminars, lectures, commercial exhibits, seminars, and demonstrations. For more information contact Computer-Using Educators, P.O. Box 18547, San Jose, CA 95158.

• •**Electronic Fun Expo,** New York's first state-of-the-art **consumer electronics show,** is slated for **November 3-6, 1983,** at the **New York Coliseum.** The show's sponsor, Electronic Fun Magazine, expects attendance to reach 60,000. For more information contact Electronic Fun Magazine, 350 East 81st St., New York, NY 10028.

• •Teachers from New York, New Jersey, Connecticut, and Pennsylvania are invited to demonstrate and observe how teachers use computers in their classrooms at **TC/TC, a Teachers College conference on Teaching with Computers**, Saturday, **November 19**, at Teachers College/**Columbia University.** Sixty teachers are expected to present computer-based lessons that they have developed during the one-day conference.

Teachers who would like to demonstrate a teaching-with-computers curriculum should write to Professor Mary Alice White, Box 227, Teachers College/Columbia University, New York, NY 10027. Teachers who wish to attend the conference may write or call The Office of Continuing Education, Box 132, Teachers College/Columbia University, New York, NY 10027; (212) 678-3065. Admission is $15.

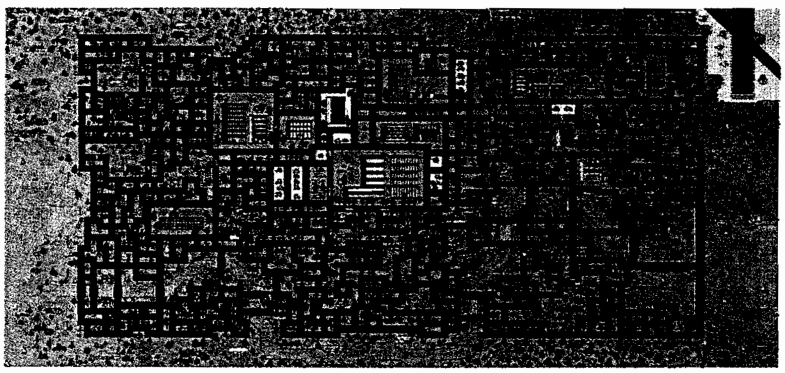
• •**Softcon,** a trade show and conference for the **software industry,** will be held **February 21-23, 1984,** at the Louisiana Superdome in **New Orleans.** Attendees will include buyers, sellers, developers, and marketers of software. For more information contact Northeast Expositions, 822 Boylston, St., Chestnut Hill, MA 02167.

• •A software company president offers these four **pointers for parents buying educational software** for their children: make sure the game meets its educational objective; check for visual and entertainment value; determine whether or not the game is easy to use and understand; decide whether or not the game will remain valuable even after it is mastered; look over the documentation and manuals — they should be playful and entertaining as well. Thanks to Bruce Zweig of Lightning Software for these tips.

• •More than **one third of all households using computers do not have children,** according to a report from TALMIS, an information and consulting service for the microcomputer industry. The report also said that VIC-20, Atari 400, and TI 99/4 users are most likely to have children; VIC-20 users have the most children and TI 99/4 users have the most children under six.

.NEWS...NEWS...NEWS...NEWS...NEWS...NEWS

You've got the loot ... now, *GETAWAY!* to a great new game from the Atari® Program Exchange!

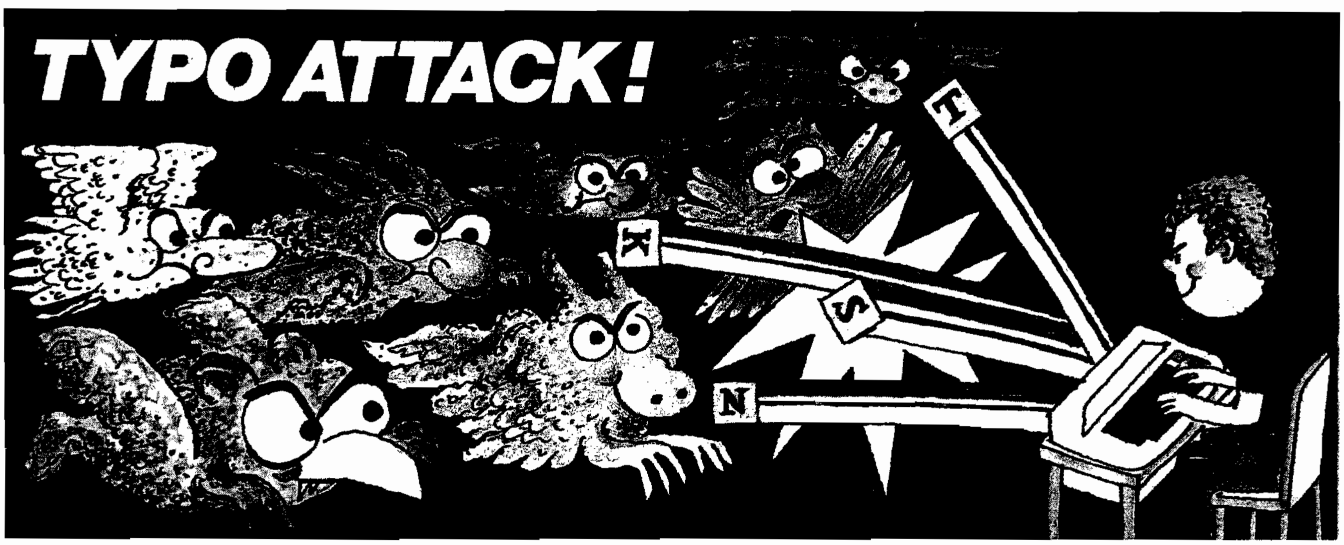


It's *all* there! The squall of sirens, the crazy turns down endless city streets, the anxious search for ill-gotten gain, the race against time for a safe place to stash your cash! Now your gas tank is nearly empty and night is about to fall. The coppers are closing in fast. Before you learn again that crime doesn't pay...Quick! ***GETAWAY!***

Ask for *GETAWAY!* at your local Atari software retailer, or order direct. Phone 800-538-1862, or 800-672-1850 in California. Or write Atari Program Exchange, P.O. Box 3705, Santa Clara, CA 95055.

Cassette (410)	APX-10195	32K	$29.95
Diskette (810)	APX-20195	32K	$29.95

For direct orders, add $2.50.



You're in for a nasty spell ... unless you stop the Typos!

In the dusky world beneath your keyboard the gruesome Typos dwell ...waiting to attack! Term paper due tomorrow? Got to get that book report typed? Fool! The Typos will devour your letters as you type! That could spell D-O-O-M-E for *you!*

Before you start typing, get down to the real work: destroy the Typos before they destroy your proase... uh, proze...prrrose...Oh NO! THE TYPOS!!! Get ***TYPO ATTACK***, a grand and glorious game from Atari® Program Exchange. It might even improve your typing!

Ask for ***TYPO ATTACK*** at your local Atari software retailer, or order direct. Phone 800-538-1862, or 800-672-1850 in California. Or write Atari Program Exchange, P.O. Box 3705, Santa Clara, CA 95055.

Cassette (410):	APX-10180	8K	$29.95
Diskette (810):	APX-20180	16K	$29.95

For direct orders, add $2.50.

TRUNKS FOR THE MEMORIES.

Introducing the most logical place to store Elephant Memory Systems® (or lesser brands of disks): The Trunk.

With its alphabetized library index, you can file or retrieve up to 60 disks, instantly.

The Trunk is made of durable molded plastic with a hinged, one-piece lid, to keep disks safe from dust, dirt, and other detriments which disks despise.

And, it's portable. Because the lid doubles as a carrying handle so your Elephant Memory Systems® disks can go anywhere you do.

There's a model for 5¼" and 8" floppies, as well as a cassette-and-game file and a special Atari® version.

So if you're looking for the best disk storage system on the market . . .

The Trunk is an open-and-shut case.

THE TRUNK. ENDORSED BY ELEPHANTS.

Elephant Memory Systems® Disks

A full line of top-quality floppies, in virtually every 5¼" and 8" model, for compatibility with virtually every computer on the market. Guaranteed to meet or exceed every industry standard, certified 100% error-free and problem-free, and to maintain its quality for at least 12 million passes (or over a life-time of heavy-duty use).

Marketed exclusively by Leading Edge Products, Information Systems and Supplied Division, 55 Providence Highway, Norwood, MA 02062. Dealers: Call toll-free 1-800-343-8413; or in Massachusetts, call collect (617) 769-8150.