can become very costly. The next type is the dot matrix printer. These are more expensive to buy than thermal printers, but since they use regular paper they may be more economical in the long run. Dot matrix printers are widely used and are the best choice for most classroom word processing applications. Their main disadvantage is they tend to be noisy. The third type is typewriter-quality printers. These produce the nicest print, but are much more expensive. When checking into printers, be sure to check the cost of the interface you will need to attach the printer to your computer.

One worry is that children have to learn to type in order to use word processing programs. We have found that with just a little practice most children prefer typing to writing with a pen or pencil. Also, several programs, such as *Typing Tutor* by Microsoft, are available to help master typing.

We do not have the space here to mention all the relevant projects, ideas and products. (Fortunately, we do all our writing on a word processor, so that when we realized we had written too much, it was easy to edit and reorganize this article to fit our space.) We have covered just a few of the many possible uses of word processing programs in education. We hope to hear from you about other innovative projects and ideas. ©

D!SK DRIVE WOES?
PRINTER INTERACTION?
MEMORY LOSS?
ERRATIC OPERATION?

Don't Blame The Software!

Power Line Spikes, Surges & Hash could be the culprit! Pat. #4,259,705
Floppies, printers, memory & processor often interact! Our patented ISOLATORS eliminate equipment interaction AND curb damaging Power Line Spikes, Surges and Hash.

- ISOLATOR (ISO-1) 3 filter isolated 3-prong sockets; integral Surge/Spike Suppression; 1875 W Maximum load, 1 KW load any socket $62.95
- ISOLATOR (ISO-2) 2 filter isolated 3-prong socket banks; (6 sockets total); integral Spike/Surge Suppression; 1875 W Max load, 1 KW either bank $62.95
- SUPER ISOLATOR (ISO-3), similar to ISO-1 except double filtering & Suppression $94.95
- ISOLATOR (ISO-4), similar to ISO-1 except unit has 6 individually filtered sockets $106.95
- SUPER ISOLATOR (ISO-11) similar to ISO-2 except double filtering & Suppression $94.95
- CIRCUIT BREAKER, any model (add-CB) .... Add $ 8.00
- CKT BRKR/SWITCH/PILOT (-CBS) .... Add $16.00

AT YOUR DEALERS

Master-Card, Visa, American Express
Order Toll Free 1-800-225-4876
(except AK, HI, PR & Canada)

ESP Electronic Specialists, Inc.
171 South Main Street, Natick, Mass. 01760
Technical & Non-800: 1-617-655-1532

SILENT MAGIC

How to Get Started with CP/M
Carl Townsend
This practical book gives you a good overview of the operating system, an idea of what it will be like to use it, and what it can do for you.
ISBN 0-918398-32-0 $13.95

Nailing Jelly to a Tree
Jerry Willis and William Danley, Jr.
This is a book about software. The emphasis is on learning to use the thousands of available programs that have already been written, and adapting them to your machine.
ISBN 0-918398-42-8 $15.95

Microsoft BASIC
Ken Knecht
Want a complete introduction and tutorial on programming in Microsoft BASIC? This book starts with the simplest and most commonly used commands and progresses on with examples that actually run.
ISBN 0-918398-23-1 $13.95

Instant BASIC, 2nd Astounding! Edition
Jerald Brown
The new edition of this best-selling programming book keeps both the flavor and style of the old edition. It provides more annotations, extra activities and additional text.
ISBN 0-918398-57-6 $12.95

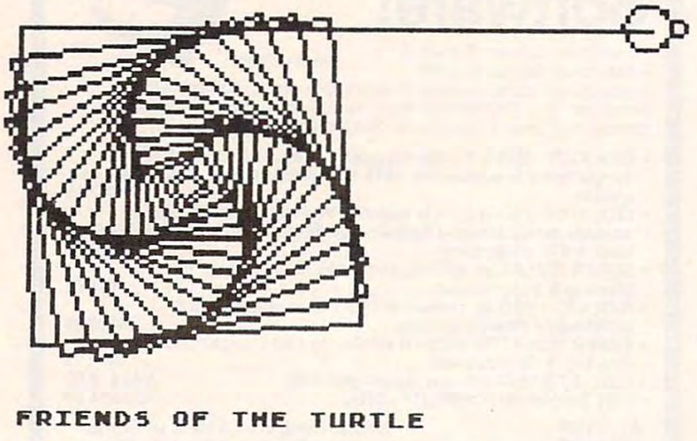
With dilithium Press

Yes, please send me the following book(s). I understand that if I am not satisfied, I may return for full refund.
☐ How to Get Started with CP/M ☐ Microsoft BASIC ☐ Instant BASIC ☐ Nailing Jelly to a Tree
Enclosed is $________ plus $1 for shipping and handling. ☐ Please send me your free catalog.
NAME________ dilithium Press
ADDRESS________ P.O. Box 606
CITY, STATE, ZIP________ Beaverton, OR 97075

dilithium Press
P.O. Box 606
Beaverton, OR 97075
800-547-1842
Call for the store nearest you that carries our books.

Friends Of The Turtle

David D. Thornburg
Los Altos, CA


FRIENDS OF THE TURTLE

Procedures And Pathways

All turtle languages incorporate at least two basic commands – one to move the turtle forward and another to make it turn. In Atari PILOT, for example, one can have the turtle draw a 40 unit square by entering the commands:

```
GR: DRAW 50
GR: TURN 90
GR: DRAW 40
GR: TURN 90
GR: DRAW 50
GR: TURN 90
GR: DRAW 40
GR: TURN 90
```

Figure 1.

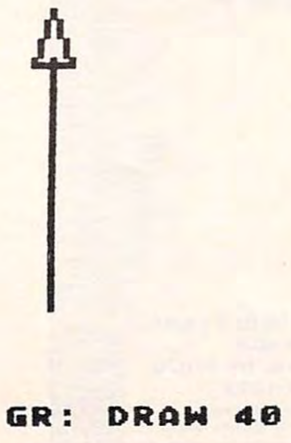

GR: DRAW 40

Figure 2.

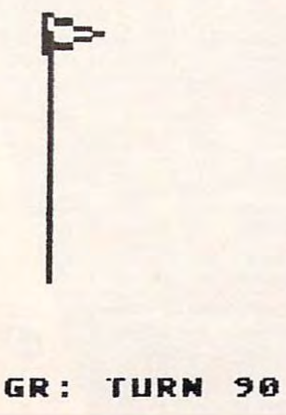

GR: TURN 90

Figure 3.

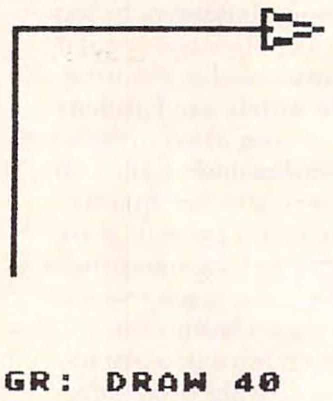

GR: DRAW 40

Figure 4.

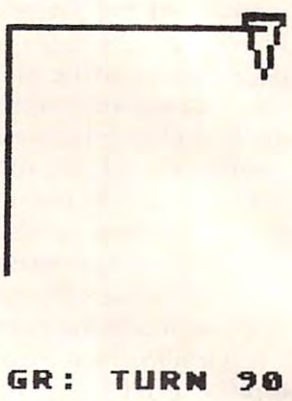

GR: TURN 90

Figure 5.

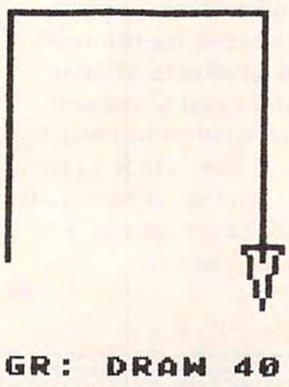

GR: DRAW 40

Figure 6.

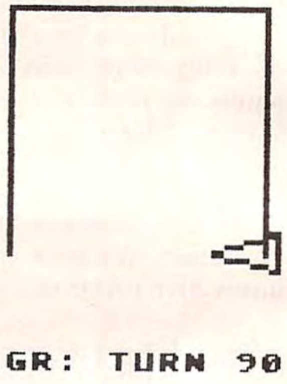

GR: TURN 90

Figure 7.

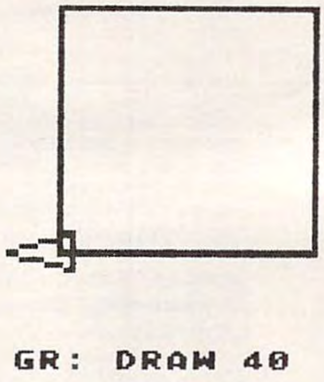
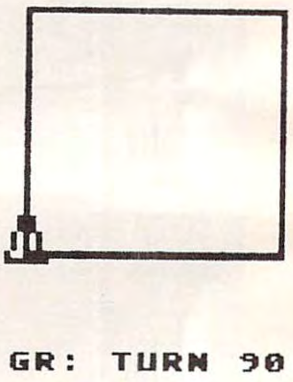

GR: DRAW 40

Figure 8.


GR: TURN 90

If you want lots of these squares, most turtle environments will let you create a *procedure* which can be used anytime you want to draw this figure. In our case (using Atari PILOT), the procedure starts with a name (for example, *SQUARE). Next, the commands shown above are entered, and finally the *end* command is entered. In PILOT this last command is simply E:.

Once a procedure is defined, it can be used to create copies of squares at any screen location,

COLLEGE BOARD SAT* PREP SERIES

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

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

S.A.T., P.S.A.T., N.M.S.Q.T. — Educator Edition set includes 25 programs covering Vocabulary, Word Relationships, Reading Comprehension, Sentence Completion, and Mathematics. **Independent tests of S.A.T. series performance show a mean total increase of 70 points in students' scores.** Price **$229.95**

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

COMPETENCY PROFICIENCY EXAM PREP SERIES

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

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

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

M.I.T. Logo for Apple $179.95

(Includes **Alice In Logoland**)

Odyssey In Time

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

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

Time Traveler

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

☆ NEW ☆
☆MICRO-DEUTSCH☆

Micro-Deutsch set includes 24 grammar lessons, covering all material of an introductory German course. Four test units also included. Grammar lessons use substitution transformation drills, item ordering, translations and verb drills. Drill vocabulary based on frequency lists. Suitable for use with any high school or college textbook. Extensively field tested at SUNY Stony Brook. Available for Apple II and PET/CBM. (PET version includes a special foreign language character chip.) Also available soon: MICRO-FRANCAIS, MICRO-ESPAÑOL, MICRO-IVRIT, MICRO-YIDDISH, MICRO-CHINESE, MICRO-JAPANESE. **$179.95**

Isaac Newton +

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

F.G. Newton

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

☆ NEW ☆
☆Pythagoras and The Dragon☆

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

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

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

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

N.Y.S. residents add sales tax.

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

KRELL SOFTWARE CORP.

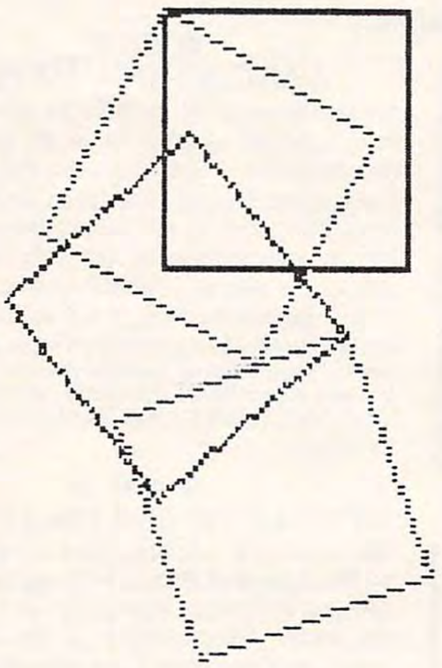
"The State of the Art in Educational Computing"

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

orientation, or color you may desire. In our case, one simply uses the procedure with the *use* command; e.g., U: *SQUARE. In this manner, procedures let you extend the number of things the turtle can "understand". To see how handy this is, look at the following program which draws several squares:

```
GR: PEN YELLOW
GR: GOTO -30,0
U: *SQUARE
GR: PEN BLUE
GR: TURN 30
U: *SQUARE
GR: GOTO 20,30
GR: TURN 40
U: *SQUARE
GR: PEN RED
GR: TURN 70
U: *SQUARE
```

Figure 9.



While this isn't a particularly pretty picture, it does illustrate how to use procedures to save a lot of typing! Procedures also make programs easier to read.

An even greater value of procedures is the freedom they give you while you are writing a program. As you think about what you want your program to do, you can write the program in outline form, with procedure names being used for those activities you haven't fully defined. Next, you can create each procedure and test it out independently of the others to make sure it works. In this way you can make steady progress from the outline to the final program without having to deal with massive numbers of statements at a time. I tend to keep procedures short and sweet – and to use lots of them.

The next topic for this month is the idea of a closed *pathway*. Closed turtle paths have some interesting properties. If you look at the figures shown above for the square, you might think that we were done when we drew the fourth side (Figure 7). If you think about it some more, you will see that the turtle is back at the place where it started, but that it hasn't returned to its original orientation. Closed turtle pathways have the property that the turtle returns to its original location and orientation at the end of the trip. This is a very important point to remember.

Now that we have defined a pathway, let's look at a simple way to create some special closed paths in Atari PILOT. One type of closed path creates geometric shapes called regular polygons. A regular polygon is a closed figure which is made from equal length sides and equal turning angles. While we could repeat our DRAW and TURN commands for each side and angle, this would make our procedures very long and tedious to type out. Fortunately, Atari PILOT allows some shorthand to make this task easier. For example, the command:

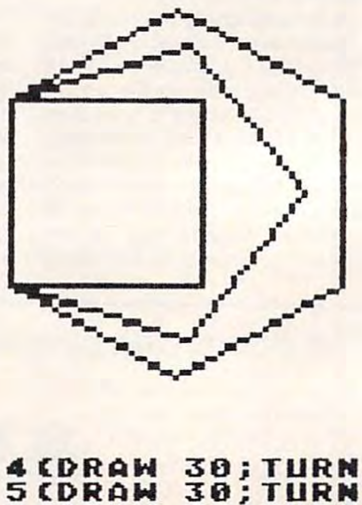
```
GR: 4(DRAW 30; TURN 90)
```

will draw a square on the display screen. The command says, in effect, "Repeat, four times, the commands DRAW 30 and TURN 90".

Using this shorthand, we can create several polygons to study.

```
GR: 4(DRAW 30; TURN 90)
GR: 5(DRAW 30; TURN 72)
GR: 6(DRAW 30; TURN 60)
```

Figure 10.



```
GR:  4(DRAW 30;TURN 90)
GR:  5(DRAW 30;TURN 72)
GR:  6(DRAW 30;TURN 60)
```

We have created three closed paths – a square, a pentagon, and a hexagon. If you look at the commands which created these figures, you will notice that the only thing that changed was the number of sides and angles, and the amount that was turned each time. If you are really on your toes, you might have noticed that the total amount turned for each figure was the same: 4x90 = 360, 5x72 = 360, and 6x60 = 360. The total amount of turning for simple closed paths is 360 degrees, regardless of the number of sides on the polygon. This is called the Turtle Total Trip Theorem, and it is a beautiful unifying concept that makes turtle geometry quite valuable.

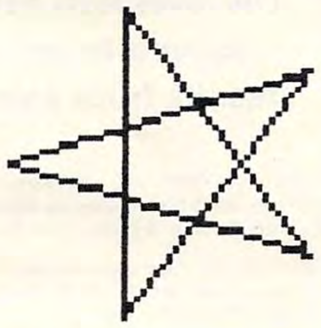
If you would like some challenges until next time, think about these two problems.

1. Can you use the Turtle Total Trip Theorem to help you make a figure which looks like a circle?

2. Look at the picture which results from this command:

GR 5(DRAW 50; TURN 144)

Figure 11.



How much total turning did this figure require? Why?

Until next time, keep those turtles moving, and send me ideas, pictures, programs, and anything else you want to share with your fellow members. Friends of the Turtle chapters should be started in your home town. Let me know what you are doing.

Resource List

Turtle graphics is increasing in popularity both as an educational and as an artistic tool. From time to time, we will publish updates of books, languages, and organizations which incorporate and/or describe turtle geometry. As you look at this list, you might find that I have left some important references out – please let me know what is missing! In the meantime, here is a beginning list to get us started.

Books:

Mindstorms: Children, Computers, and Powerful Ideas by Seymour Papert (Basic Books, 1980).

Turtle Geometry: The Computer as a Medium for Exploring Mathematics, by Harold Abelson and Andrea diSessa (MIT Press, 1981).

Computer Languages and Products:

Big Trak (programmable robot vehicle from Milton Bradley)

Atari PILOT (language cartridge for Atari 400 and 800 from Atari)

TI LOGO (language cartridge for the TI 99/4 and 99/4A from Texas Instruments)

WSFN (language disk or tape for the Atari 400 and 800 from Atari Program Exchange)

WSFN (language tape for the Commodore PET from Peninsula School Computer Project, Peninsula Way, Menlo Park, CA 94025)

Organizations:

Young People's LOGO Association
1208 Hillsdale Drive
Richardson, TX 74081

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

©

Mathematics, Basic Skills

Paper Exercises in Arithmetic

For use with
PET/CBM Computer & Printer
plus
Compatible Disk System

The Teacher's Aide

Computer programs designed for use by the classroom teacher as a primary source of exercises in mathematics, basic skills. Through simple question and answer, and with the use of only one computer system, a teacher may satisfy all individualized, in-class and homework requirements for drill in arithmetic. Students work directly upon exercise sheets. Difficulty level is easily adjustable. Answers are always provided. 23 programs included, covering integers, decimals, fractions, percent and much more.

On Disk $99.99

Algebra

Explicitly Produced Exercises in Algebra

Sixteen programs in linear and fractional equations, simultaneous equations, quadratics, signed and complex number arithmetic.

On Disk $99.99

(Arizona residents please add 4% sales tax.)
Please add $1.50 for postage and handling.

T'Aide Software Company
P.O. Box 65
El Mirage, Arizona 85335

Sample diskette, one program, each group:
$14.95 postpaid, Cost Deductible

Large Alphabet For The VIC

Doug Ferguson
Elida, OH

There are many exciting applications for the 64 programmable characters on the VIC-20. David Malmberg's article in the first issue of *Home and Educational COMPUTING!* explains fully how the VIC can generate programmable characters merely by changing the contents of memory location 36869, and by redefining the 64 eight-pixel tall characters beginning at 7168.

Another interesting memory location in the VIC is nearby: 36867. Changing its contents creates double-sized characters. By POKEing a 47 into 36867, the bottom border of the screen drops out of sight and vertically-paired characters occupy "stretched" screen locations. After clearing the screen, type an *A* and get $\overset{B}{C}$. Actually, the VIC's first character is the "@" (screen POKE 0) which yields $\overset{@}{A}$. Continue to type the alphabet and see how the stacked letters follow a pattern. To return to normal, POKE 36867,46 or hit the RESTORE and RUN/STOP keys simultaneously.

I set about to combine these two ideas so that I could get a large alphabet. I painstakingly reprogrammed the B to look like the top of a stretched "A" and the C to look like its matching bottom half. Continuing on for nearly two hours, I made it to the "O" and gave up for the night.

Somehow, the clear light of day the next morning directed me toward a much simpler approach: if the characters already reside in ROM, just read each eighth of a character *twice* into the RAM space for programmable characters to program two letters at a time!

Clearly, only 32 such stretched characters can be made since only 64 unstretched characters can be readily programmed. The space key and all the numerals fall in the wrong half of the 64, but all 26 letters of the alphabet can be stretched with the following, surprisingly short, program:

```
10 POKE 56,28: REM RELOCATE END-OF-MEMORY
   POINTER
20 CH=32776: REM LOCATION OF ALPHABET
   IN ROM
30 FOR X=7184 TO 7600 STEP 2: REM ALPHABET
   IN RAM
40 POKE X, PEEK(CH): POKE X+1, PEEK(CH):
   REM STRETCH
50 CH=CH+1: NEXT X: REM LOOP
60 POKE 36879,25: REM NO MORE BORDER
70 POKE 36869,255: REM PROGRAMMABLE
   CHARACTERS
80 POKE 36867,47: REM STRETCHED CHARACTERS
90 PRINT "(clear)ABCDEFGHIJKLMNOPQRSTUV
   WXYZ": END
```

Lines 20 through 50 read the normal alphabet (8x8 pixels) out of ROM and into RAM. Since RAM is also where a longer program will do its work, line 10 tells the computer not to go beyond 7134 (28 times 256). Line 60 is for the purist who notices the lack of a bottom border with the "normal" screen.

Simple? Certainly. The biggest drawback is the lack of numerals and spaces. In string variables with spaces, e.g., A$="HELLO THERE", the space can be replaced by the symbol for cursor-right.

The applications of this large alphabet program are left to the reader. Although it is obvious that any characters can be programmed for stretching, only the alphabet (and a few insignificant symbols) can be programmed in a way that an exact keyboard-to-character correspondence can be realized.

I would appreciate hearing from anyone who can expand on this or who has a clever application. ©

LEARN

Discover how easy it is for you to get useful results from your VIC.

Understanding Your VIC Volume 1: Basic programming uses a proven step-by-step approach to teach programming. It costs $11.95

A cassette tape with two demonstration programs from the book is available for $7.95. It will save you typing time and eliminate typing errors.

VIC Software
Easy to use program for you to create new characters for graphics or games. Cassette $9.95.

See your dealer or order direct
VISA/MC accepted
Money back guarantee
Please add $2 ($8 overseas)
for shipping and handling

TIS INC.

Total Information Services, Inc.
Box 921, Dept. H
Los Alamos, NM 87544

Dealer inquiries invited.

SOFTWARE · HARDWARE

FOR THE VIC 20®

COMMUNICATIONS AND COMPUTER TOOLS

- **UMI RS232 COMMUNICATOR INTERFACE** **$49.95**
  Our RS-232 Communicator Interface enables the VIC-20 to talk to most RS232 devices.
- **VICTERM A** **$19.95**
  With the VICTERM A program, begin telephone communications.
- **UMI 3K RAM CARTRIDGE** **$39.95**
  The easiest way to have 6655 bytes of program memory in your VIC.
- **UMI 3K RAM EXPANDER** **$79.95**
  For programmers, ROM developers and advanced users, our 3K RAM EXPANDER adds two ROM slots and 3K of RAM memory to the Commodore VIC-20 to give a total of 6655 bytes of available user memory. An 8-position dipswitch provides switch - selectable base address and single-socket enable/disable to two ROM sockets.
- **UMI 8K RAM CARTRIDGE** **$89.95**
  Our 8K RAM CARTRIDGE adds 8K of user memory to the VIC-20.
- **UMI SOCKETED 8K RAM/ROM BOARD** **$29.95**
  Our 8K board has four sockets for mixing and matching RAMS, ROMS, or EPROMS.
- **BUTI** (BASIC PROGRAMMER'S UTILITY ROM) **$34.95**
  A beauty of a ROM that plugs into CVH-0002, our 3K RAM EXPANDER, or CVU-0006, our RAM/ROM/EPROM Board.
- **UMI VIC-20 REFERENCE CARD** **$4.95**
  Our 8-1/2" by 11" laminated reference card is packed with commonly needed Commodore VIC20 information.

PROGRAMS FOR THE COMMODORE PET®

- **SATELLITES AND METEORITES** **$49.95**
  Satellites and Meterorites is a greater challenge than most space pilots have met.
- **SKYMATH** **$12.95**
  Decimal addition and subtraction with super PET graphics!
- **MATH SERIES DISKETTE I** **$29.95**
  Pedagogically correct programs SKYMATH, SPACEDIV, and LONGDIV, on a 5-1/4" Diskette for Commodore Disk Drives.
- **SPACEDIV** **$12.95**
  Division practice with graphics that hold the child's interest.
- **LONGDIV** **$14.95**
  20 problems of step-by-step long division with super graphics and 1-inch high digits.
- **SUPERHANGMAN** **$16.95**
  Superior graphic version of the popular spelling game.

GAMES AND ENTERTAINMENT

- **KIDDIE PAK I** **NOW $39.95**
  The KIDDIE PAK I includes the following games:
  KIDDIE CHECKERS **(SEPARATE PRICE $7.95)**
  Checkers for very small children.
  SHAPE MATCHER **(SEPARATE PRICE $11.95)**
  Fun for very small children, matching hidden patterns.
  DOGGIE MAZE **(SEPARATE PRICE $11.95)**
  A dog's race against time to reach the hydrant.
  FUN TENSES **(SEPARATE PRICE $11.95)**
  Nonsense sentences which aid children.
- **SUPER FOUR I** **NOW $49.95**
  SUPER FOUR I includes the following games:
  3-D MAZE **(SEPARATE PRICE $14.95)**
  Choose a 4 to 9 level maze, then find your way down the hallways.
  BREAKOUT **(SEPARATE PRICE $14.95)**
  No program library is complete without *BREAKOUT!*
  CAROM **(SEPARATE PRICE $14.95)**
  Planning and skillful play with deflector zaps the target.
  RACEWAY **(SEPARATE PRICE $14.95)**
  Beat the competition with a high-speed race car.
- **FABULOUS FOUR I** **NOW $59.95**
  Fabulous Four I contains the following games:
  STAR WARS **(SEPARATE PRICE $16.95)**
  Blast the tie fighter!
  DRAGON MAZE **(SEPARATE PRICE $16.95)**
  Lurking man eating dragon.
  LASER WAR **(SEPARATE PRICE $18.95)**
  Blast away space debris and aliens.
  INVADER FALL **(SEPARATE PRICE $21.95)**
  Alien paratroopers descend from space. (Requires 3K Expn.)
- **SUPER STARS I** **NOW $69.95**
  SUPER STARS I includes the following games:
  THE ALIEN **(SEPARATE PRICE $24.95)**
  Fast work with inflatable alien traps. (Requires 3K Expn.)
  AMOK **(SEPARATE PRICE $18.95)**
  Killer robots with one instruction - get the intruder.
  ALIEN BLITZ **(SEPARATE PRICE $24.95)**
  Split second reactions at the 9th level of ALIEN BLITZ will find how good you are.
  GLOBBLER **(SEPARATE PRICE $24.95)**
  The Conqueror Worm gobbles globs under your control.

ORDER NOW WHILE SUPPLIES LAST!

EDUCATION

- **SUPER ADDITION, SUBTRACTION, DON'T FALL** **$14.95**
  One cassette contains all three educational games.
- **SKYMATH** **$14.95**
  With excellent graphics, SKYMATH shoots 5-digit, 3-place addition and subtraction problems into screen's "sky" in 1-inch high digits.
- **SPACEDIV** **$14.95**
  Division practice with graphics that hold the child's interest.

GAME PROGRAM CARTRIDGES NOW AVAILABLE—ORDER NOW!

- **DEALER INQUIRIES INVITED**
- **ATTENTION SOFTWARE DEVELOPERS**
  Please contact UMI for distribution and top royalties. **Blank Cartridge kits available in quantity.**
- **MASTERCARD/VISA Accepted**

united microware industries, inc.

• 3431 H POMONA BLVD. • POMONA, CA 91768 • PHONE (714) 594-1351 •

VIC 20 and PET are Registered Trademarks of Commodore Business Machines

Clip Coupon for FREE CATALOG

united microware industries inc.
3431 H Pomona Blvd
Pomona, CA 91768

Please send me my FREE CATALOG describing your Hardware and Software Products

NAME ______________________

ADDRESS ______________________

CITY ______________ STATE ________

ZIP __________ COMPANY ______________

Concentration

Charles Brannon
Editorial Assistant

One application of a user-definable character set is high-resolution, five-color games in GRAPHICS modes one and two. For example, the invaders in Atari's Space Invaders game are GRAPHICS 1 characters. The illusion of smooth motion is performed with the aid of a special feature of the Atari, horizontal fine scrolling. Although my game is less ambitious, it shows what you can do with minimum effort – I spent no more than three hours programming – from the design to the finished game.

The game is based on the card game "Concentration." Two decks of cards are thoroughly shuffled together, then laid out in a matrix of 8 by 13 cards. Each player takes his turn by turning over two cards. If they match, they are removed from the set and this "point" is credited to the player. If not, they are flipped back over. The game continues until all the cards have been matched and removed.

The Atari Version Is Slightly Different

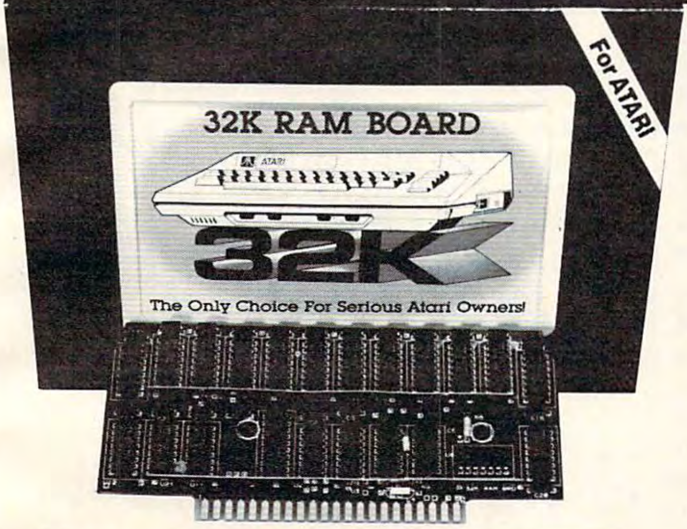
The Atari version of the game is rather different, but the idea is similar. Nineteen different graphics symbols (people, sailboats, "happy faces," cars, etc.) are randomly hidden in a 16 by 20 array. When the game is run, the computer draws the "board," a solid green rectangle. It then flashes the prompt "START/SELECT" at the bottom of the screen. Press [SELECT] to change the number of players, and [START] to begin play. A solid red cursor is placed at the top left corner of the board. Move the cursor with joystick #1 (everyone uses the same joystick). When you wish to "flip" a card, press the red button. Then try to match the revealed symbol by selecting another. If successful, your score is increased by one. The play then passes to the next player. Since the array is 16 by 20 elements, (a total of 320) there could be as many as 160 matches. Unlike the card game version, there are multiple pairs of each symbol. This could make for a very long game, so, instead, the first player to get ten matches wins. SuperFont (**COMPUTE!** #20) could be used to design other gaming characters.

```
100 REM |    Concentration    |
110 REM
120 REM (C) 1981 Small Systems
           Services, Inc.
130 REM Charles Brannon 12/03/81
140 REM
150 GOSUB 740
160 GRAPHICS 1+16:POKE 756,BASE
170 POKE (PEEK(560)+256*PEEK(561)+3),7+6
4
180 SETCOLOR 2,0,10:SETCOLOR 4,6,0:SETCO
LOR 1,12,6
190 IF T=0 THEN DIM A(16,20),CH$(20),SC(
4),PROMPT$(24)
200 FOR I=1 TO 4:SC(I)=0:NEXT I
210 CH$=")*+,-./:;<=>?@[\]^_"
220 COLOR 1
230 PROMPT$="|START |SELECTSTART |SELECT
|"
240 FOR Y=1 TO 20:FOR X=1 TO 16:A(X,Y)=I
NT(19*RND(0)+1):PLOT X+1,Y+2:NEXT X:NEXT
 Y
250 POSITION 3,0:? #6;"|concentration|"
260 NP=1:POSITION 2,2:? #6;"ABCDEFGHIJKL
MNOP":FOR I=1 TO 20:COLOR 224+I:PLOT 1,I
+2:NEXT I
270 POSITION 5,1:? #6;"|PLAYERS| ";NP:PO
KE 53279,8:POKE 20,26:K=0
280 IF PEEK(20)>25 THEN POSITION 4,23:?
#6;PROMPT$(1+K*12,12+K*12):POKE 20,0:K=1
-K
290 IF T THEN 310
300 T=PEEK(53279):IF T=7 THEN T=0:GOTO 2
80
310 IF PEEK(53279)=T THEN 310
320 IF T=5 THEN NP=NP*(NP<4)+1:T=0:P=T:G
OTO 270
330 IF T<>6 THEN 300
340 POSITION 4,23:? #6;"            "
350 REM MAIN LOOP
360 P=P*(P<NP)+1:POSITION 2,1:? #6;"|PLA
YER| ";P;" score ";SC(P)
370 GOSUB 610:X1=X:Y1=Y:V1=V
380 GOSUB 610:IF V=V1 THEN 450
390 SOUND 0,20,2,8:SOUND 1,100,12,8:FOR
W=1 TO 50:NEXT W:SOUND 0,0,0,0:SOUND 1,0
,0,0:POSITION 5,23:? #6;"|PRESS FIRE|"
400 IF STRIG(0)=1 THEN 400
410 IF STRIG(0)=0 THEN 410
420 POSITION 5,23:? #6;"            "
430 COLOR 1:PLOT X+1,Y+2:PLOT X1+1,Y1+2:
SOUND 0,12,12,8:FOR W=1 TO 20:NEXT W
440 SOUND 0,0,0,0:GOTO 360
450 FOR I=1 TO 15 STEP 0.4:SOUND 0,I*17,
12,I:SOUND 1,I*17,12,I:NEXT I:SOUND 0,0,
0,0:SOUND 1,0,0,0
460 SC(P)=SC(P)+1:POSITION 17,1:? #6;SC(
P):FOR I=1 TO 10:POKE 709,PEEK(53770):PO
KE 53279,0
470 FOR W=1 TO 10:NEXT W:NEXT I:POKE 709
,198:IF SC(P)=10 THEN 520
```

,198:IF SC(P)=10 THEN 520

THE MOSAIC 32K RAM FOR ATARI

THE BEST:



✓ **DESIGN**
✓ **MATERIALS**
✓ **SAFETY**
✓ **GUARANTEE**

THE ONLY CHOICE

You own the best micro computer available, the Atari*. At Mosaic we've made Atari computers our only business and have made our products the best anywhere. You've seen the advantage of having a 32K RAM board. The Mosaic 32K RAM is the only board designed to meet your needs now and in the future too. It has designed-in flexibility to be compatible with products available in the near future. See for yourself, Mosaic is the only choice for the serious Atari owner.

THE BEST SCREEN CLARITY

Here's what A.N.A.L.O.G. magazine had to say: "Mosaic uses, what we feel are the highest quality components, which may improve reliability." and "The Mosaic showed no sign of interference and gave the best screen clarity."

THE MOSAIC ADVANTAGE	MOSAIC 32K RAM	OTHER 32K RAMS
Works in both Atari 400 & 800	■	■
Gold edge connectors for better reliability	■	■
Fits Atari 400 without modification	■	■
Custom components for better performance & reliability	■	
Highest quality components for the best screen clarity	■	
Full year warranty	■	
Designed to take advantage of Atari 800's superior bus structure.	■	
Can be used with 8K, 16K and future products.	■	
Allows Atari 800 to have 2 slots for future expansion	■	
Designed so there's no danger of damaging your computer	■	
Designed for inter-board communication in Atari 800	■	
Easy to follow instructions for simpler no-solder installation in Atari 400	■	
Available companion board ($5) to allow running 32K board independent of other boards	■	
Full flexible memory configuration	■	

Atari 800 Memory Configuration	with the MOSAIC 32K RAM	with other 32K Boards
Empty / 32K / 16K	48K RAM 40K With BASIC Cartridge	48K RAM 40K With BASIC Cartridge
Empty / 32K / 8K	40K RAM	**Danger!** This Configuration Can Damage Computer
Empty / 16K / 32K	48K RAM 40K With BASIC Cartridge	**Danger!** This Configuration Can Damage Computer
Empty / 8K / 32K	40K RAM	**Danger!** This Configuration Can Damage Computer
8K / 32K / 8K	48K RAM 40K With BASIC Cartridge	**Danger!** This Configuration Can Damage Computer
Empty / Companion / 32K	32K RAM	**Danger!** This Configuration Can Damage Computer

Now from your nearest Mosaic dealer **$179.95**

Direct from Mosaic $189.95

MOSAIC ELECTRONICS

P.O. Box 748 Oregon City, Oregon 97045 503/655-9574

*Trade Mark of Atari, Inc.

```
480 POSITION 5,23:? #6;"|PRESS FIRE!"
490 IF STRIG(0)=1 THEN 490
500 IF STRIG(0)=0 THEN 500
510 POSITION 5,23:? #6;"             ":GOTO
 360
520 POSITION 0,2:? #6;"Player number ";P
;" |wins|":POKE 53279,8
530 FOR I=0 TO 15 STEP 0 4:SETCOLOR 4,I,
6+INT(4*RND(0)):SOUND 0 10+5*RND(0),10,4
:SOUND 1,50+10*I,12,8:NEXT I
540 SOUND 0.0,0,0:SOUND 1,0,0,0:SETCOLOR
 4,6,0
550 T=PEEK(53279):IF T=7 THEN 550
560 IF T<>3 THEN 160
570 FOR X=1 TO 16:FOR Y=1 TO 20
580 LOCATE X+1,Y+2,Z:IF Z<>1 THEN COLOR
Z-128:PLOT X+1,Y+2:GOTO 600
590 COLOR ASC(CH$(A(X,Y)))+128:PLOT X+1,
Y+2
600 NEXT Y:NEXT X:GOTO 550
610 X=1:Y=1
620 LOCATE X+1,Y+2,Z:COLOR Z+32-160*(Z>1
):PLOT X+1,Y+2:TX=X:TY=Y
630 ST=STICK(0):TR=STRIG(0):IF TR=0 AND
Z=1 THEN 720
640 IF PEEK(53279)<7 THEN COLOR Z:PLOT X
+1,Y+2:GOTO 550
650 IF ST=15 THEN 630
660 T=INT(100*RND(0)+50):SOUND 0,T,10,8:
SOUND 1,T+20,10,8
670 IF ST=14 OR ST=10 OR ST=6 THEN Y=Y-1
:IF Y<1 THEN Y=20
680 IF ST=9 OR ST=5 OR ST=13 THEN Y=Y+1:
IF Y>20 THEN Y=1
690 IF ST>8 AND ST<12 THEN X=X-1:IF X<1
THEN X=16
700 IF ST>4 AND ST<8 THEN X=X+1:IF X>16
THEN X=1
710 COLOR Z:PLOT TX+1,TY+2:SOUND 0,0,0,0
:SOUND 1,0,0,0:GOTO 620
720 FOR I=1 TO 7:COLOR 1+I:PLOT X+1,Y+2:
SOUND 0,100+I*10,12,8:FOR W=1 TO 20:NEXT
 W:NEXT I:SOUND 0,0,0,0
730 V=A(X,Y):COLOR ASC(CH$(V,V))+128:PLO
T X+1,Y+2:RETURN
740 REM INITIALIZE CHARACTER SET
750 BASE=PEEK(106)-8:CHSET=BASE*256
760 GRAPHICS 2+16:POSITION 3,4:? #6;"|C|
On|cE|Nt|rA|Ti|oN|"
770 POSITION 2,6:? #6;"Patience Please"
780 FOR I=CHSET TO CHSET+127:READ A:POKE
 I,A:POKE 712,A:SOUND 0,A,10,8:NEXT I
790 FOR I=CHSET+26*8 TO CHSET+32*8+7:REA
D A:POKE I,A:POKE 712,A:SOUND 0,A,10,8:N
EXT I
800 FOR I=CHSET+59*8 TO CHSET+63*8+7:REA
D A:POKE I,A:POKE 712,A:SOUND 0,A,10,8:N
EXT I
810 FOR I=128 TO 207:A=PEEK(57344+I):POK
E CHSET+I,A:POKE 712,A:SOUND 0,A,10,8:NE
XT I
820 FOR I=264 TO 471:A=PEEK(57344+I):POK
E CHSET+I,A:POKE 712,A:SOUND 0,A,10,8:NE
XT I
830 SOUND 0,0,0,0:RETURN
840 DATA 0,0,0,0,0,0,0,0
850 DATA 255,255,255,255,255,255,255,255

860 DATA 0,255,255,255,255,255,255,255
870 DATA 0,0,255,255,255,255,255,255
880 DATA 0,0,0,255,255,255,255,255
890 DATA 0,0,0,0,255,255,255,255
900 DATA 0,0,0,0,0,255,255,255
910 DATA 0,0,0,0,0,0,255,255
920 DATA 0,0,0,0,0,0,0,255
930 DATA 24,24,19,124,88,24,20,54
940 DATA 0,0,0,28,20,127,34,0
950 DATA 129,66,60,36,36,60,66,129
960 DATA 0,0,0,96,95,101,5,0
970 DATA 0,16,40,68,254,124,0,0
980 DATA 0,102,102,0,129,66,60,0
990 DATA 0,56,124,84,124,56,40,68
1000 DATA 0,0,68,34,63,34,68,0
1010 DATA 0,195,102,60,126,36,0,0
1020 DATA 66,255,102,90,90,102,255,66
1030 DATA 16,16,16,56,56,56,124,254
1040 DATA 0,0,56,68,84,68,56,0
1050 DATA 0,16,40,68,254,68,40,16
1060 DATA 0,170,108,198,16,198,108,170
1070 DATA 170,85,170,85,170,85,170,85
1080 DATA 16,56,124,254,84,16,16,124
1090 DATA 14,8,12,8,8,56,120,48
1100 DATA 0,255,0,255,0,255,0,255
1110 DATA 24,56,120,8,8,136,127,62
```

©

Scotch® Diskettes

Rely on Scotch® diskettes to keep your valuable data safe. Dependable Scotch diskettes are tested and guaranteed error-free. The low abrasivity saves your read/write heads. They're compatible with most diskette drives.

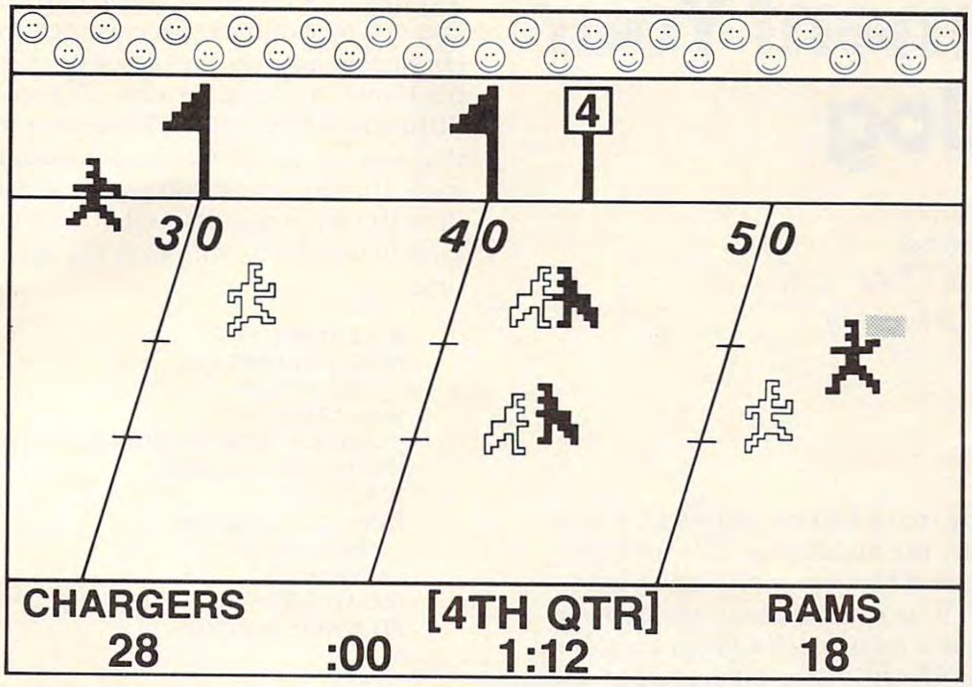
(800)235-4137

VISA

Dealer Inquiries
Invited

PACIFIC EXCHANGES
100 Foothill Blvd
San Luis Obispo, CA
93401 In Cal. call
(800)592-5935 or
(805)543-1037

ARCADE PRO FOOTBALL™



An Armchair Quarterback's Dream Come True!

It's 4th down and 14 on your opponent's 44 yard line. You're out of time outs with just over a minute left in the game. You call it...sweep right, screen pass left or go for the bomb!

Arcade Pro Football™ puts you in the middle of 60 minutes of exciting gridiron play. You control the action *and* call all the shots. It's the most realistic computer sports game available and it's designed exclusively for the Atari 400/800™ personal computers by Arcade PLUS.

With features like these, Arcade Pro Football™ is in a league of its own!

- Play offense *and* defense head-to-head against the computer or against another player!
- Control two full-color, animated teams in 3-D perspective on a 100 yard, scrolling field!
- 4 game variations, including player handicapping!
- Over 25 offensive and defensive play possibilities...passing, *and* catching, running and kicking!
- All the fun and excitement of real Pro Football...penalties, interceptions, fumbles, bad snaps, 30 second clock—even 4-channel sound and a crowd to cheer you on!

Arcade Pro Football™ is available on cassette or disk for Atari® 400/800™ computers with 16K minimum memory from your local Atari computer dealer. Or send $29.95 (cassette)/$34.95 (disk) plus $2.50 postage and handling (California residents please add 6% sales tax).

arcade PLUS

5276 Hollister Avenue Suite 208 Santa Barbara, CA 93111 (805) 683-2305

Comment Your Catalog

Richard Cornelius
Department of Chemistry
Wichita State University
Wichita, KS

Since the first day that I had my Apple II, I have been frustrated by the inability to fully identify stored programs and files except by using long names. Wouldn't it be nice, for example, to have the date of the latest revision of a program stored along with its name? Of course, a person can always make the date part of the name, but I thought that there ought to be a better way. There is a better way. I have written a program to make writing comments in the catalog easy.

Control Characters

You may have already discovered that some control characters can be part of program and file names in the catalog. For example, a CTRL-J at the end of a program name is helpful in formatting the catalog. The CTRL-J is a linefeed which, when entered as the last character in a program name, has the effect of leaving an empty line between that program name and the next one when the catalog is listed. Another control character which can be inserted into a program name is CTRL-G which will make the Apple beep when the name of the program is listed in the catalog.

Most of the other control characters can be entered into program names, but generally they are not particularly useful. One application they do have is based on the fact that control characters in a name do not actually appear on the screen in the catalog, but they must be used in order to access the program on the disk. Their invisibility can provide a measure of security by preventing someone else from readily loading programs off of your disk. (See your Apple DOS manual for a program to detect most of these control characters.)

The control character that I have found useful in creating comments for the catalog is CTRL-H, the backspace character. This character cannot easily be entered directly into a program name. Typing CTRL-H is the same as pressing the left arrow; you can backspace over characters, but the character that you backspace over is deleted from the name as you backspace. The solution to this difficulty is to put CHR$(8) into a string variable that you use as the program name. In *immediate mode,* [not in a program – just type it on the screen directly] try going through the routine below using an initialized disk with only the HELLO program on it:

```
]CATALOG
DISK VOLUME 254
 A 002 HELLO
]D$=CHR$(4)
]NAME$="ABC"+CHR$(8)+CHR$(8)+"DEF"
]?D$"SAVE";NAME$
]CATALOG
DISK VOLUME 254
 A 002 HELLO
 A 004 ADEF
]LOAD ADEF
FILE NOT FOUND
]
```

The lines that start with a "]" prompt are the ones that I typed into the Apple. The others are those that the computer wrote. When I try to load ADEF the computer tells me FILE NOT FOUND because the name is not ADEF, but "ABC" + CHR$(8) + CHR$(8) + "DEF". Although the program name in the catalog appears to be four characters long, if you were to ask ?LEN(NAME$) you would find that it is actually eight characters long.

This information about CHR$(8) is really all that you need in order to be able to write comments into your catalog. You simply create a string variable that contains enough backspace characters to backspace over the letter that identifies the file type and the number that gives how many sectors are occupied on the disk by the file. Once all of that information is backspaced over, the desired comment is entered into the string. The string variable is then used as shown above to SAVE a program – any program. The "comment" is actually the name of a program – whatever program you had in memory when you do the SAVEing – but it doesn't look like a program name because the file type and sector-count information is missing.

Some Limitations

This commenting technique does have its limitations. Names of programs are limited to 30 characters by DOS. Since the first character of a name cannot be a control character, seven backspaces are needed to erase the information that is normally printed. The first character, plus these backspaces, consume eight of the available 30 characters, so only 22 characters can go into a comment. In addition, you have only limited control over where in the catalog the comment appears. This kind of comment is best used for disks on which people are not going to be making many changes. As long as you start with a fresh disk and put the files, programs, and comments onto the disk in the order you wish them to appear, the catalog will come out fine. If you modify programs in such a way as to change their length, then the order of items in the catalog may be changed and the comments will no longer be adjacent to the program name. One more limitation is that hard copies of the catalog are harder to make appear as nice as the screen listing of the commented catalog. If you try to print the catalog directly, the printer will backspace and overstrike the original characters.

This difficulty can be overcome by listing the catalog on the screen and then, using a program such as that by Jeff Schmoyer (**COMPUTE!** #6) to route the screen image to the printer. In spite of these limitations, I have prepared commented catalogs such as the one in Figure 1. Each line of letters is actually a program name, but the only programs of interest are the ones that have the file type and sector count next to them. The other program names serve only as comments, and the actual programs could be anything (or nothing).

Clearly typing all of these names with the CHR$(8) feature inserted could be quite a chore at the keyboard, so I wrote a program to enter the comments into the catalog. The program is called simply "Catalog Commenter" and is a short BASIC (Applesoft) program. The program shows just how long the name can be and lets you either erase or write names. It then gets a catalog so that you can see what you have done. Hitting any key clears the screen and takes you back to the beginning of the program. This program is the one that was used to prepare the catalog Figure 1. After the backspace characters, two spaces are inserted into the initial part of the string variable used for the name. This spacing makes the comments appear lined up with the sector count of the "real" program names in the catalog, but further limits the length of the comments to 20 characters.

Figure 1.

```
DISK VOLUME 254

 A 025 PH PLOT-BUFFER CAPACITY
    (MAIN PROGRAM WHICH
    LOADS OTHER FILES)

*B 002 OR LOADER & LINE ERASE
    (OVERLAYS HIRES PAGE
    2 ONTO PAGE 1 AND
    ERASES HIRES TEXT
    LINES. A$300; A$325)

*B 027 MZCHAR3
    (SPECIAL WHITE CHAR-
    ACTER SET. A$6000)

*B 006 INSTRUCTIONS
    (BINARY TEXT FILE OF
    INSTRUCTIONS.A$8000)

*B 034 COVER PAGE
    (BINARY HIRES FILE.
```

```
100 REM  ** CATALOG COMMENTER**
110 REM BY RICHARD CORNELIUS
120 REM CHEMISTRY DEPARTMENT
130 REM WICHITA STATE UNIV.
140 REM WICHITA, KS  67208
150 REM (316) 689-3120
160 REM  **INITIALIZATION**
170 D$= CHR$(13) + CHR$(4)
180 REM D$ SIGNALS DOS COMMAND
190 N$= CHR$(8) + CHR$(8) + CHR$(8)

200 REM CHR$(8) IS BACKSPACE
210 N$="A"+N$+N$+CHR$(8)+"  "
220 HOME: VTAB 5
230 REM  **GET COMMENT**
240 PRINT "TYPE IN COMMENT"
250 PRINT"---UP TO THIS LONG--"
260 INPUT"";C$
270 PRINT
280 PRINT"WRITE(W), ERASE(E), OR QU
    IT(Q)?";
290 GET G$
300 IF G$= "Q" THEN 410
310 IF G$ <> "E" AND G$ <>"W" THEN
    GOTO 220
320 REM  **CREATE PROGRAM NAME**
330 N$= N$ + C$
340 REM  **WRITE TO DISK**
350 IF G$= "E" THEN 370
360 PRINT D$"SAVE";N$:GOTO 380
370 PRINT D$"DELETE";N$
380 PRINT D$"CATALOG"
390 GET G$
400 IF G$ <> "Q" THEN 220
410 PRINT:PRINT"THE END"
```

©

STARFIGHT3

David R. Mizner
Houston, TX

STARFIGHT3 is a program that will let you fight off Klingons to save the Federation. Before you start typing away, a little word of warning is needed. This program *loves* memory. In fact, STARFIGHT3 will use it all up; so be careful entering the program. An extra space added now may cause a "no memory" message later.

Have fun!!!

Program Description

A new Galaxy is generated each time the program is RUN. A random number of stars (maximum of 25) and Klingons (maximum of 3) are generated and, along with the Enterprise, are randomly placed in a 10x10 Galaxy.

The Enterprise is equipped with three photon torpedos for every Klingon, and three shield units. Three hits on the Enterprise from Klingon attacks will deplete its shield, a fourth hit will destroy the enterprise. There will be self-destruction if the Enterprise runs into a star or Klingon while traveling around the Galaxy.

Klingons (all that have not been destroyed) will fire at the Enterprise if your response time for a command is too slow or if your torp misses. Only one hit on the Enterprise is allowed per attack. Take note that the Klingons fire their torps in eight directions while the good guys can only fire in one direction at a time. However, neither side can fire through a star.

The stars and Klingons remain stationary throughout the game.

Program Directions

1. Observe operating procedures for VIC20.

2. Commands

a. Move: VIC will request direction and distance. Direction is a number from 1 through 8, while distance is the number of spaces you want to move.

b. Torp: VIC will request a direction. Torp does not have a distance since a photon torpedo will travel until it hits a star, Klingon, or Galaxy boundary.

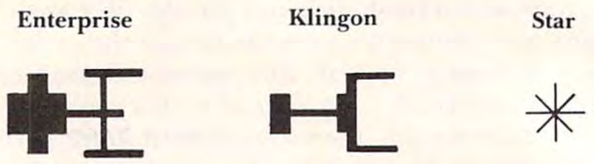
c. End: This command ends the game. "You surrendered" is the real meaning of "end."

3. Scan

a. A scan is generated before each command request.

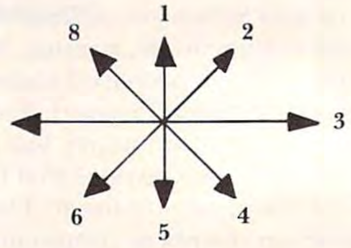
b. The Galaxy is displayed so you can see the actual location of stars, Klingons, and the Enterprise. At the same time, the direction code is printed out.

c. Scan code.



4. Direction

The direction for moving the Enterprise or firing a photon torpedo is given by entering a number from 1 through 8. These numbers will let you move or fire a torp every forty-five degrees.



5. Changing the game's difficulty

a. You can change the number of torps allowed by modifying line 120.

b. Another way is to change the time you are allowed before the Klingons fire. The value of TIS is changed by modifying lines 450,545, and/or 1530.

```
10 PRINT"{CLEAR}  ** STARFIGHT3 **"
20 PRINT:PRINT"DAVID R MIZNER,SEP81"
30 X=PEEK(56)-2:POKE52,X:POKE56,X:POKE51
   ,PEEK(55):CLR
40 CS=256*PEEK(52)+PEEK(51)
50 FORI=CSTOCS+511:POKEI,PEEK(I+32768-CS
   ):NEXT
60 FORI=7168TO7175:READJ:POKEI,J:NEXT
70 DATA15,68,228,254,228,68,15,0
80 FORI=7448TO7455:READJ:POKEI,J:NEXT
90 DATA7,12,204,252,204,12,7,0
100 POKE36869,255
110 DIMA%(10,10),KL(6)
120 FORI=1TO10
130 FORJ=1TO10
140 A%(I,J)=0
150 NEXTJ
160 NEXTI
170 K=INT(RND(1)*3+1):S=INT(RND(1)*25+1)
180 KC=K:T=3*K:H=3
190 FORI=1TOS
200 GOSUB840
210 IFA%(C1,C2)<>0THEN200
```

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

NEW VIC-20 SOFTWARE

General Purpose

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

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

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

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

Mathematics

MathPak
*** Complex Function Calculator:** Use your VIC as a handy complex number calculator with eight registers to store complex numbers, recall them, operate on them, and restore them.
*** Matrix Operations:** Add, subtract, multiply, find the inverse and determinant of any matrices with this menu driven program.
*** Systems of Linear Equations:** Solve any system of linear equations with this menu driven program.
*** Function Analysis:** Finds zeros, min., max., derivative, or integral of any user supplied polynomial function.
*** Statistics I and II:** Computes mean and standard deviation. Performs linear and nonlinear regression on a user supplied data set as well as histograms.
*** Special Functions:** Use trigonometric functions not found in VIC's instruction set with this easy to use program.
*** Data and Function Plotting:** Plots any user supplied functions or data set with labels on the VIC printer.
Each program $9.50
Complete MathPak $49.50

Educational

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

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

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

Games & Entertainment

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

Color Sound Graphics

Each $7.00
Any three $17.00

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

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

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

```
220 A%(C1,C2)=1
230 NEXTI
240 FORI=1TOK
250 GOSUB840
260 IFA%(C1,C2)<>0THEN250
270 A%(C1,C2)=2:KL(I)=C1:KL(I+3)=C2
280 NEXTI
290 GOSUB840
300 IFA%(C1,C2)<>0THEN290
310 A%(C1,C2)=3:E1=C1:E2=C2
320 PRINT:PRINT:PRINT"KLINGONS",K
330 PRINT:PRINT"TORPS",T
340 PRINT:PRINT"STARS",S
350 FORI=1TO3000:NEXT
360 GOSUB860
370 PRINT:PRINT"ENTER YOUR COMMAND"
380 PRINT"1=MOVE 2=TORP 3=END"
390 TI$="000000"
400 INPUTC
410 IFTI$<"000015"THEN440
420 GOSUB1130
430 GOTO360
440 ONCGOTO470,580
450 PRINT">YOU SURRENDERED"
460 GOTO1420
470 PRINT:PRINT"ENTER DIRECTION,DISTANCE"

480 C1=E1:C2=E2:TI$="000000"
490 INPUTC,D
500 IFTI$<"000015"THEN530
510 GOSUB1130
520 GOTO350
530 IFC>8ORD>14THEN490
540 A%(E1,E2)=0:GOSUB670
550 E1=T1:E2=T2
560 IFA%(E1,E2)=1ORA%(E1,E2)=2THENPRINT">
    HIT A STAR OR KLINGON":GOTO1420
570 A%(E1,E2)=3:GOTO360
580 IFT>0THENGOSUB1270
590 IFT>0ANDKC>K0THEN360
600 PRINT">NO MORE TORPS"
610 IFKC>1THEN640
620 PRINT">RAM LAST KLINGON"
630 GOTO470
640 PRINT">YOU'RE OUTNUMBERED"
650 PRINT">FEDERATION IS LOST"
660 GOTO1420
670 ONCGOTO690,700,710,720,730,740,750
680 U=-1:V=-1:GOTO760
690 U=-1:V=0:GOTO760
700 U=-1:V=1:GOTO760
710 U=0:V=1:GOTO760
720 U=1:V=1:GOTO760
730 U=1:V=0:GOTO760
740 U=1:V=-1:GOTO760
750 U=0:V=-1
760 FORI=1TOD
770 T1=C1+I*U:T2=C2+I*V
780 IFT1<1ORT1>10ORT2<1ORT2>10THEN820
790 IFA%(T1,T2)>0THEN830
800 NEXTI
810 GOTO830
820 T1=C1+(I-1)*U:T2=C2+(I-1)*V
830 RETURN
840 C1=INT(RND(1)*10+1):C2=INT(RND(1)*10+
    1)
```

```
850 RETURN
860 PRINT:PRINT" *** SCAN ***"
870 PRINT:PRINT" ++++++++++++"
880 FORI=1TO10
890 PRINT" +";
900 FORJ=1TO10
910 ONA%(I,J)+1GOTO940,960,980
920 PRINT"@";
930 GOTO990
940 PRINT" ";
950 GOTO990
960 PRINT"*";
970 GOTO990
980 PRINT"#";
990 NEXTJ
1000 ONIGOTO1020,1030,1040,1050,1060,1070,
    1080
1010 GOTO1090
1020 PRINT"+  COURSE":GOTO1100
1030 PRINT"+":GOTO1100
1040 PRINT"+     1":GOTO1100
1050 PRINT"+   8 2":GOTO1100
1060 PRINT"+  7   3":GOTO1100
1070 PRINT"+   6 4":GOTO1100
1080 PRINT"+     5":GOTO1100
1090 PRINT"+"
1100 NEXTI
1110 PRINT" ++++++++++++"
1120 RETURN
1130 FORM=1TOK
1140 C1=KL(M):C2=KL(M+3):D=14
1150 IFA%(C1,C2)=0THEN1210
1160 PRINT">KLINGON SHOOTING"
1170 FORC=1TO8
1180 GOSUB670
1190 IFA%(T1,T2)=3THEN1230
1200 NEXTC
1210 NEXTM
1220 GOTO1260
1230 H=H-1:IFH<0THEN650
1240 PRINT:PRINT">ENTERPRISE IS HIT"
1250 PRINTH"SHIELD UNITS LEFT"
1260 RETURN
1270 PRINT:PRINT"PHOTON TORP DIRECTION"
1280 TI$="000000"
1290 INPUTC
1300 IFTI$<"000015"THEN1330
1310 GOSUB1130
1320 GOTO1410
1330 C1=E1:C2=E2:T=T-1:D=14
1340 IFC>8THEN1270
1350 GOSUB670
1360 IFA%(T1,T2)<>2THEN1400
1370 A%(T1,T2)=0:KC=KC-1
1380 IFKC=0THENPRINT"> FEDERATION SAVED <"
    :GOTO1420
1390 GOTO1410
1400 GOSUB1130
1410 RETURN
1420 PRINT:PRINT
1430 INPUT"ANOTHER GAME 1=YES";Z
1440 IFZ=1THEN120
1450 END
```
©

"Probably the best-documented programs I've seen for the PET/CBM."

- Robert Baker, Sept. 1981 KILOBAUD/Microcomputing

HESBAL : Assembler/editor for PET and VIC
HESCOUNT, HESLISTER : Debugging tools for PET and VIC
HESPLOT : Hi-res graphics subroutines for VIC
HESCAT : Disk cataloging system for PET
HESCOM : Exchange data, programs between PET and VIC

Also, many games for the VIC

VIC

Write today for your free catalog

HES

Human Engineered Software
3748 Inglewood Blvd. Room 11
Los Angeles, California 90066

(213) 398-7259
24 HOURS
SOURCE ID:
TCK308

PET, CBM, and VIC are trademarks of Commodore

Dealer Inquiries invited

Swirl And Scribble

Matt Giwer
Annandale, VA

Swirl produces extremely complex designs in Graphics 8 which have to be seen to be appreciated. These are not simple sinusoidal or trigonometric plots, but rather are of some artistic merit and may be suitable for logos, letterheads and the like.

The basis for these plots is the set of equations in lines 230 and 235. They arise from the study of modern control theory and are of interest in that a very small change in the two input constants, A and C, can produce a very large change in the shape and character of the plots. The program is easily adaptable to computers other than the Atari by simply plotting the values of X and Y as in line 250. The values of R and T merely center the plot on the screen.

On your first few plots you will notice that, for the first minute or so, the points will all be in a small area in the center. REM lines 2249 and 2250 show how to change line 250 to show this region. To get you started the 501 through 660 REM lines show pairs of values for A and C respectively.

A note of caution: since this uses Graphics 8 + 16, if the program should end, the display will go back to Graphics 0 and tell you that it is ready. This is why the I loop in line 215 is set to 3000. Although a few hundred would be more than enough to fill enough to fill the screen the extra hundreds hurt nothing and permit unexpected phone calls and the like.

Scribble

A computer program should be scaled to its users. Scribble is a simple program thrown together at the insistance of my six year old who remembered that our last computer had a built in game called scribbling. The Atari would never be up to his standards until there was a way to scribble on the screen. So in order to keep down the heated discussions as to which computer to hook up to the TV I threw this short program together. To my surprise this little program is held higher in his estimation than Star Raiders and is second only to his favorite sea serpent. I offer it here for your child's enjoyment.

To use, a joystick is inserted into position number one and this draws a line on the screen. Pushing the trigger erases the screen. No other provision for operator interaction is made. Keeping it simple kept it popular.

Scribble

```
1 REM NAME SCRIBBLE
1100 GRAPHICS 5+16
1102 COLOR 1
1210 A=STICK(0)
1220 IF A=7 THEN X=X+1
1230 IF A=11 THEN X=X-1
1240 IF A=14 THEN Y=Y-1
1250 IF A=13 THEN Y=Y+1
1260 IF A=6 THEN X=X+1:Y=Y-1
1270 IF A=5 THEN X=X+1:Y=Y+1
1280 IF A=9 THEN X=X-1:Y=Y+1
1290 IF A=10 THEN X=X-1:Y=Y-1
1400 IF X<0 THEN X=0
1410 IF X>79 THEN X=79
1420 IF Y<0 THEN Y=0
1430 IF Y>47 THEN Y=47
1500 PLOT X,Y
1510 IF STRIG(0)=0 THEN GRAPHICS 5+16
1550 GOTO 1210
```

Swirl

```
50 GRAPHICS 0
80 ? :? :? :?
90 ? "INPUT A AND C :";
100 INPUT A,C
110 ? "A=";A;"  C=";C
151 GRAPHICS 8+16:COLOR 1
152 R=150
153 T=85
154 SETCOLOR 2,1,0
155 SETCOLOR 1,4,13
170 X=1
180 Y=1
215 FOR I=1 TO 3000
220 S=X
230 X=A*Y+C*X+S*X*X*(1-C)/(1+X*X)
235 Y=-S+C*X+2*X*X*(1-C)/(1+X*X)
250 TRAP 315:PLOT X+R,Y+T:TRAP 40000
315 NEXT I
```

```
320 GOTO 220
330 END
501 REM 1.01,-1
502 REM 1.01,-.95
503 REM 1.01,-.92
504 REM VERY GOOD, BLACK HOLE 1.01,+0.8
505 REM 1.01,-.1
600 REM 1.0001,-2
601 REM A RANGE .999 AND .992; C RANGE -
2.0055 AND -1.9
650 REM 1.01,0
651 REM 1.008,+.001<>-.001
660 REM 1.008,+.05<>-.05
2249 REM TO SEE THE CENTER OF THESE PLOT
SCHANGE LINE 250 TO
2250 REM LINE 250 TRAP 315:PLOT X*10+R,Y
*10+T:TRAP 40000
4900 END
5000 GRAPHICS 0:LIST 1,330
```

©

ATARI®

Software for Personal Computers

A collection of 10 challenging programs created to provide a unique entertainment value — and 2 personal/business programs with broad functional value. In disk and/or cassette as indicated.

GAMES	• HELICOPTER BATTLE	Req. — 16K RAM/Cassette	$ 9.95
		— 16K RAM/Disk	14.95
	• HORSE RACING	Req. — 16K RAM/Cassette	9.95
		— 16K RAM/Disk	14.95
	• KENO	Req. — 8K RAM/Cassette	9.95
		— 16K RAM/Disk	14.95
	• LIGHTNING BOLTS and REACTION	Req. — 16K RAM/Cassette	9.95
		— 24K RAM/Disk	14.95
	• THE MAD MARBLE	Req. — 8K RAM/Cassette	9.95
		— 16K RAM/Disk	14.95
	• MUSIGAME (2 Games)	Req. — 16K RAM/Cassette	9.95
		— 24K RAM/Disk	14.95
	• SUPERMASTER	Req. — 8K RAM/Cassette	9.95
		— 16K RAM/Disk	14.95
	• TAG	Req. — 16K RAM/Cassette	9.95
		— 16K RAM/Disk	14.95
	• TRACTOR BEAM	Req. — 8K RAM/Cassette	9.95
		— 16K RAM/Disk	14.95
	• WAR AT SEA	Req. — 16K RAM/Cassette	14.95
		— 24K RAM/Disk	19.95
BUSINESS	• CCA Data Management System	Req. — 48K RAM/Disk	99.95
	• LETTER WRITER	Req. — 24K RAM/Disk	19.95

® Trademark of ATARI, Inc.

DIVISION OF CUSTOM ELECTRONICS, INC.
CE SOFTWARE

238 Exchange St., Chicopee, Massachusetts 01013
(413) 592-4761

Mastercard & VISA Accepted
• Dealer And Distributor Inquiries Invited
• Closed Mondays — Open Daily 'Til 5:30 — Fridays 'Til 8

Every PET Needs a Friend.

CURSOR™ is the best friend your Commodore PET will ever have. Since July, 1978 we have published 170 of the most user-friendly programs for the PET available anywhere. When we write or edit a program, we spend lots of time fussing about how it will treat you. We pay attention to all the little things that help make using a computer a pleasure instead of a pain.

Naturally, **CURSOR** programs are technically excellent. Each program that we publish is extensively edited or rewritten by a professional programmer. But *imagination* is just as important as being user-friendly and technically good! We delight in bringing you off-beat, unusual programs that "show off" the abilities of your PET or CBM.

CURSOR is user-friendly, technically great and full of imaginative programs. And every issue of **CURSOR** is still available! We continue to upgrade previously published programs so that they'll work on the four varieties of Commodore ROMs. All issues from #23 on also work on the 80 column CBM.

For only $5.95 you can buy a sample issue and judge for yourself, or send $20.00 and we'll mail you the four most recent issues of **CURSOR** (over 20 excellent programs for your PET). Each C-30 cassette tape includes five programs, a graphic Front Cover, ready to LOAD and RUN on your PET. VISA and Mastercard welcome.

Who knows? After your PET meets **CURSOR**, things may never be the same!

CURSOR is a trademark of The Code Works.

AUTHORIZED DISTRIBUTORS:

Great Britain
AUDIOGENIC, Ltd.
P.O. Box 88
Reading, Berkshire

Holland
COPYTRONICS
Bergemeester
Van Suchtelenstraat 46
7413 XP Deventer

Japan
SYSTEMS FORMULATE CORP.
Shin-Makicho Bldg. 1-8-17
Yaesu, Chuo-ku, Tokyo 103

Australasia
MICROCOMPUTER HOUSE, LTD.
133 Regent Street
Chippendale, Sydney
N.S.W. Australia 2008

THE CODE WORKS

Box 550
Goleta, CA 93116
805-683-1585

WEBS

Loran Gruman
Burnsville, MN

Here is a one-player game for a 40-column PET [or an 80 column machine with the program in **COMPUTE!** #12, pg. 130 loaded — Ed.]. If your machine has sound, turn it on.

```
100 REM WEB WRITTEN 1980 BY LORAN G
    RUMAN 2300 SO SKYLINE DR. ~
    BURNSVILLE
110 REM MINNESOTA, 55337
120 POKE59467,0
130 PRINT"{CLEAR}{02 DOWN}{08 RIGHT
    RIGHT}{REV}WEB INSTRUCTION
    S:{OFF}"
140 PRINT"{02 DOWN}YOU ARE THE NUMB
    ER."
150 PRINT"KEEP THE MOVING NUMBER FR
    OM TOUCHING ANYWEB ON THE ~
    SCREEN."
160 PRINT"{DOWN}THE NUMBER IS CONTR
    OLLED BY PUSHING:
170 PRINT"{DOWN}{03 RIGHT}8=UP      ~
              8"
180 PRINT"{03 RIGHT}4=LEFT          ~
        B"
190 PRINT"{03 RIGHT}6=RIGHT         ~
      4C5C6"
200 PRINT"{03 RIGHT}2=DOWN          ~
        B"
210 PRINT"{03 RIGHT}5=STOP          ~
        2"
220 PRINT"{DOWN}            TEN HITS A
    ND YOUR OUT.{DOWN}"
230 PRINT"{02 DOWN}{04 RIGHT}PUSH A
    NY KEY WHEN READY TO START
    "
240 GETK$:IFK$=""THEN240
250 PRINT"{CLEAR}":A=32768:F=49
260 R=INT(RND(1)*500)+1:Q=A+R
270 GETB$
280 IFB$="4"THENC=-1:S=1
290 IFB$="6"THENC=1:S=1
300 IFB$="8"THENC=-40:S=1
310 IFB$="2"THENC=40:S=1
320 IFB$="5"THENC=0:S=0
330 IFC=40ORC=-40THEN360
340 IFP+C>39ORP+C<0THENC=0:S=0:GOTO
    360
350 P=P+C
360 IFAA=ETHENE=INT(RND(1)*25)+1:I=
    TT:TT=INT(RND(1)*4)+1:AA=0
370 IFTT=1THENQ=Q+1
380 IFTT=2THENQ=Q-1
390 IFTT=3THENQ=Q-40
400 IFTT=4THENQ=Q+40
410 IFTT=4ANDI=3THENQ=Q+1
420 IFTT=3ANDI=4THENQ=Q-1
430 IFQ>33768THENTT=3:GOTO360
440 IFQ<32768THENTT=4:GOTO360
450 LETAA=AA+1
460 POKEQ,81
470 IFA+C>33767ORA+C<32768THENS=0:G
    OTO270
480 T=T+S:IFS<>0THEN GOSUB680
490 A=A+C
500 V=PEEK(A)
510 IFV<>32ANDDV<>FTHENN=1
520 IFV=FTHENN=1
530 IFC=0THENN=0
540 IFN=1THENGOSUB650
550 F=F+N:IFF=58THEN570
560 N=0:POKEA,F:GOTO270
570 PRINT"YOU SCORED A TOTAL OF";T"
    {LEFT} ":PRINT:GOSUB690
580 PRINTTAB(30);"          ";
590 PRINT"                          ~
                     {02 LEFT}{0
    2 UP}"
600 PRINT"DO YOU WISH TO PLAY AGAIN
     (Y/N)";
610 GETPG$:IFPG$=""THEN610
620 IFPG$="Y"THENCLR:GOTO250
630 IFPG$="N"THENPRINT"{CLEAR}THANK
    S FOR PLAYING ":END
640 IFPG$<>"Y"ORPG$<>"N"THEN610
650 POKE59466,0:POKE59467,16:POKE59
    466,15
660 FORNN=30TO90STEP6:POKE59464,NN:
    NEXT
670 POKE59467,0:RETURN
680 POKE59464,150:POKE59467,16:POKE
    59466,15:FORZ=1TO10:NEXT:P
    OKE59467,0:RETURN
690 POKE59466,0:POKE59467,16:POKE59
    466,51
700 FORNN=225TO120STEP-2:POKE59464,
    NN:NEXT:FORNN=120TO255STEP
    2
710 POKE59464,NN:NEXT:POKE59467,0:R
    ETURN
```

©

Review:

Votrax Type 'n Talk: TNT

Charles Brannon
Editorial Assistant

The concept of the Votrax Type 'n Talk speech synthesizer is simple: you send the device a word, and it pronounces it. For example, the command PRINT#1, "HELLO" would cause the Votrax to say "hello." This makes programming it simple and fun. Other synthesizers can require you to construct words from one or two letter *phonemes*, the simplest units of speech. For example, the word "hello" might be coded as: "H EH3 L O" or "[@X&." Yet another kind of synthesizer lets you send English words, but has a memorized vocabulary which is limited by memory size. What makes Votrax unique is the combination of ease-of-use and flexibility.

The voice is distinct and understandable, but it is obviously artifical. It sounds robotic, similar to the voice synthesizers found in many arcade and electronic pinball games. Both volume and frequency (pitch) can be adjusted with knobs. The voice sounds most natural at its lower frequency.

Built into the unit is a "text-to-speech" algorithm that converts English words into phonemes that can be pronounced by the device – no easy task. Considering the complexity of the English language, it is a remarkably good algorithm, permitting you to generate speech with straightforward PRINT statements. Its arbitrary methods can cause some problems. "COMPUTE!" sounds like "comput." "HELLO" sounds a bit slurred, "HUH LO" sounds better. It is sometimes necessary to intentionally misspell. "COMPUTE!" sounds excellent when spelled "COM PEWT." The space breaks longer words into distinct syllables. Some few words are tougher to generate; for example, MOUSE becomes "mus" (the *ous* is treated like the *ous* in danger*ous*). Spelling it MOWSE doesn't help; it comes out "mose" as in *most*. To solve any such problems you can also program speech directly with phonemes.

Is Votrax for you? It depends on the application. Votrax can be the basis for some fascinating dialogue games, such as ELIZA and Adventure. It can liven up arcade games with threats, taunts, and warnings (We Are The MURLOD Invaders).

Voice synthesis is an alternate (superior?) man-machine interface; it can streamline business (can you imagine your computer saying "Please insert the Word Processing Disk?"). It would be of tremendous aid to the blind, where every character typed could be spoken and, when SPACE was pressed, the preceeding word spoken.

Votrax can be attached to almost any computer, via an RS-232 interface. It can even be attached between the computer and another device, permitting data to be spoken automatically (CompuServe becomes TalkuServe?). Although a one-watt amplifier is built in, you must provide a speaker (eight-ohm).

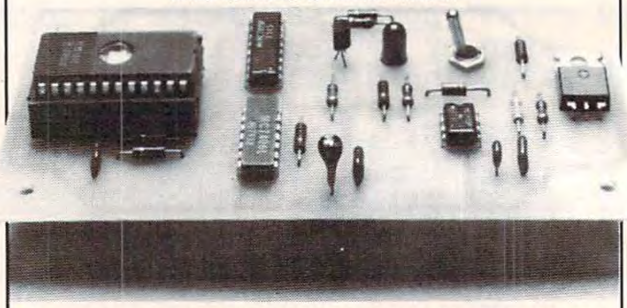
The significance of the Votrax Type 'n Talk is its text-to-speech routine. It permits beginners to use it immediately, and relieves professionals of the tedium of phoneme construction. The Votrax deserves its acronym – any device that can pack so much power into such a small box is truly TNT!

Votrax Division of Federal Screw Works
500 Stephenson Highway
Troy, Michigan 48084
$375

©

EPROM PROGRAMMER

Shown assembled. EPROM not included.



For single supply 2516, 2716 & 2758 EPROMs. Connects through a user supplied interface to any computer system. Interfacing requires two 8-bit ports plus hand-shake lines. One of the ports must be software controllable for input or output. Timing is done via hardware, thus is independent of MPU clock rate. Verify erased. Program — entire or partial. Auto verify after programming. Transfer contents to RAM for modifying or duplicating.

Select Documentation for:	**Interface to:**
6502	6820 PIA or 6522 VIA
6800	6820 PIA
6809	6820 PIA
8080/8085/Z80	8255 PPI

Comprehensive documentation booklet contains schematic, instructions for construction, check-out and use, and a well commented assembly listing for the specified MPU.

Complete kit of parts (includes ZIF socket)......... $ 45.00
Bare PC board and Documentation................ $ 25.00
Software listings for additional MPUs
(with purchase of Kit or PC board) .... $ 5.00

Ordering: Specify MPU. Add 5% for P&H. Overseas add 10%. Ariz. residents add 5% tax.

MasterCard **Micro Technical Products, Inc.** VISA
814 W. Keating Ave., Dept. M
Mesa, Arizona 85202 • 602-839-8902

Review:

Olympia's ES 100 KRO Typewriter/ Printer

Richard Mansfield
Assistant Editor

The ES100, one of a line of Olympia printer-typewriter combinations, can serve as an advanced, stand-alone typewriter with correction facilities or as a computer printer. It contains a built-in RS 232-C serial interface and will work with most personal computers. As one of the new "intelligent" typewriters, it operates somewhat differently from the venerable machines so common only a few years ago.

The first thing you notice is that very little is *mechanical* – you don't move margin stops, you simply set them with left and right margin keys. All keys are repeating, when used with the "repeat" key. Reverse vertical half-line spacing (for superscripts), choice of two pitches, reverse tabulation, CR without LF, and several line spacings are all key-selected. Unlike the older generation of electrics, most of the formatting and spacing is done from the keyboard. As when using a word processor, you can move around the page without taking your fingers from the keys.

Another feature of this latest generation of typewriters is their feel. They resemble a computer keyboard in layout, versatility, and touch. Instead of a direct mechanical relationship between a pressed key and struck paper, the keys simply click to let you know that they've been acknowledged by the system. The 96-character typewheel responds at 16 characters per second (if you could type that fast). This separation of the mechanical from the keyboard activities makes sense when the printing mechanism does not care whether it gets information from the keys or from a computer.

A green LED shows, on a numbered scale, the precise typing position. The value of some of these features might not be immediately obvious, but, in use, their utility becomes clear. The carriage return without line feed, for example, makes underlining easier. Reverse tabulation means that you don't need to return to the left hand margin to access the tab stops – you can move left through the stops as well as tabbing right, the traditional direction.

Specifications

The typewriter stores functions in an accumulator with the margin release and tab settings "remembered" for 70 to 90 hours. A "correction memory" allows the revision of up to eight characters if the mistake is noticed at once. Depending on the platen size (13/15 inches) the printer supports a maximum paper width of 12.9/15.3 inches and a line length of 11.6/15.5 inches. The unit weighs 30.3/36.3 pounds.

There are 92 characters on the keyboard and line spacing can be either 1, 1½, or double. Horizontal spacing (keyboard selected) is between ten and twelve characters per inch. A variety of typestyles are available on the printwheels and there are five types of ribbon cartridges (black, black/red, carbon, correctable film, or multi-strike).

Using a standard Type D 25 connector, the interface permits odd, even, or no parity bits and the data rate is jumper selectable between 110, 134.5, 150, or 300 baud.

There are a variety of "daisywheel" printers on the market. These printers feature excellent, crisp lettering and typefaces which are easily and quickly changed. The Olympia ES 100 KRO deserves to be considered even if the intent is simply to upgrade an older electric typewriter to the new generation of intelligent electrics. If you ever want computerized, full word processing – the purchase of one of the state-of-the-art electrics would make the transition painless.

Olympia USA, Inc.
Box 22
Somerville, NJ 08896
$1680

©

MEMOREX
FLEXIBLE DISCS

WE WILL NOT BE UNDERSOLD!! Call Free **(800)235-4137** for prices and information. Dealer inquiries invited and C.O.D.'s accepted.

PACIFIC EXCHANGES
100 Foothill Blvd.
San Luis Obispo, CA
93401. In Cal. call
(800)592-5935 or
(805)543-1037

VISA

RPL: A FORTH Sequel?

Jim Butterfield
Toronto, Canada

RPL is a FORTH-related language produced by Samurai Software. There are versions for all PET/CBM machines, and it will fit in systems as small as 8K. It is similar to FORTH in many ways ... but there are fundamental differences.

RPL stands for Reverse Polish Language. This is the backwards-type of coding which calls for you to write X + Y as X Y + and PRINT M as M PRINT. Owners of Hewlitt-Packard calculators will be used to this kind of thing by now and, in fact, it makes good coding sense to do it this way.

Proprietary

Since RPL is a proprietary system, the language must be considered in a different category from FORTH, FORTRAN, or BASIC. It seems unlikely that competing RPL's would be generated by various sources, and RPL literature will be confined to a relatively small community of purchasers.

Timothy Stryker, the author of the language and compiler, has taken many of the characteristics of FORTH, rebuilding and reconceptualizing as he saw desirable. The result will not please FORTH traditionalists – it has a different style from FORTH – but it does form an interesting new language.

Faster? Simpler?

One-to-one comparisons of FORTH versus RPL programs shows that RPL fits in slightly less space and runs slightly faster. This is surprising, since FORTH is known for its compactness and high speed.

Savings in time and memory are achieved, at least in part, by reducing the generality of the language. FORTH works interactively with a user; each program module can be checked out the moment it is typed in, and the user can try things out as he builds his program. RPL is less interactive: the user writes code and then gives the command COMPILE to generate a runnable program. This allows RPL to be more efficient, but reduces user interaction; however, RPL has features to offset this problem during debugging.

Another reason for RPL's speed and compactness is in the internal representation of the program. FORTH uses threaded code, where each "action" of a command is represented by a subroutine address; RPL uses p-code, with each action represented by a token value.

RPL has a streamlined vocabulary of operators; slightly over forty commands are implemented, and all are useful. This compares well with FORTH, which seems to the beginner to be cluttered up with hundreds of commands, many of which are seldom needed by the programmer. The commands are nicely chosen for newcomers; many closely parallel BASIC keywords.

PET/CBM owners will be pleased to see that their machine's characteristics are well supported by RPL. BASIC can co-exist with RPL, and file input/output capabilities are preserved. There's a danger, of course: Programs using "custom" features won't transport well to other computers.

SIM, a symbolic debugger, is sold as a separate package. It allows users to try out sequences of commands before writing them into a program. It has a nice way of presenting the stack visually which may help give users an intuitive feel for how RPL works.

Considerable documentation comes with RPL (60 pages) and SIM (12 pages). The material is nicely written and is quite well done; the approach is tutorial in nature and uses examples liberally.

We've been comparing RPL to FORTH because of the similarities in the languages. RPL deserves to be rated on its own merits.

It's not as easy as BASIC or as pretty as APL. But RPL is fast, compact, and relatively straightforward to program. Users will have to learn to cope with stacks and the backwards-like Reverse Polish Notation. It may take a particular mentality to get hot in an RPL-like language; but the payoff in efficiency can be very good.

Samurai Software, P.O. Box 2902, Pompano Beach, FL 33062. RPL Compiler, $49.95 on disk, $44.95 on cassette; Debugger Compiler and Debugger are ordered together. Specify computer ROM system and disk type. ©

Review:

Ricochet

Richard Mansfield
Assistant Editor

An intriguing new game from Automated Simulations, Ricochet (for the Apple, Atari, or TRS-80) demonstrates why there is so much new interest in games. With the advent of the computer, suddenly there are entirely new categories of games: simulations, interactive adventure stories, exciting hybrids which combine the preplanning involved in traditional strategy games like chess with the visual, physical action of games like pinball.

Ricochet falls into the hybrid category; it has to be seen to be understood, but it's something of a combination of pool and checkers. Each player (you vs. the computer or you vs. a friend) has nine "bars" which initially appear in front of a set of "bumpers." The bars start out in a 4-3-2 pattern, guarding your bumpers, since your opponent can score points by landing in your bumpers.

There are two possible ways to react during your turn. You can change the arrangement of your bars or you can *launch* which sends a ball out from one of your corners ricocheting off walls, bars, and bumpers, and gaining points for each one hit. The ball continues to ricochet until it goes past a bumper into space or hits a corner launcher. Hitting a corner, aside from ringing up points, can render that particular launcher useless for the remainder of a game. You make your moves and launches either from the keyboard or with joysticks.

Broadly defined, the idea is to arrange your bars (which toggle between vertical and horizontal orientation, when hit) so that you best protect your bumpers and launchers. Likewise, you attempt to launch in such a way as to maximize the damage to your opponent.

A Smart Clock

Ricochet takes full advantage of the computer's ability to handle many variations of play. If you play against the computer, it can take on four distinct "personalities" each of which use different strategies. Beyond this, there are five variations of the game itself. In variant two, you can win extra launches, and variant three adds two extra bumpers to each side. Variant five removes all the position markers from the playfield and it becomes more difficult to predict the ricochet effects of a launch.

If a player takes too much time planning or arranging his bars, a *smart clock* starts giving points to the opponent. It is smart because it determines how much is "too much time" by averaging the opponent's decision-making time. In effect, if you make your moves quickly, you force your opponent to move quickly too.

The game is "intelligent" in several senses. If you lose a game, the next game adds point value to your opponent's bumpers while your bumpers retain their original value. This evens things up since you will score more points when you hit the opponent's bumpers.

In the past few years, with computers becoming widely available in homes and game arcades, a variety of new types of games have appeared. Ricochet is an excellent example of this emerging art form.

Automated Simulations, Inc.
P.O. Box 4247
Mountain View, CA 94040
Apple, Atari, TRS-80. $19.95

©

Memory Expansion Boards for the ATARI* Computer

Completely compatible with ATARI hardware and software

No modifications necessary

Fully assembled and tested

User installable — simply plug it in

One year warranty

Gold connector tabs

16K Memory Board — $49.95
32K Memory Board — $124.95
Build Your Own Memory
16K board · no components — $10.00
32K board · no components — $30.00

Add $2 Shipping & Handling per board

*ATARI is a trademark of Atari Inc.

Anderson Peripherals, Inc.
P.O. Box 629 • Richardson, TX 75080 • 214-231-6866

Review:

Atari Microsoft BASIC (Part I)

Jerry White
Levittown, NY

Editor's Note: This review is in three parts. The second and third parts will appear in **COMPUTE!** *April and May. — RTM*

Not long ago, the Atari Personal Computer owner had two programming alternatives: 8K Atari BASIC and Assembler Language. Now there are three versions of BASIC from which to choose, plus PILOT and PASCAL.

The most recent Basic on the market is called Atari Microsoft BASIC (AMSB). Those of you familiar with other versions of Microsoft will feel right at home with the Atari version. It is said to be the most powerful Microsoft of them all and will certainly make program conversion much easier. The manual provides all the information needed for converting from many other versions of BASIC including PET BASIC, Apple and Applesoft BASIC, Radio Shack Level II BASIC, and Atari 8K BASIC.

This series of articles is being written to help you decide if AMSB is for you. If the Atari is the only computer you've ever had, and 8K Atari BASIC is the only version you've ever used, you will need some specific comparisons to understand the advantages and disadvantages of using AMSB.

Disadvantages??? Yes, although AMSB provides dozens of advantages over 8K Atari BASIC, there are always two sides to every story. So let's get the bad news out of the way first.

The most obvious of the bad news is cost, about $80.00. You'll also need at least 32K RAM and a disk drive since, as of this writing, AMSB is available only on diskette and requires 11,252 bytes more than 8K Atari BASIC. Since the language must load from disk, there's 40 seconds of boot and load time.

Some Tradeoffs

If you can live with the previously mentioned disadvantages, you'll surely find the power and flexibility of AMSB worth looking into. There are, however, a few other sacrifices that must be made by the 8K BASIC user. AMSB has no immediate syntax error checking and permits only two abbreviations, ? = PRINT and ! = REM. Oh how I miss typing GR.0. You also must give up that unlimited length string in trade for string arrays. The 8K STICK, STRIG, PADDLE, and PTRIG commands are not included, but they are easily replaced with PEEK and POKE.

Now for the good news! Here are a few of the most significant advantages AMSB has to offer:

COMMAND	PROVIDES...
AUTO	Automatic line numbering.
COMMON	Variable values are passed from one program run to another.
DEF	Define integer, single, and double precision.
DEL	Delete range of lines from program.
DIM	Three Dimensional Alpha/Numeric Arrays
ELSE	IF THEN ELSE decisions.
INSTR	Search for a small string within a larger string.
MOVE	MOVE a number of bytes from one area of memory to another.
OPTION	Reserve RAM for Assembler Routines, Player Missile Graphics, Redefined Character Sets.
PRINT	AT specified coordinates.
PRINT	(TAB) and (SPC) positioning.
PRINT	USING for formatting output such as right justified currency amounts.
RENUM	Renumber lines and references.
TIME	In 60ths of a second.
TIME$	Current time in HH:MM:SS format.
TRON	Current line number trace display on.
	Turn off trace function.
VERIFY	Verify Program in memory with program on tape or disk.
WAIT	Loop until specified conditions exist.

Many commands are identical in both Atari BASICs. Some commands perform identical functions but are formatted differently. For example, 8K BASIC uses the XIO command for many useful functions. AMSB makes things easier to remember with commands like FILL, KILL, LOCK, MERGE, NAME, and UNLOCK. ASMB uses PLOT TO instead of DRAWTO, CLS instead of ?CHR$(125), and SCRN$ instead of LOCATE.

Some of the other commands available in AMSB include AFTER, CLEAR STACK, EOF, ERL, ERR, ERROR, INKEY$, LEFT$, LINE INPUT, MID$, ON ERROR, RANDOMIZE, SAVE with LOCK, STACK, and STRING(n,X$).

One beautiful feature was added to the SOUND command. An optional fifth variable for duration has been added. The duration is a value of up to 255 JIFFIES (60ths of a second). Up to 25 SOUND commands may be stored on the STACK, eliminating the need for many time delay loops. AMSB can go on to calculations or display work while SOUND commands execute at previously specified intervals.

The ability to define integers allows floating point routines to be bypassed. This can account for significantly faster execution. How much faster, you ask? I'll get into speed comparisons and routine examples in part two of this series.

I Use All Three

Before closing this segment, I'd like to voice some of my own personal opinions. AMSB will certainly find its place in the rapidly growing Atari software market. Both the beginner and experienced programmer can benefit from the wide range of commands offered. The buyer should also be aware of another alternative called BASIC A+.

Anything you can do using AMSB can be done in 8K BASIC with occasional help from an Assembler subroutine. AMSB offers a great deal to the BASIC only programmer, but cannot be used by those with less than 32K RAM or without a disk drive. Personally, I've grown to really appreciate the amazing number of features Atari BASIC has squeezed into an 8K ROM cartridge. I've also learned to appreciate fast binary I/O and the DIR (Disk Directory) feature available in BASIC A+, as well as the speed made possible by the AMSB integer feature. They all have their advantages and disadvantages. Which one do I recommend you ask? I use all three. ©

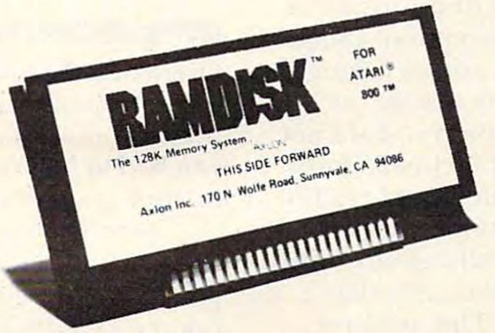
HAVE YOUR CAKE AND EAT IT TOO

IT'S A PIECE OF CAKE TO CONNECT AN ACTEK TRIX I INTERFACE TO YOUR OLIVETTI PRAXIS 30 OR 35 CORRECTING ELECTRONIC TYPEWRITER. DAISY WHEEL QUALITY AT DOT MATRIX PRICES!

- 10+ CPS
- 15 MINUTE INSTALLATION
- HALF SPACE JUSTIFICATION
- CABLE REMOVES IN SECONDS
- TYPEWRITER FUNCT. UNIMPAIRED
- AVAILABLE NOW: ATARI & APPLE
- OTHER DIRECT CONNECTIONS AND RS232 AVAILABLE SOON
- PRINT AND PRINT #N OPERATE
- NO INTERFACE NEEDED: USES FRONT CONNECTOR · ATARI USES CONTROLLER PLUG · APPLE
- PRICE: $215 · APPLE ADD $10 †BASED ON WARDS PR30 PRICE
- TYPEWRITER AND SERVICE WIDELY AVAILABLE

Actek
12225 SW 2nd/SUITE 200 - E
P.O.B. CCC
BEAVERTON, OR 97075

ATARI*800*OWNERS



PLUG IN AND GO!

The Axlon RAMDISK Memory System provides 128K of RAM memory which can be utilized as an additional disk device or bank selectable RAM memory! The DOS supplied with the system allows you to utilize the RAMDISK Memory System as a disk device in conjunction with your Atari 810*. The system is up to 20 times faster than the Atari 810 and is compatible with existing Atari 800 software. As user memory, the RAMDISK Memory System is organized as eight (8) 16K banks. The system is installed with two 16K RAM modules giving you a 160K Atari 800 system. Drop by your local computer store for a demonstration or contact Axlon Inc. for more information.

- Plug-in Compatibility
- 128K Bytes of RAM Memory
- Compatible with existing Atari 800 Software
- Can be utilized as an additional disk - function for function, up to 20 times faster than the Atari 800
- Includes DOS Memory Management Software
- Can be utilized as Bank Selectable User Memory
- Gold Plated Contacts
- 90 Day Warranty

170 N. Wolfe Road
Sunnyvale, CA 94086
(408) 730-0216

AXLON

* Indicates Trademark of Atari, Inc.

TELECOMMUNICATIONS

Modem Applications

Michael E. Day
Chief Engineer
Edge Technology

In **COMPUTE!**, *September, 1981, #16, Mr. Day discussed technical specifications for MODEM's. Here he explores several uses for MODEM's in everyday computing. — The Editors*

One of the questions I am often asked is: "Why do I need a modem?" It is interesting that this question would be asked rather than just "Do I need a modem?" since this indicates several things. The need for the modem is already felt.

The feeling of the need for the modem comes about because of the large amount of information presented to the person about telecommunications both in magazines such as this one and in talking to other computer users. This tends to lead to the belief that if you do not have a modem you are not using your computer to its fullest potential. Unfortunately, the reasoning for this belief is not readily apparent. Analysis of the information generally presented on telecommunications shows why this is so. The most common type of information that is presented is of a technical nature. This assumes that you already know why you do or do not need a modem, and are simply after "how does it work" information. The other type of information that is presented is applications information. Again this assumes that you already know why the modem is needed, and that you are simply after the information on how to use it for a particular type of application.

The question *why* is one of the hardest of this type to answer. It cannot be answered directly. When you ask *why,* what you are really saying is give me more information so that I can decide if I really need it. The information that is normally provided is reference information with which you are familiar. In answer to "Why do I need a car?", one might answer "In order to get to and from work." This provides a base point that you can expand upon to gain the information needed to determine how the car would fit into your lifestyle. A response could be "But I can take the bus." with a return of "But what if you work odd hours when the bus doesn't run?" This generates the pros and cons necessary to make a final decision.

The problem that we have with the modem is the same problem that the computer has experienced — a lack of readily discernible common reference points. In answer to why do I need a computer, the easily determined reference points tended to be rather weak, such as to balance your checkbook, or keep records of your gas mileage. Since these could be done far more cheaply with existing alternative methods, they hardly generate a decision in favor of the computer. The computer is slowly overcoming this problem by creating its own reference points. The computer is doing things that were not possible before (controlling heating and lighting to minimize utility bills, or writing letters (or magazine articles) with greater ease than ever before, even playing exciting new games and, as a side benefit, you can balance your checkbook too.

The modem is going through the same stage of development of use. It is a device that has entirely new uses and concepts that are not currently realized, and it must "create" these in our awareness so that they can be realized of their own accord.

Computerized Bulletin Boards

Originally the question was easy to answer, the modem was for the purpose of operating a computer from a remote location. If you had to do this, you had to have the modem. If you did not have to, then you did not need a modem.

Now, however, that use of the modem has been radically altered. With the advent of the personal computer we can put the computer at the remote location along with the user.

If you are only going to use the computer to play games or balance your checkbook, you probably don't need a modem. If you want to communicate with other computer users, however, there is a very good probability that, at some point, you will need a modem.

One of the new uses is the Computerized Bulletin Board Systems that appeared. These are public access message systems which can be used by anyone to post messages or read those left by others. These tend to be messages that don't fit into normal modes of communication and include calls for help, general notices of information, advertisements, classifieds, and personal messages. There is no charge for the use of these systems, they are

HUNTINGTON COMPUTING

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

Atari®

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

Order by Phone

800-344-5109

800-692-4146

In California

Pet®

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

Apple®

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

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

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

Great Grandma Huntington

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

TRS-80®

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

FOREIGN ORDERS

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

VISCALC

Special for Pet®, Atari® & Apple®

Regular $200.00 List

Now $149.00

HUNTINGTON COMPUTING
Post Office Box 1235
Corcoran, California 93212

Order by Phone 800-344-5109
In California 800-692-4146

Apple* is a registered trademark of Apple Computer, Inc.
Pet* is a registered trademark of Commodore.
TRS-80* is a registered trademark of Tandy Corp.
Atari* is a registered trademark of Atari, Inc.

(209) 992-5411

VISA

master charge

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

wholly supported through donations and out-of-pocket expenses by the owners.

As an outgrowth of the BBS's are the remote computer systems and database systems. Although many of them are open to the general public, they are not readily usable due to the technical knowledge needed. Additionally, these systems tend to be very specific in the application to which they are oriented and are generally of little or no use to the general public.

Because the bulletin boards are privately supported, they are limited in the scope of services they can provide. For those who are willing to pay, there are more elaborate systems available. The most widely-known are Compuserve, The Source, and Micronet. These systems provide a wider range of services including message transfer, information retrieval (stock reports, news, etc.), conferencing, program storage and retrieval, and running programs.

Often there is a need to find information of a more extensive or technical nature than can be provided by the general services systems. This need is provided for by the technical information database systems. These systems are usually oriented around a particular subject area or group of areas. The technical data systems, by being very specific can carry a much wider range of information on a subject than is possible on a general information system. Because this information is also the most expensive to obtain, these systems are the most expensive to use. They can cost over $100 an hour.

Multiuser Systems

Finally we come to the original multiuser computer systems, time-share computer systems. These systems are rented on a usage basis to anyone who needs a computer, but, for some reason, does not have a computer of his or her own available. These are generally used for overflow work, temporary, or occasional applications where it is not possible or practical to use one's own computer. The cost of using these systems can vary widely depending on how the usage is determined.

It is interesting that now that the personal computer has come into being, another application appears to be evolving. This can best be understood by describing the need that has been generated.

If you wish to say something to George who lives down the street, you could go to his house and speak to him directly, or you could call him up on the telephone and talk to him. In the first case there was no *equipment* involved in talking to George, you went to his house. This is *direct* communications. In the second instance you used the telephone to talk to him. Rather than expend the energy to go to George, you used a device which allowed you to talk to George without actually going to his house and thus you were *communicating at a distance*.

If you and George both have a computer and you wish to share programs you have written, there are many ways this could be done. You could put a copy of the program on a cassette or floppy disk and give it to George to read into his computer. This works great if George has a similar computer and can read the tape or disk.

If the two systems are not compatible, another way will have to be found. One way that has been used a lot is for you to simply provide George with a written copy of the program and let him type it into his system. This isn't too bad if the program isn't very long and is in human-readable form. This is the way most magazines provide programs as it is the surest way to cover a wide range of computers. But, as mentioned, if the program is not in a human-readable form, or is excessively long, this method does not work very well.

Computers Talking To Computers

A method of communication that computer hobbists have often used is to directly tie their computers *back to back*. This is a form of *direct communication*. This allows the computers to talk to each other, but has the disadvantage of requiring that both computers be next to each other. To date, it has also meant that the computer operator be fully knowledgable of the way the computer internals work as well as the programming needed to allow the two computers to talk to each other. This can be a bit much for the general user and, in fact, has baffled quite a few experienced computer technicians.

The modem provides a common link that both computers can communicate through. By defining a standard of how the interconnection between the computers is to be accomplished, the problem of how to hook the two computers together is eliminated. What is occuring now is a definition of the method of communication between the computers. Although there are some communications programs in use already, they are currently machine-type dependent. An Apple can talk to another Apple, but it can't talk to an Altair. Most of the programs that are used to allow one computer to communicate to another are in the early stages of development: they allow the communication to occur, but there is little or no provision for options or alternatives. They tend to be very restrictive in their use.

As the need to communicate between different types of computers grows, the communications programs will become simultaneously more comprehensive and easier to use. ©

7 ATARI PRODUCTS

THE MONKEY WRENCH

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

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

Cartridge and Manual – $49.95

TYPING EXERCISE FOR ATARI

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

EPROM CARTRIDGE

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

Cartridge and Manual – $19.95

OTHER NICE STUFF

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

MAE (Macro Assembler Editor)

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

- MAE was written entirely in machine language – not in Basic like some assemblers we know of. Thus you get very fast and accurate assemblies.
- Contains a machine language monitor with numerous commands for debugging machine code.
- 38 error codes, 27 commands, 26 pseudo ops, and 5 conditional assembly operators.
- Contains a word processor, example files, and learning aid.
- Requires at least 32K of memory.
- All commands oriented for disk operation with ATARI 810 disk drive.
- Macro, Conditional Assembly, and Interactive Asembly capability.
- Sorted Symbol Table.
- Optionally creates executable object code in memory or relocatable object code on disk.
- 50 page manual. **810 Diskette and Manual – $169.95**

(requires license agreement)

MACRO ASSEMBLER AND TEXT EDITOR (ASSM/TED)

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

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

810 Diskette and Manual – $53.95

MACHINE LANGUAGE MONITOR FOR ATARI

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

Cassette version – $24.95 Diskette version – $29.95

MEMORY TEST FOR ATARI

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

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

Call Orders: (919) 924-2889

VISA

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

MasterCard

OLYMPIC SALES COMPANY

Authorized Dealer
apple computer

NEW APPLE FAMILY SYSTEM

LARGE INVENTORY LOWER-THAN-EVER PRICES

- ★ 48K Apple II +
- ★ Disk w/controller
- ★ RF Modulator
- ★ 7 Personal & game disks

Retail: $2495.00
Call for BEST Price!

Large Inventory of:
Disk with controller DOS 3.3
Second Disk Drives
Pascal–Fortran–Cobol languages
Dow Jones & Quote reporter
Graphics Tablets
Visicalc for Apple II & III
Smarterm 80 column card
Micromodem II by DC Hayes

WE ALSO CARRY SOFTWARE!

Personal Software	Innovative
Peach Tree Software	American
Microsoft	System Plus
Micropro	and more.

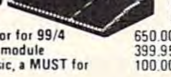
We are an authorized servicing Apple dealer for Apple II & III Immediate delivery on all models–we carry an enormous inventory of Apple products at all times!

Just some of the software in stock NOW–
Magic Window, Raster Blaster, Ultima, Home $ Minder, Tax Preparer, Pool 1.5, Typing Tutor, Olympic Decathalon, ABM, Robot War, Castle Wolfenstein, Cranston Manor, Expediter, OO-Topos, Cyborg, Gorgon, Falcons, Wizardry, Personal Filing System, PFS: Report, Flight Simulator. . .

Texas Instruments Home Computer

New–1982 Model with full typewriter-style keyboard, U/L case & more!
TI-99/4A New KEYBOARD
$369.95

Retail $950.00 Your Cost $369.95!



New keyboard

	Retail	Your Cost
10" color monitor for 99/4	650.00	319.95
32K Exp. mem. module	399.95	314.95
Extended Basic, a MUST for 32K module	100.00	75.00
Speech synthesizer	150.00	129.95
Disk memory drive	500.00	397.95
RF modulator	69.50	49.50
Telephone coupler (modem)	225.00	189.95
Printer (solid state)	400.00	319.95

We carry a large inventory of software, & accessories

HEWLETT PACKARD

HP-11C Advanced Programmable Scientific LCD *Retail: $130.00* *Your Cost:* **$114.95**

HP-12C Advanced Programmable Financial LCD *Retail: $150.00* *Your Cost:* **$129.95**

	Retail	Your Cost
HP-125 New Microcomputer 64K CPU/Terminal/Keyboard	3750.00	3195.00
HP-85 Microcomputer–built-in printer/monitor	3250.00	2499.00
HP-83 Microcom–built-in mon	2250.00	1799.00
HP-2631B Printer, dot mtrx	3950.00	3199.00
HP82905A 80 col printer	945.00	755.00
HP-2601A Letter quality prter		3495.00
HP-82901M Dual master	2500.00	1999.00
Call us for the lowest prices on 7 disk drives		
Graphics plotter 7225	2450.00	2089.00
Personality module–a must for 7225	750.00	636.00
Graphics Tablet 9111A	2050.00	1699.00
HP-41CV	325.00	256.00
Card Reader for 41CV	215.00	171.00
Printer for 41CV	385.00	294.00
Quad Ram	95.00	84.95
Optical Wand for 41CV	125.00	99.95
HP-41C Calculator	250.00	188.95
Memory mod. for 41C		26.95
HP-97 Progrmble printer	750.00	579.95
HP-34C Progrmble scientific	150.00	117.95
HP-38C Progrmble business R/E	150.00	119.95
HP-32E Advanced scientific	55.00	48.95
HP-37E Business management	75.00	58.95

We carry a complete inventory of Libraries, accessories & supplies.

Mattel 1670 Race Horse Computer
Ret. $125.00 Y/C: $24.95

ATARI Programmable Color TV Game Video Console: *Retail:* $225.00 **$137.95**

MATTEL ELECTRONICS INTELLIVISION
Retail: $325.00 *Your Cost:* **$234.95**

VIDEO TAPES

Min order: 3, mixed OK

SONY	L500	2 hr	$11.89
SONY	L750	3 hr	14.69
RCA	VK250	6 hr	13.95

TEXAS INSTRUMENTS

	Retail	You Pay
TI-59	295.00	179.95
TI-3011 LCD Stu Slide/NEW		18.95
TI-35SP LCD Sci		22.50
TI-40 LCD Sci/NEW		28.95
Bus. Anal. I		19.95
Bus. Anal. II		44.95
Bus. Card		39.95
MBA		54.95
Invest. Analyst		48.95
TI-54 Adv. LCS Sci/NEW		39.95
TI-5511 NEW		44.95
TI-57 Prog. Sci		39.95
TI-58C 480 Step. Prog.		89.95
PC-100C Print/Plot		149.95
LCD-Programmer/NEW		59.95

ATARI COMPUTERS

Model		Retail	Your Cost
800	16K	1080.00	759.95
810	Disk Drive	600.00	469.95
Atari	VisiCalc	200.00	169.95
825	Impact Printer (Centronics 737)	1000.00	779.95

SUPER SPECIAL–ATARI 400 (16K)
Retail: $595.00 Your Cost: $339.95
Language Cart. not included–Opt'l at $54.95

OHIO SCIENTIFIC

Professional Computers

	Retail	Your Cost
C8PDF–48K	3495.00	3195.00

- Dual 8" Drives
- 64 col x 32 line/color
- .7 MIPS *FAST!*
- Many more stndard featres

Fortran & Pascal available

Many other OSI products available –at discounted prices, of course!

VISA · master charge

Main Showroom & Offices:
216 South Oxford Avenue
Los Angeles, CA 90004
WE HONOR
VISA and MASTERCHARGE

TELEX: 67 34 77
ORDER DESKS open 7 Days a Week!
7:00 AM to 7:00 PM Mon thru Sat
Sunday Noon to 5:00 PM
Order Desks: (213) 739-1130

TOLL-FREE (outside Calif.)	TOLL-FREE (within Calif.)
800-421-8045	800-252-2153

All goods subject to availability; this ad supersedes all previous ads; we are not responsible for typographical errors; we will meet or beat any advertised prices if the competition has the goods on hand.
Minimum shipping and handling $4.95.
All orders subject to verification and acceptance.

EPSON PRINTERS

	Retail:	Your Cost
MX80	645.00	499.00
Optional Graftrax Chip 80		95.00
MX80 FT	745.00	599.00
MX100	995.00	749.00

DIABLO 630

Letter Quality daisy wheel w/tractors
Retail: $3395.00
Y/C: $2595.00

PAPER TIGER PRINTERS

460	995.00	895.00
460G graphics	1094.00	969.95
560	1295.00	1099.00
560G graphics	1394.00	1195.00
445	795.00	695.00
445G	894.00	789.00

SANYO MONITORS High resolution

13" Color (new) high quality	550.00	419.95
12" Green phosphorous	360.00	259.95
12" Black & white	340.00	239.95
15" Black & white	370.00	259.95
9" Black & white (the best seller)	235.00	169.95

AMDEK (Leedex) High Quality Monitors

100	12" B/W, 12 MHz	179.00	139.95
100-G	12" Green, 12 MHz	199.00	174.95
300-G	12" Green, 18 MHz	249.00	199.95
Color I	13" Color, NTSC comp. input,	449.00	339.95

Machine Language: Loops And Quality

Jim Butterfield
Toronto, Canada

Program loops seem to be a byproduct of laziness. When a programmer tires of writing a series of instructions, he produces a loop to save coding time and processor memory. Yet something more profound happens at the same time: the program usually becomes more generalized.

Suppose I wanted to place the value hexadecimal 20 into locations $8000 to $8027. My first instinct is to code: LDA #$20 : STA $8000 : STA $8001 : STA $8002 ... and so on. Around the time I reach $800B, it will probably occur to me that I'm writing a lot of essentially similar code. Creative sloth comes into play. I observe that the repeated instruction is STA $something. Racking my brains, I decide that if I could vary the "something" part, I could then do most of the job with a variable instruction.

"Indexing!", I cry, and proceed to tear up the old sheets and code LDX #$00: LDA #$20: (loop) STA $8000,X: INX: CPX #$28: BNE (loop). This drops coding to six instructions instead of forty-one and memory usage to twelve bytes instead of one hundred and twenty-two; but the running time increases from 162 to 443 microseconds. There's no use crying over spilt microseconds: the time difference is less than a three-thousandth of a second, and I'll usually happily take it rather than a case of writer's cramp.

But something more important has happened than just mechanics. If I want to convert my first ("hard way") program so that it stored into 64 locations, or stored to address $0400 and up, I have no choice but to rewrite. On the second program which uses loops, it's a snap. A mere stroke of the coding pen, a one or two byte change, and the job's done. We've somehow created a program that's more general and more applicable to a range of tasks.

As we consolidate our program, we have to generalize. And as we generalize, we not only shorten the code: we create sturdier and more broadly applicable code.

A word to those picky bit-and-microsecond counters who will point out that we could save two bytes and a few dozen microseconds by starting our index X at 39 and counting it down to zero. Sure you can. But that kind of picking is not what makes sounder code. We want to look for methods that generalize; they are the ones that will produce sturdy and reliable code ... and perhaps save us a few coding lines and bytes.

A Larger Scale

The same ideas apply to coding that repeats several lines. When you find yourself writing the same code, look for a generalization. Take these two sets of coding:

```
ONE   LDX  #$09          ZRO   LDX  #$06
      PHA                      PHA
ONE1  BIT  CLKRDI        ZRO1  BIT  CLKRDI
      BPL  ONE1                BPL  ZRO1
      LDA  #126                LDA  #195
      STA  CLK1T               STA  CLK1T
      LDA  #$A7                LDA  #$A7
      STA  SBD                 STA  SBD
      ...                      ...
```

The above subroutines are from the tape write program of the KIM. ONE writes a logic 1 to tape; ZRO writes a logic 0 to tape. They are very similar. The only differences are: nine versus six on the first line, and 126 versus 195 on the fifth line. How might we consolidate these two pieces of program?

At the moment, the Y register doesn't seem to be used. We could ask the calling routine to set Y to zero or one, depending on whether we wanted to call ZRO or ONE activities; and then write a common routine:

```
ZONE    LDX    TABLE,Y
        PHA
ZONE1   BIT    CLKRDI
        BPL    ZONE1
        LDA    TIMING,Y
        STA    CLK1T
               ...etc.
```

We have now consolidated the two routines. The values 6 and 9 which count the number of cycles in each signal are now stored in a table TABLE. The values 126 and 195 which set the timing of each cycle are in a second table TIMING.

Have we accomplished anything other than saving a few bytes of code? Yes, almost accidentally. Now that the number of cycles are stored neatly in a table, we can easily adjust them to change the type of signal we write. In fact, this particular coding was part of the sequence that lead to the introduction of the high speed tape format known as Hypertape.

Deeper...

The programmer doesn't always have free registers, of course; but the methodology of saving registers isn't hard to do.

Where addresses within a program change from routine to routine, the best way to handle this is via indirect addresses. If program 1 searches table 1, and program 2 searches table 2 and so forth, indirect address.

Consider: if you have written a game with planes and tanks moving around the screen, you may find that, with a little work, a single subroutine can move both craft around. Once you have generalized, all sorts of bonuses arrive: the bombs and shells can likely be folded into the same subroutine. Collisions and other effects can now be handled in their generalized form rather than as special coding (did a bomb hit a shell? did a plane hit a bomb? did a shell go off screen? etc...)

What seems to start out as laziness or convenience develops into something more important. In reaching for the general solution, we write much better code.

Many programmers often find themselves very pleased with a program they have written; it seems "good" to them, although they don't know exactly why. It's usually because they have solved more than the specific problem – they have solved a whole class of problems. ©

VIGIL

VIC

PET/CBM

Exciting, new games interactive language.

- Easy to learn with 60 - powerful commands
- Double density graphics, large number display, sound!
- 9 complete programs including - Breakout, SpaceWar, Anti, U.F.O., Maze
- PET/CBM version for OLD/NEW/4.0 ROMs and 8K of memory
- VIC version has color and enhanced sound (req. 3K memory expander)
- Comprehensive 50 - page manual

	U.S. & CANADA	FOREIGN
VIGIL for PET/VIC w/9 games on cassette (diskette add $3.00)	$35	$40
VIGIL User's Manual (refundable with software order)	$10	$12
VIGIL Interpreter Listing (6502 Assembler Language)	$25	$30

TINY BASIC COMPILER - PET

A true compiler that turns your BASIC program into fast machine code

- Subset of PET BASIC compiles to 6502 machine code. Req. only 8K memory
- Has full floating point capabilities and functions
- Compiler listing available with 16K version (included)
- Can load compiled machine code anywhere in memory
- On cassette (add $3.00 for diskette)

	U.S. & CANADA	FOREIGN
TINY Basic Compiler OLD/NEW/4.0 ROMS/8032 with 8K memory	$25	$30
TINY Basic User's Manual (refundable with software order)	$ 5	$ 7

PET & APPLE II USERS TINY PASCAL

Structured language alternative to BASIC for PET or APPLE II includes:

- LINE EDITOR - creates, modifies and maintains source language
- COMPILER - converts your source to an executable P-code format
- INTERPRETER - executes compiled P-code. Features built-in TRACE
- CASE-OF, WHILE-DO, IF-THEN-ELSE, REPEAT-UNTIL, FOR-TO/DOWNTO, PROC, FUNC
- Graphics version has more: GRAPHICS, PLOT, POINT, TEXT, INKEY, ABS, SQR
  APPLE II has lores & hires-COLOR, HGRAPHICS, HCOLOR, HPLOT, PDL and TONE

	U.S. & CANADA	FOREIGN
TINY Pascal PLUS - GRAPHICS PET 32K NEW/4.0 ROMS/8032/diskette	$50	$55
TINY Pascal PLUS - GRAPHICS PET 32K NEW/4.0 ROMS/8032/cassette	$55	$60
TINY Pascal PLUS - GRAPHICS APPLE II 48K and DOS 3.2/3.3	$50	$55
TINY Pascal NON-GRAPHICS PET 16K/32K NEW/4.0 ROMS diskette	$35	$40
TINY Pascal NON-GRAPHICS PET 16K/32K NEW/4.0 ROMS cassette	$40	$45
TINY Pascal User's Manual (refundable with software order)	$10	$12
TINY Pascal 6502 Interpreter Listing-GRAPHICS version	$25	$30

PIPER-THE MUSIC MACHINE

Here's the simplest way yet to compose, conduct and play music on your VIC/PET. Complete music capability — play all note values, list and edit your music, easily handles repeats and rests. Written in fast machine language. Includes sample compositions and manual. VIC version does not require additional memory. PET version requires external CB2 speaker.

	U.S. & CANADA	FOREIGN
PIPER of PET/VIC on cassette (add $3.00 for diskette)	$25	$30
PIPER User's Manual (refundable with software order)	$5	$ 7
PIPER 6502 Assembly Listing	$25	$30

PET* MACHINE LANGUAGE GUIDE

Now in its ninth printing. Learn the hidden talents of your OLD, NEW or 4.0 ROM PET/CBM with this easy to follow manual. Details 30 of the PET's builtin routines.

	U.S & CANADA	FOREIGN
PET MACHINE LAGUAGE GUIDE for OLD, NEW or 4.0 ROMS	$9	$11

Orders prepaid in U.S. Dollars, VISA, MC, ACCESS, etc. Prices include postage.

ABACUS SOFTWARE
P. O. Box 7211
Grand Rapids, Michigan 49510
616 / 241-5510

VISA

master charge

Teach your Commodore to use the telephone.

Telecommunicate with McTerm

Don't miss out on the computer/telephone revolution. With our McTerm package and a modem, your Commodore becomes an *intelligent* terminal. You can interact with large and small computers anywhere in the world.

Access databases like the Source™, or MicroNet™, or Dow Jones for up-to-the minute news, sports, stock market reports, etc. Write programs on your micro and run them on distant mainframes.

McTerm actually saves you money and time while you use it. There's no need to read data line-by-line as you receive it. Instead, you can quickly save it all to disk or printer and sign-off. You save on expensive computer time *and* long distance phone rates.

McTerm sends and receives faster than anything else on the market — up to 1200 baud. Adjustable duplex, echo, and parity. Runs on any Commodore with Basic after version 1.0. Works with most RS-232 modems and supports auto dial.

MC

McTerm connects you with the rest of the world.

Madison Computer, 1825 Monroe, Madison, WI 53711 USA. 608-255-5552

INSIGHT: ATARI

Bill Wilkinson
Optimized Systems Software

Good news! I have finally found out how and where you will be able to obtain copies of *De Re Atari* ... and it won't even cost you your left thumb. The Atari Program Exchange now has it available for $19.95 plus shipping. The part number for it is APX-90008, and you can order it through 800-538-1862 (800-672-1850 in California). There are several changes and improvements from earlier versions, including a section on the GTIA. One disappointment is that an appendix on random access files has been deleted. Oh well, leaves room for me to do a future article.

The How and Why articles on Atari BASIC that appeared in the last two issues were the result of requests for ways of "hooking into" BASIC, in order to add commands, etc. I am trying to gently break the news that you *can't* add commands to a RUNning program (though direct, keyboard commands can be done by intercepting keyboard input, as I presume the Eastern House "Monkey Wrench" does.). But I have been trying to lead up to *why* you can't add commands, so that people won't waste time on false leads in trying to prove me wrong.

However, I am suspending the How and Why series this month in order to take a look at the USR function. It is my belief that the USR function will give most of you access to all the added comands you could write, which lessens somewhat the impact of not being able to integrate your own commands. In addition to some suggestions on usage, this month we implement a really powerful USR function: one which will play a song (or most any kind of sound) in the background while your BASIC program continues to chug away (zapping Klingons, etc.). Naturally, there will also be the usual mix of tricks, etc.

In order to deliver on my promise to the BASIC users regarding the song-playing USR function, I must first lead the assembly language fanatics through a short intro to the Atari's interrupt system. As far as I know, the Atari is the only low-end personal computer that gives you such complete access to a fully-integrated, usable interrupt system. The Atari OS is structured to take advantage of several of these interrupts; and, more importantly, the user is invited to gain full or partial control of most interrupt routines. This despite the fact that Atari's interrupt service routines are in ROM.

The 6502 microprocessor supports two types of interrupts: NMI (Non-Maskable Interrupt) and IRQ (Interrupt ReQuest). A bit in the CPU status byte controls whether IRQ's will generate interrupts, but if an NMI signal is presented to it the 6502 will always call in interrupt service routine. Atari, however, allows the user to prevent NMI's from reaching the CPU (except for the RESET button), thus giving even greater control. Once again, I must refer you to the Atari Technical Manual for full details, but herewith is a summary of the available interrupts.

Table 1. Available Interrupts

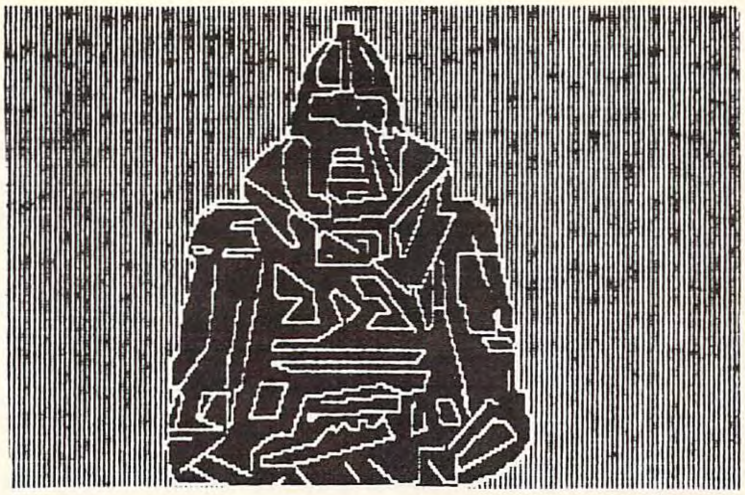
Type	Description
NMI	Reset Button (the only uncontrollable interrupt)
NMI	Display List Interrupt
NMI	Vertical Blank Interrupt (60 times per second)
IRQ	BREAK key
IRQ	any other key
IRQ	Serial Input (for SIO communication with disk, etc.)
IRQ	Serial Output (ditto)
IRQ	Serial Transmission Completed (ditto)
IRQ	Timer #4
IRQ	Timer #2
IRQ	Timer #1
IRQ	6520 parallel port "A"
IRQ	6520 parallel port "B"
IRQ	BRK instruction encountered (internal to 6502)

Each of the available interrupts, except the Reset Button and the BREAK key (and Timer #4 on all except newest machines), has a vector (two byte pointer) through RAM. To take control of an interrupt, simply put the address of your routine in the vector, and OS will call you instead of the default routine. The only exception is the Vertical Blank Interrupt, which is handled slightly differently and is the real subject of this article.

The Vertical Blank Interrupt (VBI) is really the key to many of Atari's unique features. It occurs 60 times per second, at the bottom of each scan of the TV screen, and is used by the OS ROMs to do all sorts of things. First, and perhaps most obvious, it drives the three-byte clock at locations $12,$13,$14 (18,19,20 decimal) as well as several other usable event timers (e.g., serial bus timeout), most of which are accessible to the user. Second, and most useful, it allows changes to the graphics-related hardware at a time when nothing is being displayed on the screen: it moves all the "shadow" locations (see the technical manual) to their corresponding hardware ports.

Of necessity, then, the user would not normally want to interfere with the operations of the VBI routines. But, once again, the Atari software design team thought ahead: they provided not one, but two, VBI vectors. Thus, upon receipt of a VBI request, the ROM code first calls the routine pointed

FOR ATARI



FOR ATARI

The above is a graphics 8 screen printout on the EPSON, with our new AESD II (tm)

★ ★

001 ATARI EPSON SCREEN DUMP
This is a screen dump program that transfers a screen image to the MX-80. . . . . (D) $19.95

002 ATARI EPSON SCREEN DUMP II
This is a screen dump program which allows you to copy anything from the screen. It also supports all graphics modes and text modes. It supports all the features of the EPSON(tm) MX-80 and MX-100. The program is in machine language and is relocatable. . . . . (C) $26.95
(D) $29.95

003 ATAR-RENUM
This is a utility that will renumber any tokenized BASIC program that is co-resident in RAM. . . . . (C) $19.95
(D) $21.95

004 INFO-FILE
This program is designed to act as an electronic filing cabinet. This is a FAST database program that utilizes a dynamic keyboard to move the user quickly through this menu driven program. Use it to create, add, delete, edit, print, selectively search, and store your custom files. (D) $21.95

005 BINARY LOAD CASSETTE TO DISK
This utility will take binary load cassette files like SPACE INVADERS (tm) and allow their transfer to disk. No more waiting for loading! The duplicate is AUTO-BOOTING and uncopyable. . . . . (D) $21.95

006 DEVIL DWELL DUNGEON
This disk based adventure has excellent graphics. Prepare yourself for hours of unpredictable entertainment as you venture into the depths of the Devil Dwell Dungeon in search of the Golden Septor and the rating of Superlord of Superlords. This is an AUTO-BOOTING program. (D) $21.95

007 DOWNLOADER
This is a true SMART TERMINAL EMULATOR PROGRAM which allows you to upload and download files between computers and save to DISK, CASSETTE, or a PRINTER. ALSO WORKS WITH THE D.C. HAY'S SMARTMODEM.
(C) $26.95
(D) $29.95

009 ELECTRONIC CALCULATOR
This program is a tool for the electronics hobbyist. It makes the necessary resistive and capacitive calculations for both series and parallel circuits. It shows formula, decodes resistors, plus power calculations for both AC and DC circuits. . . . . (C) $19.95
(D) $21.95

011 ELBBARCS
This is a word game program which is in high resolution graphics. . . . . (C) $19.95
(D) $21.95

012 UTILITY PAK 1
These four utility's are for the serious programmer. XREF is a variable cross reference utility which tells you where and when a variable is used in a program. VARIABLE-CHANGER is a program that allows you to easily change the name of any or all of the variables in your program. Lister and Denumber are also included. . . . . (C) $19.95
(D) $21.95

013 PIE BAR UTILITY
This utility is designed to provide a screen dump capability for the ATARI® GRAPH IT(tm) using EPSON® MX-80 printer. Features STORAGE and RECALL of both Pie and Bar Charts. Runs in 32K of RAM Screen Dump feature can be used separately in 24K of RAM. . . . . (C) $19.95
(D) $21.95

014 BACKUP MASTER
A machine language program that allows you to make backup copies of boot load diskettes. Also displays any sectors that the disk drive had trouble reading and skips over them. . . . . (D) $39.95

COMPUTER AGE SOFTWARE

9433 GEORGIA AVE.
SILVER SPRINGS, MD 20910
(301) 588-6565

CASH, CERTIFIED CHECK,
MASTERCARD & VISA ACCEPTED
FOR IMMEDIATE SHIPMENT.

PERSONAL CHECKS
ALLOW 10 DAYS TO CLEAR.

to by vector VVBLKI (at $0222) and then calls via the vector VVBLKD (at $0224). The 'I' and 'D' stand for "Immediate" and "Deferred," respectively.

Normally, the user routine would not replace the vector at VVBLKI. Thus the Atari ROM code can update its clocks and move its "shadow" registers in confidence that it will finish its job before the screen starts displaying the next TV frame. The user may replace VVBLKD to cause his routine to execute directly after the Atari system code.

Some cautions are in order: (1) Disaster will strike if your VBI routine is not done before the next VBI occurs. If you simply need to synchronize your routine to a vertical blank, just wait for the system clock to tick before starting (see the label WAITVB in this month's example program). (2) As with most Atari vectors, the safest way to use these is to move them somewhere in your own data area, replace them with your pointer, and have your code finish up by jumping back via the original Atari routine. This is particularly important to do with interrupt handlers, else the interrupt system may not be properly reset.

Finally, let me note that you may, if you really have to, steal the entire VBI processing for yourself. This is not necessarily bad (especially if you are writing a dedicated game, etc.), but be forewarned that you will have to worry about shadow registers, etc., yourself. There is a lot more to this subject, including what Atari refers to as time-critical I/O, but for most purposes you should be able to work within the rules I have outlined.

A Real, Live Example

The example program this month is designed to be used via USR from BASIC, but there is a simplified entry point from assembly language. You could lift this program as is and plunk it into any assembled game, etc. The idea behind the program is simple: a routine is passed a sequence of bytes which are interpreted to be commands to the sound generators of the Atari hardware. The routine examines the bytes and performs the requests. One of the available requests is to "play" sound(s) for a specified length of time; upon encountering this request, the routine waits the appropriate time before processing the next byte. Simple.

Except that this routine will operate (invisible to a running BASIC program) merrily playing

Main Assembly Listing

```
0000          1000        .PAGE "     equates, origins, etc."
              1010 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              1020 ;
              1030 ; PLAYIT  -- a demonstration of performing
              1040 ;            clocked, interrupt-driven
              1050 ;            tasks under Atari OS.
              1060 ;
              1070 ; Written by Bill Wilkinson
              1080 ;   for March, 1982, COMPUTE!
              1090 ;
              1100 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              1110 ;
0600          1120 ORIGIN =     $0600
0000          1130        *=    ORIGIN
              1140 ;
00FF          1150 LOW    =     $FF
0100          1160 HIGH   =     $100
              1170 ;
D200          1180 AUDF1  =     $D200      ; Frequency, audio channel 1 (sound
                   0)
D201          1190 AUDC1  =     $D201      ; Channel 1 control & volume
              1200 ;
0224          1210 VVBLKD =     $0224      ; Delayed Vertical Blank routine
              1220 ;
0014          1230 CLOCKLSB =   $14        ; the system clock, LSB of 3
              1240 ;
00CE          1250 PLAYADDR =   $00CE      ; 2 byte pointer in safe place
              1260 ;
              1270 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              1280 ;
              1290 ; Equates for our private sound commands
              1300 ;
00FF          1310 CMDR   =     255        ; Repeat
00FE          1320 CMDS   =     254        ; Stop sound (keep routine going)
00FD          1330 CMDN   =     253        ; Number of voices
00FC          1340 CMDTV  =     252        ; set Tone and Volume
0000          1350 CMDE   =     0          ; End (but sound not turned off)
              1360 ;

0600          1370        .PAGE "     install our PLAYIT routine "
              1380 ;
              1390 ; INSTALL is the entry point called from BASIC
              1400 ;
              1410 ; The BASIC program calls us via
              1420 ;     USR( INSTALL, ADR(playit-command-string) )
              1430 ;
              1440 ; The routine may be called from
              1450 ;     assembly language at INSTALL1
              1460 ;     by placing the address of the
              1470 ;     command string in A,Y (LSB,MSB)
              1480 ;
              1490 INSTALL
0600 68       1500        PLA              ; BASIC tells us how many parameters
0601 C901     1510        CMP  #1          ; better just have one!
0603 D0FE     1520 GOOF   BNE  GOOF        ; else only RESET will get him out!
0605 68       1530        PLA
0606 A8       1540        TAY              ; MSB to Y register
0607 68       1550        PLA              ; LSB to A register
              1560 ;
0608          1570 INSTALL1 = *            ; assembly language entry point
              1580 ;
              1590 ; first, we wait for a vertical blank
              1600 ; ...to ensure we don't get a VBLANK
              1610 ;     interrupt while we are working!
              1620 ;
0608 A614     1630        LDX  CLOCKLSB
              1640 WAITVB
060A E414     1650        CPX  CLOCKLSB    ; has clock ticked?
060C F0FC     1660        BEQ  WAITVB      ; no...keep waiting
              1670 ;
              1680 ; OKAY TO PROCEED
              1690 ;
060E 85CE     1700        STA  PLAYADDR    ; we preempted a zero page spot
```

along while BASIC continues what it is doing. To accomplish this, we have hooked into VVBLKD (as described above). The user specifies the note duration as a number of "jiffies" (60ths of a second), and we let the VBI count down the duration for us.

The commands are imbedded in a string of bytes passed to the routine. Playit recognizes six command types, as shown in Table 2. Playit is not particularly sophisticated. For example, all voices must play sounds for the same duration and, when chang-

Table 2. Playit Command Codes

Byte value	Name	Description
255 ($FF)	CMDR	Repeat the entire sound command string
254 ($FE)	CMDS	Stop all sounds (do *not* end command string)
253 ($FD)	CMDN	Number of voices is specified in next byte (0-4)
252 ($FC)	CMDTV	Specify Tone and Volume (as in SOUND 0,freq, TONE,VOLUME). Must be followed by 0-4 bytes (one per each voice as specified by CMDN), each of which specifies a Tone/ Volume for one channel.
0 ($00)	CMDE	End command, unhook from VVBLKD. Does not turn off sound, so is usually preceded by CMDS.
any other	---	Any other value is assumed to be a duration, given in 'jiffies' (60ths of a second). Must be followed by 0-4 bytes (one per voice as specified by CMDN), each of which specifies the frequency of the sound for one channel (as in SOUND 0,FREQ, tone, volume).

SPRING SALE

CAVERNS OF MARS *ATARI APX*

The surface of Mars is barren and rubble strewn, but beneath it lies a challenge only the brave and skillful dare undertake. Deep within the Red Planet lies the nerve center of the Martian's stronghold, protected by layer upon layer of the most ingenious defenses the crafty Martians can contrive. Your mission is to take on those legendary—and some say impregnable—defenses and to penetrate to the heart of the CAVERNS OF MARS! Player/Missle graphics, high-resolution color, and fine scrolling routines have been combined to create one of the most addicting games of all times.

Cat. No. 3452	16K, cass, joystick	$29.95
Cat. No. 3453	24K, disk, joystick	$29,95

48K **ATARI 400 OWNERS** 48K

Now Atari 400 owners can solve the nagging problem of "you need more memory". The 48K RAMBOARD from Intec Peripherals comes completely assembled and tested and is fully documented for installation in the Atari 400. All contacts are gold plated for maximum reliability.
Cat. No. 3474 $285.00

VERBATIM DATALIFE DISKETTES

Verbatim Datalife series has become synonymous with quality. If you have an ATARI, APPLE II, TRS-80, or any computer which uses soft-sector diskettes, you can take advantage of this low HW price.
Cat. No. 1147 **$28.00**

YAHTMAN *COMPUTER CORE SOFTWARE*

YAHTMAN is a computerized version of a dice game which has been popular with young and old for years. Up to four people, school age and up, can compete for hours of enjoyment. YAHTMAN takes advantage of the Atari computer's graphics and can be played on both the Atari 400 and 800.
Cat. No. 3405 16K, cass, joystick **$19.95**

TT#5 PLAYER/MISSILE GRAPHICS *SANTA CRUZ SOFTWARE*

This is what it's all about. P/M Graphics is what set the Atari computer a cut above the rest when it comes to graphics. Learn how to create a simple shape called a player and you're on your way. This tutorial is loaded with 25 examples to create programs ranging from a complete business application to a small game.

Cat. No. 3400	32K, cass	$29.95
Cat. No. 3401	32K, disk	$29.95

BE SURE TO STOP BY AND SEE US AT THE 7TH WEST COAST COMPUTER FAIRE IN BOOTH 1310H

HOW TO ORDER
Write or phone. Pay by check, M/C, VISA, or COD (add $1.50 for COD).
Offer expires Mar. 31, 1982
Mention this ad and we pay shipping (UPS ground only). (800) 423-5387
HW Electronics
19511 Business Center Dr. Dept. G3
Northridge, CA 91324 (213) 886-9200

WHEN IN SOUTHERN CALIFORNIA, VISIT OUR RETAIL STORES

HW ELECTRONICS

**19511 Business Center Dr.
Northridge, CA 91324**

**2301 Artesia Blvd.
Redondo Beach, CA 90277**

ing volume or tone quality, all voices must be respecified. A more sophisticated sound interpreter would presumably mean smaller command strings but a bigger interpreter. If you go to the trouble to type in both Playit and Playit From BASIC, you will see that some more than acceptable sounds can be accomodated, so I am reasonably happy with the results.

Some interesting projects remain: Why not convert Atari's Music Composer disk files to Playit-compatible strings? Or how about a real Music Compiler written in BASIC? How about making Playit relocatable, a la last month's article? Please write and tell of your successes (or failures?).

Last but not least, another caution: since I/O to anything but the screen or keyboard uses the SIO serial bus driver, and since the serial bus uses the sound generators to get its baud rates, etc., you MUST turn off sound generation (commands CMDS, CMDE) before doing such I/O.

Atari BASIC: On Sounds, Hex Numbers, And The USR Function

The featured idea and program in this issue is the Playit From BASIC listing which follows. The program itself is not very sophisticated: it simply allows the one-character command codes (R,S, N,T,E) and hex data bytes to be translated into characters in a string. It then passes the address of the string to Playit (the assembly language program) and comes back to the user, ready to compile the next string of commands. If you intend to emulate this scheme, rather than use the program as is, you might be advised to put the sound command string into memory you have reserved (e.g., via the "Simplest Method" given in previous articles in this series). Putting the command in a string is inviting trouble: if your program stops, if you ENTER new

```
0610 8DC206 1710        STA  REPEAT    ; just in case of a repeat cmd
0613 84CF   1720        STY  PLAYADDR+1
0615 8CC306 1730        STY  REPEAT+1  ; similarly for MSB
            1740 ;
0618 AD2402 1750        LDA  VVBLKD    ; prepare to save the ptr
061B AC2502 1760        LDY  VVBLKD+1
061E C93C   1770        CMP  #PLAYIT&LOW ; already saved?
0620 D004   1780        BNE  NOWINSTALL ; no
0622 C006   1790        CPY  #PLAYIT/HIGH
0624 F010   1800        BEQ  INSTALLED ; yes
            1810 ;
            1820 NOWINSTALL
0626 8DC406 1830        STA  SAVEVBLK
0629 8CC506 1840        STY  SAVEVBLK+1 ; save system vector
            1850 ;
062C A93C   1860        LDA  #PLAYIT&LOW
062E 8D2402 1870        STA  VVBLKD    ; and install our own


0631 A906   1880        LDA  #PLAYIT/HIGH
0633 8D2502 1890        STA  VVBLKD+1
            1900 INSTALLED
0636 A901   1910        LDA  #1         ; A single clock tick
0638 8DC706 1920        STA  DURATION   ; until we start playing
063B 60     1930        RTS             ; done with install!
            1940 ;


063C        1950        .PAGE "     The actual PLAYIT routine"
            1960 ;
            1970 ; PLAYIT is the entry point for our Delayed
            1980 ;   Vertical Blank routine
            1990 ;
            2000 ; PLAYIT 'reads' the sound command string
            2010 ;     and plays our 'song'
            2020 ;
            2030 ; SAM is simply the looping point for cmds
            2040 ;
            2050 PLAYIT
063C CEC706 2060        DEC  DURATION  ; keep on playing?
063F D029   2070        BNE  EXIT      ; yep...no changes
            2080 ;
            2090 SAM
0641 20B706 2100        JSR  GETCMD    ; get a byte from command string
0644 C900   2110        CMP  #CMDE     ; End it now?
0646 F053   2120        BEQ  DOEND     ; yes
0648 C9FF   2130        CMP  #CMDR     ; D.C. al Fine?
064A F05E   2140        BEQ  DORPT     ; yes
064C C9FE   2150        CMP  #CMDS     ; Stop all sound ?
064E F03F   2160        BEQ  DOSTOP    ; yep
0650 C9FC   2170        CMP  #CMDTV    ; Tone and Volume on TV?
0652 F02A   2180        BEQ  DOTV      ; yeah
0654 C9FD   2190        CMP  #CMDN     ; Number of voices change?
0656 F015   2200        BEQ  DONUM     ; uh-huh
            2210 ;
            2220 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            2230 ;
            2240 ; if none of the above, must be duration
            2250 ;
            2260 DODURATION
0658 8DC706 2270        STA  DURATION  ; we assume so
065B AEC606 2280        LDX  NUMVCS
065E 300A   2290        BMI  EXIT      ; no voices, just duration
            2300 FREQLP
0660 20B706 2310        JSR  GETCMD    ; yes...get next byte
0663 9D00D2 2320        STA  AUDF1,X   ; and set the frequency
0666 CA     2330        DEX
0667 CA     2340        DEX            ; see if more voices
0668 10F6   2350        BPL  FREQLP    ; yes...keep trying
            2360 ; no...fall through to EXIT
            2370 ;
            2380 EXIT
066A 6CC406 2390        JMP  (SAVEVBLK) ; let OS clean things up
            2400 ;
            2410 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            2420 ;
            2430 ; set number of voices
            2440 ;
            2450 DONUM


066D 20B706 2460        JSR  GETCMD    ; next byte...
0670 AA     2470        TAX
0671 CA     2480        DEX
0672 8A     2490        TXA            ; less one
0673 3003   2500        BMI  NUMOK     ; if < zero, leave it alone
0675 2903   2510        AND  #$03      ; Ensure 1-4 voices
0677 0A     2520        ASL  A         ; doubled, for ease of use
            2530 NUMOK
0678 8DC606 2540        STA  NUMVCS    ; as number of voices
```

lines, if you DIMension more variables, etc., the string may move and Playit would start playing random sounds.

The commands have simply been entered into the program via DATA statements starting at line 9000. Those of you who go to the trouble to enter all this will, I hope, be pleasantly surprised by the sounds generated by lines 9400-9418. You will probably be dismayed, however, at the idea of putting in such a complex sound yourself. That is why I encourage someone to come up with a better "Music Compiler" along these same lines.

In any case, I invite you to compose your own music or sounds to be put into this system. Generally, I wrote a sound in BASIC to test it before committing it to DATA statements. For example, the "CHOO-CHOO" sound evolved from this BASIC line:

```
FOR V=15 TO 0 STEP -1 : SOUND
O,V,O,V : NEXT V
```

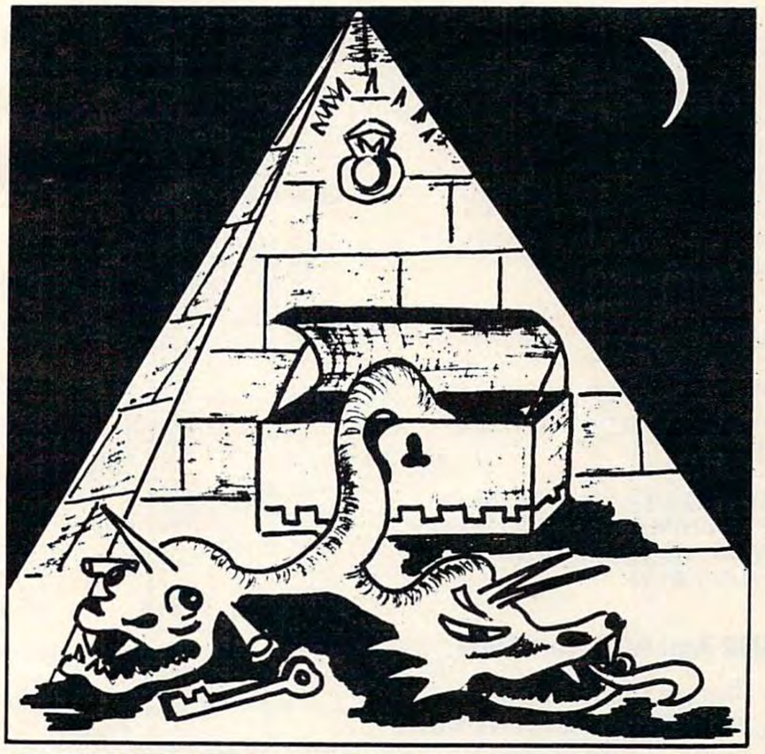
The above sounds like an explosion, but if you slow it down a little and repeat it regularly you can train it as you wish. On to the short subjects.

HexDec

If you have already peeked at the listing of Playit From BASIC, you may have noted an unusual looking hexadecimal to decimal conversion routine. In fact, I herewith present you with a "one-liner" HexDec program:

```
1 DIMH$(23),N$(9):H$=",ABCDEF
GHI!!!!!!! JKLMNO":IN.N$:F.I=
1TOLEN(N$):N=N*16+ASC(H$(ASC
(N$(I))-47))
:N.I:?N:RUN
```

The underlined characters are control characters (control-comma is the heart, etc.). The abbreviations are necessary to get it to fit on one line. To see how it works, figure out what happens when you input "9A". Recall that ASC("9") is 57 and ASC("A") is



CRYPTS OF TERROR

Beware as you enter the Crypts Of Terror. No one has survived this horror. Only your unrelenting nerve and determination will drive you deeper into the unknown.

Find what lurks in these ancient crypts!!

At last we have found an adventure with full graphics, sound and intrigue for your ATARI 400/800 computer.

• CRYPTS OF TERROR is the first adventure game that was completely designed for the Atari computers only. The graphics are the finest available using the full potential of the Atari.

ATARI ®

Atari 800/400 16K requires joysticks.
Payment: Personal Checks – allow three weeks for check to clear.
American Express, VISA, MasterCard – include all numbers on card. Please include phone number with all orders.
Orders from USA $29.95 (US funds)
Orders from Canada $39.95 (Canadian funds)
Plus $2.00 for shipping.

Ontario residents add 7% R.S.T.
Check your local computer dealer for Crypts Of Terror.
Dealer inquiries encouraged.

INHOME SOFTWARE

PH. 1-416-961-2760

1560 Yonge St.
P.O. Box 10
Toronto
Ontario Canada
M4T 1Z7

65. 57-47 is 10 and 65-47 is 18. Look at the 10th and 18th characters in H$. What is ASC("control-I")? ASC("control-J")?

You can avoid the control characters by adding the -64 shown in Playit From BASIC. Simple.

DecHex

This isn't really pertinent, but while we are on the subject of one-liners:

1DIMH$(16):H$="0123456789AB CDEF":IN.N:M=4096:F.I=1TO4:J= INT(N/M):?H$(J+1);:N=N-M*J:M= M/16:N.N:?:RUN

The USR And ADR Functions

Even though the methods of using the USR function are fairly thoroughly covered in the *Atari BASIC Reference Manual,* I find that many users are not fully aware of the real power of this function. Recall that the general syntax of this function is:

USR(addr [,expr [,expr ...]])

In other words, in addition to giving BASIC an address to call, you may pass *any number* of expressions to the assembly language routine. BASIC converts each expression to a 16-bit integer, pushes the result on the CPU stack, and cleans up by pushing on a single byte which tells the number of such expressions it pushed. (The address, which may itself be an expression, is *not* pushed and is not counted by that single byte.)

So what can we pass to assembly language? Obviously, numbers in the range of 0 to 65535. But what about characters? Conceive of

USR(addr, ASC("T"), expr) ,

where the "T" might be used as a mnemonic command to tell the routine which of several functions is desired. How about strings of characters? Recall that the three essential ingredients defining a

```
067B 4C4106         JMP  SAM
              ;
              ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              ;
              ; set tone and volume
              ;
              DOTV
067E AEC606         LDX  NUMVCS
0681 30BE           BMI  SAM         ; no voices to set
              TVLP
0683 20B706         JSR  GETCMD      ; get next byte
0686 9D01D2         STA  AUDC1,X     ; treat as t&v command
0689 CA             DEX
068A CA             DEX              ; more voices?
068B 10F6           BPL  TVLP        ; yes
068D 30B2           BMI  SAM         ; no
              ;
              ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              ;
              ;  STOP the sound (by clring all sound regs)
              ;
              DOSTOP
068F A207           LDX  #7
0691 A900           LDA  #0
              STOPLP
0693 9D00D2         STA  AUDF1,X    ;freq and vol to zero
0696 CA             DEX
0697 10FA           BPL  STOPLP
0699 30A6           BMI  SAM         ; sound stops, pgm keeps going
              ;
              ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              ;
              ; END the processing (but doesn't stop sound)
              ;
              DOEND
069B ADC406         LDA  SAVEVBLK
069E 8D2402         STA  VVBLKD     ; restore system ptr
06A1 ADC506         LDA  SAVEVBLK+1
06A4 8D2502         STA  VVBLKD+1   ; and, to OS, we aren't here
06A7 6CC406         JMP  (SAVEVBLK) ; one last time
              ;
              ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              ;

              ; repeat the same stuff again
              ;
              DORPT
06AA ADC206         LDA  REPEAT
06AD 85CE           STA  PLAYADDR
06AF ADC306         LDA  REPEAT+1
06B2 85CF           STA  PLAYADDR+1 ; just reset the address
06B4 4C4106         JMP  SAM         ; and try it again

06B7                .PAGE "     the GETCMD subroutine"
              ;
              ; simply gets next byte from
              ; command string
              ;
              GETCMD
06B7 A000           LDY  #0
06B9 B1CE           LDA  (PLAYADDR),Y ; get the byte
06BB E6CE           INC  PLAYADDR   ; bump LSB of pointer
06BD D002           BNE  GCEXIT     ; done
06BF E6CF           INC  PLAYADDR+1 ; and the MSB
              GCEXIT
06C1 60             RTS
              ;

06C2                .PAGE "     ram usage"
              ;
06C2 0000     REPEAT .WORD 0         ; in case we hear it again
06C4 0000     SAVEVBLK .WORD 0       ; so we can jmp indirect
06C6 00       NUMVCS .BYTE 0         ; controls TVLP and FREQLP
06C7 00       DURATION .BYTE 0       ; how long we hold a sound
              ;
              ;
06C8                .END

=0600 ORIGIN       =00FF LOW          =0100 HIGH         =D200 AUDF1
=D201 AUDC1        =0224 VVBLKD       =0014 CLOCKLSB     =00CE PLAYADDR
=00FF CMDR         =00FE CMDS         =00FD CMDN         =00FC CMDTV
=0000 CMDE          0600 INSTALL       0603 GOOF         =0608 INSTALL1
 060A WAITVB        06C2 REPEAT        063C PLAYIT        0626 NOWINSTALL
 0636 INSTALLED     06C4 SAVEVBLK      06C7 DURATION      066A EXIT
 0641 SAM           06B7 GETCMD        069B DOEND         06AA DORPT
 068F DOSTOP        067E DOTV          066D DONUM         0658 DODURATION
 06C6 NUMVCS        0660 FREQLP        0678 NUMOK         0683 TVLP
 0693 STOPLP        06C1 GCEXIT
```

Imagine being able to print the letter "A" and get a multi-color space ship. Using THE NEXT STEP and a minimum of programming effort, you can do it in no time at all.

THE NEXT STEP contains well-written, easy-to-use documentation with simple BASIC programming examples that show you how THE NEXT STEP can help develop colorful graphic displays. Graphics you never thought possible until now.

THE NEXT STEP is a user friendly, menu driven graphics tool kit that allows you to create new character sets or redefine characters to make shapes for use with your basic or machine language programs. THE NEXT STEP allows you to save these "new" characters on disk for future use.

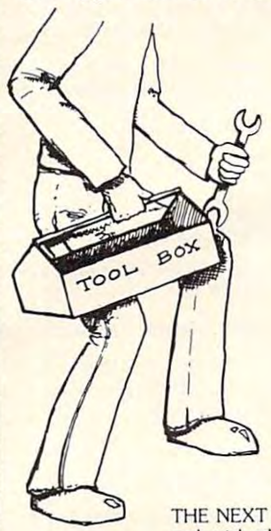
THE NEXT STEP is perfect for use on shapes for animation and features a joystick controlled color menu to make your graphics come alive. THE NEXT STEP even generates its own code to help you incorporate new characters and shapes into your programs.

THE NEXT STEP allows you to see your shapes as you make them. Now you can determine ahead of time how characters will interact with one another when creating shapes for Character Set or Player-Missile Graphics.

THE NEXT STEP helps you to mix any of ATARI's 14 graphics modes in the same display. THE NEXT STEP is a perfect graphics utility for the BASIC or machine language programmer-novice and professional alike.

THE NEXT STEP features full joystick control for ease-of-use and quick editing.

THE NEXT STEP runs on any 32K ATARI 400/800 with a disk drive and is available for $39.95 at your local computer store or order direct from . . . . .



THE NEXT STEP

ON-LINE systems

VISA, MASTERCARD, CHECK, C.O.D.

Add $1.00 for Shipping

36575 MUDGE RANCH ROAD • COARSEGOLD, CA 93614 • 209-683-6858

string in Atari BASIC are its DIMension, LENgth, and address. Since your program presumably DIMensioned the string, you know that value and may pass it as an expression. And the address and length are available from the ADR and LEN functions!

Would you like your assembly language routine to modify your string, affecting its length? Try something like this:

```
DIM XX$( XXDIM )
XX$( USR( addr, ADR(XX$), XXDIM ) + 1 ) = " "
```

Recall that the USR function may return any 16-bit value to the BASIC program, which is automatically converted to floating point as needed. Assume that this USR routine puts something in the XX$ string and returns the number of characters it put in. The above will then set the LENgth of XX$ properly for use by other BASIC statements and functions.

Finally, there is floating point. How about writing a matrix inversion program? If we are limited to passing 16-bit integers, how do we pass a floating point number via USR? Simple: we pass the address of the number, just as we do with a string. And how do we get the address of a number, when the ADR function only works with strings? Like this:

```
DIM FF$(1),FF( dim1, dim2 )
JUNK = USR( addr, ADR(FF$) + 1, dim1, dim2 )
```

A little published fact about Atari BASIC is that DIMensioning of both strings and arrays proceeds in an orderly fashion according to the DIM statements encountered. And you are guaranteed that the order you DIM strings and arrays is the order they will occur in memory! So, by DIMensioning that one-byte string, FF$, directly before the DIMension of the array, FF( ), we *know* that the address of the array is one greater than the address of the string. Thus we can pass all the pertinent information about the array (its address and dimensions) to our assembly language routine. Incidentally, if you don't want to waste a one-byte string for this purpose, there is no reason FF$ can't be any DIMension you need: just adjust the '+ 1' to reflect the actual DIM you use.

One last note on this subject: the fact that you can predict the memory order of strings and arrays has fascinating possibilities in regards to record structures, etc. But (and how many times have you read this from me) that's a topic for another article.

Program 1.

```
10 AUDCTL=53768:DBL=120
20 AUDF1=53760:AUDC1=53761
30 SOUND 1,10,10,15:SOUND 3,10,10,15
40 POKE AUDC1,0:POKE AUDC1+4,0
50 POKE AUDCTL,DBL
60 FOR J=10 TO 15:POKE AUDF1+2,J:POKE AUDF1
   +6,20-J
```

```
70 FOR I=0 TO 255:POKE AUDF1,I:POKE AUDF1+4
 ,255-I:NEXT I
80 NEXT J

        ...VERY SMOOTH GLIDES...
```

Program 2.

```
10 AUDCTL=53768:DBL=120
12 OSC=1789790/2
20 AUDF1=53760:AUDC1=53761
30 SOUND 1,10,10,0
40 POKE AUDC1,0:POKE AUDC1+4,0
50 POKE AUDCTL,DBL
60 P2=2^(1/12)
70 NTE=16:REM C IN THE REAL BASS
80 FOR I=1 TO 109
90 FREQ=INT(OSC/NTE-7+0.5):F0=INT(FREQ/256)
92 F1=FREQ-256*F0
100 POKE AUDF1,F1:POKE AUDF1+2,F0
102 POKE AUDC1+2,175
103 PRINT "NOW PLAYING ";INT(NTE+0.5);" HZ"
105 FOR J=1 TO 100:NEXT J
110 NTE=NTE*P2
120 NEXT I
130 GOTO 70

      ...9 OCTAVE CHROMATIC SCALE...
```

Playit From BASIC

```
1000 REM ******************************
1020 REM *
1040 REM * PLAYIT FROM BASIC, SAM
1060 REM *
1080 REM * This routine is a simple
1100 REM * sound "compiler", which
1120 REM * takes DATA statements and
1140 REM * converts them into command
1160 REM * strings suitable for use by
1180 REM * the interrupt-driven PLAYIT
1200 REM * routine.
1220 REM *
1240 REM *
1260 REM * Written by Bill Wilkinson
1280 REM *
1300 REM *   for March, 1982, COMPUTE!
1320 REM *
1340 REM ******************************
1360 REM
1380 REM First, constants, routine addresses, etc.
1400 REM
1420 DIM HX$(2),CMD$(11),PLAY$(1000),HEX$(23),TYPE$(1),
     PLAYIT$(1000)
1440 HEX$="@ABCDEFGHI!!!!!!!JKLMNO"
1460 DOCMD=2300:LOOP=1800:HEXDEC=2600
1480 AGAIN=1700:EXITLOOP=2100
1500 PLAYIT=6*256:REM or wherever you put the routine
1520 REM
1530 SOUND 0,0,0,0:REM needed to initialize properly
1540 REM The command equates...
1560 REM notice that these match the
1580 REM assembly language routine
1600 CMDR=255:CMDS=254:CMDN=253:CMDTV=252:CMDE=0
1620 REM
1640 REM *****************************
1660 REM
1680 REM This is the AGAIN of
1700 REM PLAY IT AGAIN, ATARI
1720 REM
1730 PRINT "    <processing...please wait>"
1740 PLAY$="":PLAY=0
1760 REM
1780 REM This is LOOP
1800 PLAY=PLAY+1:REM to next cmd byte
1820 READ CMD$:REM a bunch of commands
1840 REM
1860 TYPE$=CMD$:REM use the command character
1880 IF TYPE$="R" THEN PLAY$(PLAY)=CHR$(CMDR):GOTO EXIT
     LOOP
1900 IF TYPE$="S" THEN PLAY$(PLAY)=CHR$(CMDS):GOTO LOOP
1920 IF TYPE$="N" THEN NUMVCS=1:CMD=CMDN:GOSUB DOCMD:NU
     MVCS=DEC:GOTO LOOP
1940 IF TYPE$="T" THEN CMD=CMDTV:GOSUB DOCMD:GOTO LOOP
1960 IF TYPE$="E" THEN PLAY$(PLAY)=CHR$(CMDE):GOTO EXIT
     LOOP
1980 REM *** IF TO HERE, ASSUME DURATION & FREQ ***
2000 HX$=CMD$:GOSUB HEXDEC:CMD=DEC:REM command is
     duration
2020 CMD$=CMD$(2):REM to fool DOCMD
2040 GOSUB DOCMD:GOTO LOOP
2060 REM
2080 REM exitloop
2100 REM
2120 REM do the sound playing
2140 REM
2150 PLAYIT$=PLAY$:REM else we alter what we are playing
2160 JUNK=USR(PLAYIT,ADR(PLAYIT$))
2180 REM
2200 PRINT "HIT RETURN FOR NEXT SOUND ";:INPUT TYPE$
2220 GOTO AGAIN
2240 REM
2260 REM
2280 REM ***************************
2300 REM THE SUBROUTINES
2320 REM
2340 REM first, DOCMD
2360 REM
2380 PLAY$(PLAY)=CHR$(CMD):REM The command byte
2400 IF NUMVCS=0 THEN RETURN
2420 REM we process NUMVCS bytes
2440 FOR I=2 TO NUMVCS+NUMVCS STEP 2
2460 HX$=CMD$(I):GOSUB HEXDEC:REM convert the byte
2480 PLAY=PLAY+1:PLAY$(PLAY)=CHR$(DEC):REM and stuff it
     away
2500 NEXT I
2520 RETURN
2540 REM
2560 REM ............
2580 REM ****************************
2600 REM and now HEXDEC
2620 REM
2640 DEC=0:REM our accumulator
2660 FOR L=1 TO LEN(HX$)
2680 DEC=DEC*16+ASC(HEX$(ASC(HX$(L))-47))-64
2700 NEXT L
2720 RETURN
8999 REM ...a siren-like sound...
9000 DATA N01,TCF,1408,1412,R
9099 REM ...a fanfare of sorts...
9100 DATA S,N01,TA2,30F3
9102 DATA N02,TA3A3,30F3C1
9104 DATA N03,TA4A4A4,30F3C1A1
9106 DATA N04,TA5A5A5A5,60F3C1A17A
9108 DATA T00000000
9110 DATA N00,C0,R
9199 REM ...beeping off the seconds...
9200 DATA S,N01
9202 DATA TAE,0130
9204 DATA TAC,0130
9206 DATA TAA,0130
9208 DATA TA8,0130
9210 DATA TA6,0130
9212 DATA TA4,0130
9214 DATA TA2,0130
9216 DATA T00,3500
9218 DATA R
9299 REM ...choo-choo ??? ...
9300 DATA S,N01
9302 DATA T0E,010E
9304 DATA T0C,010C
9306 DATA T0A,010A
9308 DATA T08,0108
9310 DATA T06,0106
9312 DATA T04,0104
9314 DATA T02,0102
9316 DATA T00,0300
9318 DATA R
9400 DATA S,N01,TAC
9402 DATA 3051,305B,3044,183C,182D,3035
9404 DATA [illegible]3C,182D,3035,3044,303C,3051,305B
9406 DATA N04,TACA4A4A8
9408 DATA 30516C89A2,305B7990B6,30446C89A2
9410 DATA 183C4879B6,182D4879B6,3035485BD7
9412 DATA 183C4879B6,182D5BB6B6,3035445B89
9414 DATA 3044516CA2,38325179F3
9416 DATA 423C485BB6,50445B6C89
9418 DATA S,N00,F0,R
9898 REM ...stop and end...to quit...
9999 DATA S,E
```

©

PART I

Disk Checkout For 2040, 4040, And 8050 Disks

Jim Butterfield
Toronto, Canada

Editor's Note: In Part I of this article Jim explains disk manipulations via machine language. Next month, in Part II, he concludes with a machine language disk routine and a program that can analyze the condition of files and blocks on the disk. — RTM

The disk doesn't know or care who's giving it instructions: BASIC or Machine Language. All that's needed is to send or receive the same information as BASIC uses.

For all input and output, I recommend opening the necessary channels from BASIC. It's easier and works the same in all systems. Machine language may then take over and use the previously opened files as it wishes, connecting and disconnecting at will.

You'll often want to check the status byte ST. It's located at hexadecimal 96 in PET's memory. It's especially important for checking end-of-file on sequential records and end-of-record on relative records. You can also detect IEEE problems here, especially timeouts.

Let's take a simple example. We might want to do a Block Read of a given track and sector from disk and then dump part of the contents to the screen. To make our example easy, we'll display only bytes one through eight. Byte zero is sometimes hard to get on early disk systems due to a bug in the Buffer-Pointer routine; we'll sidestep that question.

The BASIC Program

We're planning to read bytes one through eight of track 18, sector 0. That might be the BAM (Block Availability Map) block, but perhaps not: these programs will also work on 8050 disks.

We must: Open the Command channel, secondary address 15; Initialize the disk, in case it's a 2040; Open a direct access channel; Cause the block read; Set the Buffer pointer; and, finally, read the channel. At the finish we should close our channels. Our BASIC program would read:

```
100 OPEN 6,8,15                    (Command Channel)
110 PRINT#6,"I0"                   (Initialize)
120 OPEN 2,8,3,"#"                 (Direct Access
                                   channel)
130 PRINT#6,"U1:";3;0;18;0         (Read Block)
140 PRINT#6,"B-P:";3;1             (Set Buffer Pointer)
150 GET#2,X$                       (Get a byte)
160 PRINT ASC(X$+CHR$(0) );        (Print it)
170 C=C+1                          (Count them)
180 IF C<8 GOTO 150                (Do more?)
190 CLOSE 2:CLOSE 6                (Quit)
```

You might like to try this to see it work. If you like, change the buffer pointer (line 140), the number of values displayed (line 180) or the track and sector (line 130). Now let's try the same thing in machine language.

The BASIC Driver

It's convenient to OPEN from BASIC, so we type NEW and enter the following BASIC program which will set things up for Machine Language:

```
100 OPEN 6,8,15
110 PRINT#6,"I0"
120 OPEN 2,8,3,"#"
125 SYS 1200
190 CLOSE 2:CLOSE 6
```

Don't run this yet, since the Machine Language is not in place.

Planning The Machine Language Program

We want to send exactly the same stuff as was sent by BASIC, to the same logical channels. We know that the ML equivalent of PRINT#6... is LDX #$06, JSR $FFC9 ... JSR $FFCC. Note that we use the logical file number, 6. Similarly, we know the equivalent of GET#2 is: LDX #$02, JSR $FFC6, JSR $FFE4,... JSR $FFCC. So we can code:

```
LDX   #$06
JSR   $FFC9    (Open channel 6)
LDA   #$55     (Letter U)
JSR   $FFD2    (.. print it)
LDA   #$31     (Digit 1)
JSR   $FFD2    (.. print it)
LDA   #$3A     (Colon)
JSR   $FFD2
LDA   #$20     (Space)
JSR   $FFD2
LDA   #$33     (Digit 3)
JSR   $FFD2
LDA   #$20     (Space)
JSR   $FFD2
LDA   #$30     (Digit 0)
JSR   $FFD2
LDA   #$20     (Space)
```

```
JSR    $FFD2
LDA    #$31      (Digit 1)
JSR    $FFD2
LDA    #$38      (Digit 8)
JSR    $FFD2
LDA    #$20      (Space)
JSR    $FFD2
LDA    #$30      (Digit 0)
JSR    $FFD2
LDA    #$0D      (Return)
JSR    $FFD2
JSR    $FFCC     (End transmission)
```

Note that we are sending exactly what BASIC sent from line 130. Most programmers would quickly realize that a program loop would save a good deal of memory here. In Part II of this article, we'll rewrite the code and complete it.

COMPUTE!
The Resource.

SWEET-TALKER, IT GIVES YOUR COMPUTER AN UNLIMITED VOCABULARY.

As Featured in Ciarcia's Circuit Cellar, Byte Magazine, September 1981.

The Sweet-Talker voice synthesizer allows you to add speech of unlimited vocabulary to your computer. Utilizing the Votrax SC-01A chip, you can output any message by programming individual phonemes. Comes in two versions; one plugs directly into your Apple II, the other connects to any computer with an 8-bit parallel printer port. + 12 volts and + 5 volts required for parallel board.

- Contains 64 different phonemes accessed by a 6-bit code.
- Automatic and manual inflection modes.
- Parallel port driven or plug-in compatible with Apple II.
- Super text-to-speech algorithm on disk for Apple II. Makes Sweet-Talker equivalent to units 3 times the cost.
- On board audio amplifier
- Sample program on cassette with Apple II board.
- Optional power supply for parallel board.

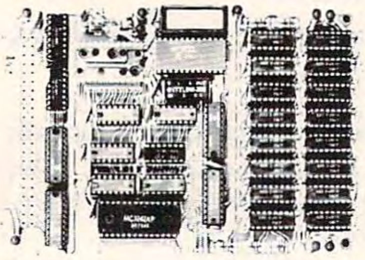
ST01	**Sweet Talker Parallel Port Board A & T**	**$139.00**
ST02	**Sweet Talker Apple II plug in board**	**149.00**
ST06	**Text-to-Speech algorithm on disk for Apple II**	**35.00**
SC01A	**Votrax Speech Synthesizer chip**	**70.00**
	5 or more	**55.00 each**
UPS01	**Universal Power Supply- A & T**	**35.00**

Add $2.00 for shipping & handling.

To Order: Call Toll Free 1-800-645-3479
(In N.Y. State Call: 1-516-374-6793)
For Information Call: 1-516-374-6793

MICROMINT INC.
917 Midway
Woodmere N.Y. 11598

MICRO MINT



BETA 32K BYTE EXPANDABLE RAM FOR 6502 AND 6800 SYSTEMS
AIM 65 KIM SYM PET S44-BUS

- Plug compatible with the AIM-65/SYM expansion connector by using a right angle connector (supplied).
- Memory board edge connector plugs into the 6800 S44 bus.
- Connects to PET using an adaptor cable.
- Uses +5V only, supplied from the host computer.
- Full documentation. Assembled and tested boards are guaranteed for one full year. Purchase price is fully refundable if board is returned undamaged within 14 days.

Assembled	with 32K RAM	$349.00
& Tested	with 16K RAM	329.00
Bare board, manual & hard-to-get parts		99.00
PET interface kit. Connects the 32K RAM board to a 4K or 8K PET		$ 69.00

See our full-page ad in BYTE and INTERFACE AGE

wabash

8" or 5¼" flexible diskettes certified 100% error free with manufacturer's 5-year limited warranty on all 8" media. Soft-sectored in boxes of 10. 5¼" available in 10-sector.

(Add $3.00 for plastic library cases)

8" single sided, single density	$27.50
8" single sided, double density	35.50
8" double sided, double density	45.50
5¼" single sided, single density	27.50
5¼" single sided, double density	29.50
5¼" single sided, double density, 10-sector	$29.50

TERMS: Minimum order $15.00. Minimum shipping and handling $3.00. Calif. residents add 6% sales tax. Cash, checks, Mastercard, Visa and purchase orders from qualified firms are accepted. (Please allow two weeks for personal checks to clear before shipment.) Product availability and pricing subject to change without notice.

INTERNATIONAL ORDERS: *Add 15% to purchase price for all orders. Minimum shipping charge is $20.00. Orders with insufficient funds will be delayed. Excess funds will be returned with your order. All prices are U.S. only.*

PAPER TIGER PRINTERS

IDS 460G 9x9 Dot Matrix Printer	$890.00
IDS 560G Wide Carriage Printer	1099.00

TERMINALS

ADDS Viewpoint	$569.00
TeleVideo 910	579.00
TeleVideo 912C	679.00
TeleVideo 920C	729.00
TeleVideo 950	929.00

8" DISK DRIVES

Shugart 801R	$399.00
NEC FD1160 (double sided)	569.00

DYNAMIC RAMS

4116 (200ns)	set of 8	$24.00
4164 (64Kx1)		$18.00

BETA COMPUTER DEVICES
1230 W. COLLINS AVE.
ORANGE, CA 92668
(714) 633-7280

Visit us at the West Coast Computer Faire-Booth 1714-C.

Organizing Data Storage

John Hudson
Los Angeles, CA

There are many storage media available to minicomputer users. Minicomputer users with a disk unit know that the disk unit enhances the storage and retrieval powers of their minicomputer. One type of file that can be created for the purposes of storage and retrieval is a text file (for storage of such things as mailing addresses, telephone numbers, receipts, etc.).

For small text files, the time involved in disk retrieval and storage is not a problem. However, when a text file becomes larger than 2,000 records, the retrieval and storage of information can become time consuming.

Large text files can be organized in one of two ways: sequentially, and randomly. In sequentially organized text files, fields are stored back to back, where the beginning character of a new field immediately follows the return character ending the previous field. Information is retrieved in a linear fashion, i.e., from the beginning to the end of the file.

Disk Can Also Be Slow

When a text file does not require much updating or ongoing revision, sequential organization of the text file is indicated. However, if a large text file is ordered sequentially, and there is need for frequent updating or revision of the file, or frequent retrieving of information from end of text file, a disk unit is not much better than a cassette unit. This accessing of information at end of file may take a couple of minutes, due to the reading and verification of each record, each time.

In this type of situation, the random method of text file organization is more effective. A random-access text file is like a collection of equally-sized records; the records may be full, or they may be empty, but the length of each record in a random text file is fixed. Thus, a record at the end of the file can be accessed at approximately the same speed as records in any other location in the file.

However, the controlling program needs to know where in the file a specific record is located. Most random files are organized by 'keying' a field within the record. For example, a mailing address text file can be organized by last names. The problem when using a random text file keyed to a specific field in the record is *collision*. Collision is when two or more records address the same location within the text file, as, for example, when two people have the same last name (B. JONES and J. JONES).

A method of reducing collision is called *hashing* the key field. The basic idea of hashing, or hash addressing, is that each stored record occurrence is placed in the text file at a location whose address may be computed as some function (the hash function) of a value which appears in the occurrence – usually the primary key value.

One of the disadvantages of hash-addressing is that the sequence of stored record occurrences within the text file will almost certainly not be the keyed field sequence. In addition, there may be gaps of arbitrary size between consecutive occurrences of records.

In fact, a text file in a hash-addressing organization is usually, though not invariably, considered to have no particular sequence.

Using Mod To Hash

The following is an example of a hash function: given that the number of unique records is 1,000; the "mod" arithmetic function can be used to assign unique address locations. The mod function divides one number by another and returns the remainder. The mod parameter used in this function should be the prime number closest to the number of the records in the text file (see Table 1 for prime numbers). For this example, the closest prime number is 997. (Note: if the key field is alphabetic, it should be converted to numeric.) The function will be (key field) MOD 997. The hash function thus minimizes collision.

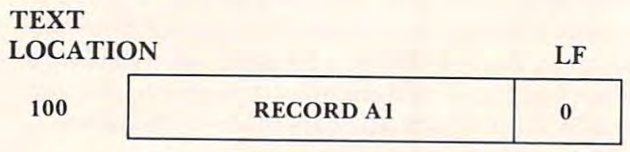
There are text files, such as a monthly inventory file, that require multiple entries of the same record over a period of time. Inventory may be taken at the end of each week, and the quantity stored into a text file. This presents a different type of collision problem – same record hash to same location in text file.

In the case where hashing records into a text file still causes collision, the controlling program needs to be able to insert the colliding record into another location and, when it goes to retrieve this record, it needs to know where it is located. A solution to this problem is to link the records in the text file. From the previous example, you have 1,000 unique records; in addition, each record is entered more than once.

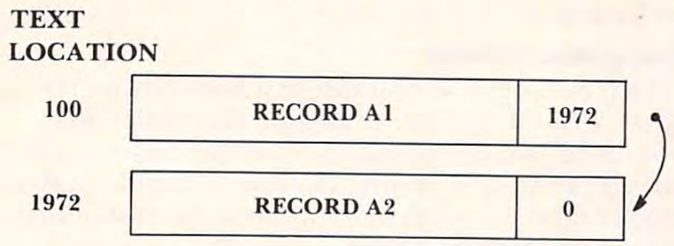
A link field (LF) can be added to the end of each record to allow the linking of records. For example:

RECORD	LINK FIELD

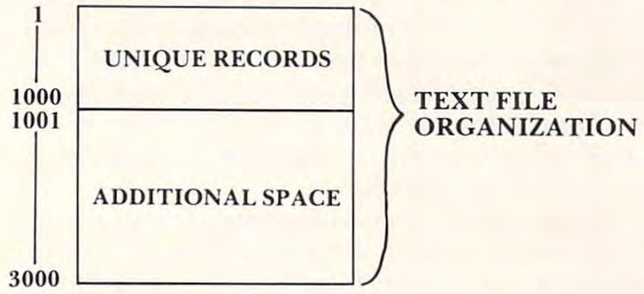
This LF is used to point to successive entries of the same type of record, and contains the address locations of the successive record entries. The first record, A1, hashed into the text file at location 100 has '0' in the link field.



When the controlling program tries to hash another record, A2, into record location 100, it notes that there already exists a record at that location, and inserts the new record, A2, at another text address. It changes the LF of the record A1 from 0 to the next text address of record A2 (in this case, 1972), inserts 0 into the LF of record A2, and the results are as follows:



Thus, in this example, record A1 points to record A2, However, a problem arises with this type of organization: how to set up the text file? The text file can be organized with 1,000 unique hashing locations, occupying text address locations 1-1000. Any additions to a unique record can be located at text address locations 1001-3000.



This type of text file organization needs to be initialized, since the Apple system does not allow reading of a text file that does not contain records, and will produce an "END OF DATA" error message. An example of an initialization routine follows:

```
5  D$=" "
10 DLOC=66:DDTE=9999:DBS=1:DSN=2:DLP
   =333:DTRK=444:DCAST=555:DLINK=8888
11 PRINT D$; "OPEN RECORD,L29"
20 I=2001
30 PRINT D$; "WRITE RECORD,R0"
40 PRINT I: PRINT DDTE: PRINT DBS: PRINT
   DSN: PRINT DLP: PRINT DTRK: PRINT
       DCAST: PRINT DLOC:
1001 FOR J=1 TO 4200
1006 PRINT D$; "WRITE RECORD,R";J
1007 PRINT DLOC: PRINT DDTE: PRINT DBS:
     PRINT DSN: PRINT DLP: PRINT DTRK:
     PRINT DCAST: PRINT DLINK:
1009 NEXT J
1010 PRINT D$;"CLOSE RECORD"
1013 END
```

This routine initializes enough space for 4,200 records of length 29. It writes into every record a set of dummy values.

When you wish to insert a record into the main text area, the controlling program will read the text address and check a specific field for 9999, (DDTE). If it finds 9999, the controlling program can insert the record into the read text location. If it does not, then it will insert the record into the additional text area. After inserting the record, the LF of the main record is updated to point to the location of the additional record(s).

A method of keepinng track of available space in the additional text area is to store this address location and length of records into address location 0 of the text file. After each "additional text area" insertion, the available address is incremented. At the start, the controlling program will read this information, update it as needed, and, upon completion of the program, will rewrite the record 0 with the new address location.

The following is an example of a program using the link organization of a text field:

Line 70 reads text location 0 to determine the next available additional space, which is indicated by the variable "FREESPACE."

Lines 120 through 140 determine the location where the new record will be inserted. Note that this is not a hashing function.

Line 190 checks to see if the text field location DDTE has the dummy value of 9999, or if it is filled.

Lines 191 through 200 insert the new record into the unique text space.

Lines 212 through 214 traverse the link lists to get to the last record in the link.

Lines 220 through 260 update the last record in the link, and insert the new record into the additional text space area.

Lines 280 through 290 update record 0 when the program is completed.

2	3			
5	7	11	13	17
19	23	29	31	37
41	43	47	53	59
61	67	71	73	79
83	89	97	101	103
107	109	113	127	131

137	139	149	151	157
163	167	173	179	181
191	193	197	199	211
223	227	229	233	239
241	251	257	263	269
271	277	281	283	293
307	311	313	317	331
337	347	349	353	359
367	373	379	383	389
397	401	409	419	421
431	433	439	443	449
457	461	463	467	479
487	491	499	503	509
521	523	541	547	557
563	569	571	577	587
593	599	601	607	613
617	619	631	641	643
647	653	659	661	673
677	683	691	701	709
719	727	733	739	743
751	757	761	769	773
787	797	809	811	821
823	827	829	839	853
857	859	863	877	881
883	887	907	911	919
929	937	941	947	953
967	971	977	983	991
997	1009	1013	1019	1021

```
10 INPUT "PLEASE ENTER STORE
   NUMBER ", SN
11 CALL - 936: FOR X = 1 TO 9
   : CALL - 922: NEXT X
15 PRINT:PRINT"      I N S E R
   T     D I S K    ";SN
16 FOR X=1 TO 3000: NEXT X
17 CALL -936
20 INPUT "PLEASE ENTER DATE .
   . MMDD .. ",DTE
30 INPUT "PLEASE ENTER PURCHA
   SE OR SELL 1 = PURCHASE ..
   2 = ~ SELLS ",BS
40 D$=""
50 PRINT D$;"OPEN RECORD,L29"
60 PRINT D$;"READ RECORD,R0"
70 INPUT FREESPACE: INPUT DDT
   E: IN PUT DBS: INPUT DSN:
   INPUT ~ DLP: INPUT DTRK
71 INPUT DCAST: INPUT DLINK
80 IF FREESPACE>=5000 THEN GO
   TO 320
90 PRINT D$;"CLOSE RECORD"
100 INPUT "PLEASE ENTER RECOR
    D CODE,LPS,TRK8S,CASETTES
    ",LOC, LP,TRK,CAST
110 IF LOC=9999 THEN GOTO 270
120 LOCA=LOC/100
130 LOCA=LOC-LOCA*100
140 LOC=LOC/100:LINK=0
150 PRINT D$;"OPEN RECORD,L29"
160 PRINT D$;"READ RECORD,R";
    LOC
170 INPUT DLOCA: INPUT DDTE:
    INPUT ~ DBS: INPUT DSN:
    INPUT DLP: INPUT DTRK
171 INPUT DCAST: INPUT DLINK
190 IF DDTE#9999 THEN GOTO 212
191 PRINT D$;"WRITE RECORD,R";LOC
200 PRINT LOCA: PRINT DTE: PRINT BS
    : PRINT SN: PRINT LP: PRIN
    T TRK: PRINT CAST: PRINT L
    INK
210 GOTO 90
212 IF DLINK=0 THEN GOTO 220:LOC=DL
    INK
213 PRINT D$;"READ RECORD,R";DLINK
214 GOTO 170
220 PRINT D$;"WRITE RECORD,R";LOC
225 DLINK=FREESPACE
230 PRINT DLOCA: PRINT DDTE: PRINT ~
    DBS: PRINT DSN: PRINT DLP:
     PRINT DTRK: PRINT DCAST: ~
    PRINT DLINK
240 FREESPACE=FREESPACE+1
250 PRINT D$;"WRITE RECORD,R";DLINK

260 GOTO 200
270 PRINT D$;"WRITE RECORD,R0"
280 PRINT FREESPACE: PRINT DDTE: PR
    INT DBS: PRINT DSN: PRINT ~
    DLP: PRINT DTRK: PRINT DCA
    ST: PRINT DLOCA
290 PRINT D$;"CLOSE RECORD"
300 INPUT  "DO YOU WISH TO CONTINUE
     .. Y/N ",K$
310 IF K$="Y" THEN GOTO 10
320 END
```

©

Software Breakthrough!

THE WORD HANDLER

BY SILICON VALLEY

THE ONLY WORD PROCESSOR IN THE WORLD FOR THE APPLE THAT:

- **Gives you full line capability on the screen, no boards nec.**
- **Two character formats to choose from at all times!**

RUNS WITHOUT ANY HARDWARE MODIFICATIONS!

HIGH POWER
Simplicity of Operation!

LARGE SCALE
Direct to disk!

All functions seen on the screen!

- Lower and uppper case underlining
- Superscript, bold, and unlimited tabs
- Proportional spacing
- Even and normal justification
- Other functions seen on turnkey systems in the $13–$20,000 range

DEALER INQUIRIES INVITED

1625 EL CAMINO REAL, SUITE 4
BELMONT,CA 94002
(415) 593-4344

Machine Language Sort Utility

Ronald and Lynn Marcuse
Freehold, NJ

There have been occasional articles in the various personal computer magazines concerning the sorting of data files. Some of these have presented sort routines coded in BASIC that can be utilized by existing programs. The complex string handling required by the sort logic is not really suitable for BASIC's rather slow execution speed. Clearly, any type of repetitive string manipulations (as performed by sorting or searching functions) would benefit from machine language code. If you continue reading you will find out how much faster it really is.

Before we get into the programs themselves, it would probably be beneficial to include some background information. The verb *sort* is defined: "to put in a certain place or rank according to kind, class or nature; to arrange according to characteristics." This comes pretty close to what we sometimes want to do with the data we store in our computers and files; put it in some kind of order. Once we have arranged it we can search it quicker (imagine a disorganized phone book), list it in a more readable format, or even match it to other files that have been sorted the same way.

The Main Questions

First we must decide where will we do the actual sorting. All of us have arranged things on a desk or table. Our sort area is, therefore, the desk or table that we used. In a computer system we have a choice of using the memory within the machine (internal) or our disk drive (external). There are problems with both of these. Computer memory is limited in size and this, in turn, will limit the number of records that can be read in. The disk drive may be able to hold more data, but the speed of the device is snail-like when compared to memory. We could use both: divide the file up into smaller chunks which can be sorted in memory, store these on disk as temporary files, and then merge all of them together. This process is usually referred to as "sub-listing" or "sort-merge."

The next question involves the type of sort logic (there are many ways of putting things in order). The algorithm used here is called a *bubble* sort. The file or list is examined two records at a time. If the second has a lower sort key than the first, the two will exchange places within the file. Why then, you ask, is it called a bubble sort. Because records appear to "bubble" upward in memory (I didn't coin the phrase so don't blame me). Although this is not a very exotic methodology, it does offer several advantages. It requires no other memory allocations for sorting and is fast if the file is not too disorganized. It will also not disturb the relative positioning of records that have equal sort keys.

There are numerous other types of sort algorithms. A *selection* sort would go through a list of (n) items (n-1) times, pulling out the next lowest record and adding it to the current end of a new list. This would need double the memory, though. A *selection and exchange* would perform a similar function within the main sort area, selecting the lowest element during each pass, moving it upward in the list to be exchanged with the element occupying its new position . This method tends to upset the existing relative positioning. Other types involve binary tree searches and more complex algorithms.

Why Machine Language

The choice of language is, as stated above, rather clear. Unless you have a lot of time to kill, your sort must be in executable object code (machine language). When you're doing several hundred thousand (or million ?) character comparisons and swaps, you don't have time to pull out a "BASIC dictionary" for each line in the program (this, in essence, is what the BASIC does).

Here are some representative execution times, based on some testing we did last winter. The speeds are approximate and do not include disk input/output time. The test file consisted of 200 records, each 75 characters in length. The sort key occupied ten positions:

BASIC selection/exchange sort (in memory) – 8 minutes

BASIC bubble sort (in memory) – 12 minutes

BASIC selection sort (on disk) – 2 HOURS plus (hit BREAK key)

Machine Language bubble (memory) – 3 seconds

The sort program was developed with flexibility in mind. It will sort fixed length records up 150 bytes in size. The sort key itself may be located anywhere in the record and can be any length (up to the size of the record). It will sort in either ascending or descending order. The records themselves must be comprised of ASCII (ATASCII) characters. While in memory, they need not be terminated by end-of-line ($9B) characters.

ATARI* SOFTWARE at DISCOUNT PRICES from

COMPUTER HOUSE

	reg. retail	OUR PRICE
ADVENTURE INTERNATIONAL		
Adventures 1 thru 12 each (T)	19.95	15.95
Star Trek 3.5 (T)	19.95	15.95
Lunar Lander (T)	19.95	15.95
Galactic Empire (T)	19.95	15.95
ATARI		
Assembler Editor (C)	59.95	47.95
Asteroids (C)	39.95	31.95
Atari Word Processor (D)	150.00	115.00
Conversational Spanish (T)	59.95	47.95
Missile Command (C)	39.95	31.95
Scram (T)	19.95	15.95
Star Raiders (C)	49.95	39.95
AUTOMATED SIMULATIONS		
Invasion Orion (T or D)	24.95	19.95
Rescue at Riget (T or D)	29.95	23.95
Star Warrior (T or D)	39.95	31.95
AVALON HILL		
Conflict 2500 (T)	15.00	12.00
Empire of the Overmind (T)	30.00	24.00
Tanktics (T)	24.00	19.20
Midway Campaign (T)	15.00	12.00
C E SOFTWARE		
CCA Data Management (D)	99.95	79.95
CODE WORKS		
Iridis #1 (T)	9.95	7.95
Iridis #1 (D)	12.95	10.35
Iridis #2 (T)	15.95	12.75
Iridis #2 (D)	18.95	15.15
CRYSTAL COMPUTER		
Crystaloids (D)	29.95	23.95
Glamis Castle (D)	49.95	39.95
Fantasyland 2040 A.D. (D)	59.95	47.95
Protector (D)	39.95	31.95
Sands of Mars (D)	39.95	31.95
DATASOFT		
Le Stick	39.95	31.95
Lisp Interpreter (D)	149.95	119.95
Text Wizard (D)	99.95	79.95
DYNACOMP		
Flight Simulator (T)	17.95	14.35
Rings of the Empire (T)	16.95	13.55
ON-LINE SYSTEMS		
Jawbreaker (D)	29.95	23.95
Softporn Adventure (D)	29.95	23.95
Wiz and Princess (D)	32.95	26.35
OPTIMIZED SYSTEMS SOFTWARE		
Basic A+ (D)	80.00	64.00
OS/A+ (D)	80.00	64.00
PERSONAL SOFTWARE		
Viscalc (D)	199.95	149.00
QUALITY SOFTWARE		
Assembler (T)	24.95	19.95
QS Forth (D)	79.95	63.95
Starbase Hyperion (T)	19.95	15.95
UNITED SOFTWARE OF AMERICA		
Atari World (D)	59.95	47.95
Survival Adventure (D)	24.95	19.95

(C) ROM CARTRIDGE (D) DISKETTE (T) CASSETTE TAPE

REPRESENTING OVER 30 COMPANIES WITH 400+ PROGRAMS AND HARDWARE.

Write for a FREE catalog to:

COMPUTER HOUSE

P.O. Box 369, Dept. 10, Mammoth Lakes, CA 93546

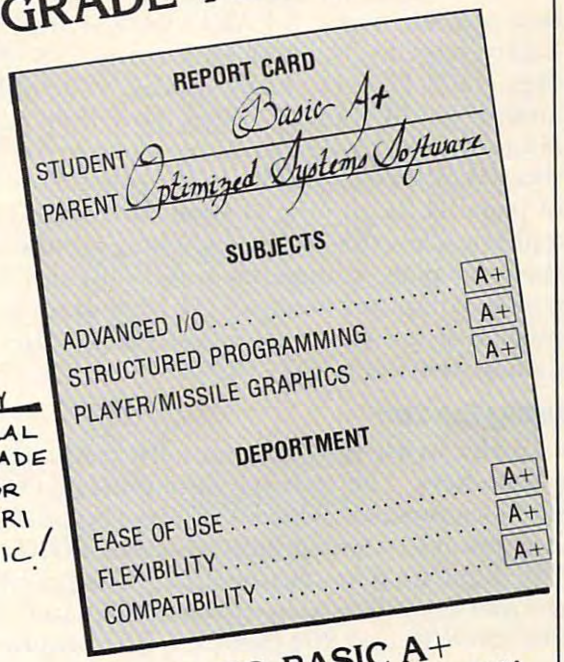
(714) 934-6538

Terms: FOR FAST DELIVERY, send certified checks, money orders or Visa or MasterCard number and expiration date. Personal checks require 3 weeks to clear. ADD $1.50 for postage. Orders over $100.00 we pay shipping. All foreign orders add $10.00. CALIFORNIA residents add 6% tax. Prices subject to change.

* ATARI is the trademark of Atari, Inc.

Atari is a reg. ™ of Atari, Inc.

UPGRADE YOUR ATARI®



THE ONLY LOGICAL UPGRADE FOR ATARI BASIC!

GRADUATE TO BASIC A+
FROM THE AUTHORS OF ATARI BASIC!

BASIC A+

BASIC A+ will rate an A+ from any Atari user! Upward compatible with Atari Basic, it adds statements and features that enhance the Atari 800's real power, flexibility, and ease of use: Superior I/O features for business and other applications. Additional file manipulation commands. Significant help in program development and debug. Structured programming aids. And MORE! A partial list of the enhancements of BASIC A+ includes:

RPUT/RGET (record I/O) **BPUT/BGET** (binary I/O) **ERASE PRINT USING SET TAB INPUT"..." DIR PROTECT RENAME TRACE WHILE...ENDWHILE IF...ELSE...ENDIF**
SUPERB PLAYER/MISSILE GRAPHICS

BASIC A+ requires a disk and 32K bytes of RAM. Since no cartridge is used, BASIC A+ will take advantage of all the RAM (48K bytes) in a maximum Atari 800 system (recommended) $80

OS/A+

Completely compatible with Atari's DOS (version 2), but with an advanced, command-driven console processor. Simple. Flexible. Powerful. With an easy-to-use **BATCH** capability. OS/A+ **INCLUDES** all the following utilities (and more):

EASMD (**E**ditor/**AS**se**M**bler/**D**ebug) is our upgraded all-in-one assembly language development package for the 6502 microprocessor. The editor — with such features as FIND and REPLACE — can even edit BASIC A+ programs. The assembler can include multiple source files in a single assembly.

DUPDSK and FORMAT create master or slave disks. Make sector by sector copies of any OS/A+ disk.

COPY a simple, single file copy utility.

All of this power is included in our OS/A+ package $80

For the utmost in capability and flexibility, our combination system — **BASIC A+** and **OS/A+** — is available $150

SOFTWARE IN A CLASS BY ITSELF
AVAILABLE NOW THROUGH LOCAL DEALERS

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

The nominal limit of 150 characters is imposed by a possible bug in ATARI's DOS II. The second half of page five (memory addresses 0580-05FF Hex, 1408-1535 Decimal) appears to be utilized as an internal I/O buffer. When more than 128 bytes are input, the excess winds up on page six. The sort program also resides in the safe (?) user area of page six (beginning at $0620 or 1568). There is a physical law that states: two things cannot occupy the same place at the same time. This also holds true in computer memory. The program has been pushed as far into page six as it can go (there is data stored behind it).

Using The Sort

In order to use the sort, you must feed it certain parameters. The record length must be POKEd into location 205 ($00CD). The sort type (0-Ascending, 1-Descending) would be POKEd into 206 ($00CE). The starting and ending positions of the sort key will also have to be POKEd into locations 203 ($00CB) and 204 ($00CC). The program is expecting to see the offset of the sort key. The offset is the number of positions in front of that byte. For example; the first position of a record has a 0 offset, the second has an offset of 1, and the 100th has an offset of 99. The USeR function that calls the sort will also pass the address of the string containing the file and the record count. For those who are a little unsure of what this is all about, there are a few examples coming up.

Now that you have a routine that will sort your data faster than you can say Rumplestilskin, how do you use it? Here are several suggestions. The best method is to link through our sort/file loader in Program 3. Your existing program that is processing the data file is probably much, much longer than the short loader. The main advantage of using a small program is that you wind up with more free memory. And, since memory is our sort area, the more that is free, the larger the file. If you don't type the REMark statements, you'll have even a larger sort area. The disk file must be fixed length records terminated by end-of-line characters. Your existing processing program must contain the POKEs mentioned above. It may look something like this:

```
POKE 203,SKEYA-1:POKE 204,SKEYB-1:POKE 205,
RECLEN:POKE 206,0 (for Ascending).
```

The call to the loader would be a RUN "D:SORTLOAD" (give the loader this file name when you save it). The sort/file loader must have your file name in the variable F$ and your program name in P$. If your processing program handles several files, you can also pass the file name by using the following statements. First, your program:

```
FOR I=0 TO 14:POKE 1776+I,32:NEXT I
FOR I=0 TO LEN(F$):POKE 1776+I,ASC(F$(I,I)):
   NEXT I
```

Note: F$ is your file's name.

The sort/file loader will require the following lines to be added:

```
70 FOR I=0 TO 14:F$(I,I)=CHR$(PEEK(1776+I)):
   NEXT I
80 IF F$(1,2)<>"D:" THEN ? "ERROR":END
```

If your processing program or file is small, you may do all of the above from within your program. Besides the same POKEs as above (you wouldn't need the file name, of course), you will need the following line added to your program:

```
IF RC>1 THEN A=USR(1568,ADR(X$),RC)
```

(RC is the number of records stored in the string X$.) Substitute your names where applicable.

Program 4 is a sort/merge utility that uses the same sort routine. This will give you the ability to handle much larger files. With a 40 or 48K machine you will be able to sort files that are 60,000 bytes long (If the record length is 60 characters, that will translate to 1,000 records). This particular version divides the file into two manageable sub-files, sorts each, and then merges them. Be careful with your disk space; the temporary file will need room also. If you have more than one drive, you can modify the program to split it three or more ways and sort even more records. For example, put the temporaries on drive 2 and the new file on drive 3. Who said micros can't handle larger files?

Your Options

The sort/merge program is a stand-alone. By swapping the front end with the sort loader (Program 3), you can do a sort/merge from a call (RUN "D:SORTMERG") in your existing software.

Now that you know how to feed the sort its required parameters and call it, you must still get it into memory. Once again, you have several options. If you have the Assembler/Editor cartridge (or a similar assembler), the source appears in Program 1. Please feel free to modify it if you so desire. If you're limited to BASIC, Program 2 will load the machine language code when it is run. After doing either of these, you should go directly to DOS (DOS II only) and do a binary save (option K) with the following parameters:

```
D1:AUTORUN.SYS,0620,069D
```

Saving the code as AUTORUN.SYS will enable the program to auto-boot when you power up with the disk (You *must* power up with that disk). Do *not* append an INIT or RUN address to the file unless you want the machine to lock up every time you turn it on.

> SEBREE'S COMPUTING <
)--- 456 Granite Ave. , Monrovia , California. zip- 91016 U.S.A. ---(
PRESENTS SOFTWARE FOR THE ATARI 8ØØ
and the 16K EXTENDED BASIC
TRS-8Ø COLOR COMPUTER

Please make sure that you specify which computer you are ordering for.
All of the games below have sound effects and graphics, and require joystks.

ATARI 800	16K EXTENDED TRS-80 COLOR COMPUTER

3-D RED BARON DOGFIGHT- Our best selling program!------------$16.95 +.95p&h

BATTLING TANKS--For 2 players, another top seller!------------$9.95 +.95p&h

WUMPUS ADVENTURE-EXPANDED VERSION!--Adventure!---------------$17.95 +.95p&h

DOWN THE TRENCH-Star Wars simulation! 1 Player. Top Seller---$16.95 +.95p&h

SUBMARINE MINEFIELD Navigator-1 or 2 Players------------------$9.95 +.95p&h

-----------------------NEWEST PROGRAMS ARE BELOW!!-----------------------
FLIGHT SIMULATOR! Graphics! 1 player. Top seller!------------$17.95 +.95p&h

AIR-SEA WATCH COMMANDER! Graphic! 1 Player-------------------$16.95 +.95p&h

----------------/ /----OUR VERY BEST!!----\ \----------------
TRIP TO JUPITER SPACE ADVENTURE!! Graphics! MANY scenes. 1 Player. Requires 32K ATARI. Loades in TWO-halves on TRS-80 COLOR COMPUTER.----$18.95 +.95p&h

3-DIMENSIONAL COMPUTER GRAPHICS PACKAGE!!! Design your own 3-D wire-frame objects and view these images from any angle(s) you wish! Many programs are included within the package! A VERY powerful package! Major differences between ATARI and COLOR COMP. versions. COLOR COMP. Requires Joystk. Large manual included W/full listings!! Disk version is recommended and available for either ATARI 800 or TRS-80 COLOR COMPUTER. Add $5.00 for DISK version.
ATARI 800 version (8K to 48K) supplied on cassette--------$29.95 +$1.50 p&h
16K EXTENDED TRS-80 COLOR COMPUTER cassette version-------$24.95 +$1.50 p&h

ATARI 800- > GRAPHICS EDITOR! < 16K min. 40K Preferred. Design 2-D or 3-D scenes with a Joystick, Save them to DISK, Retrieve them and edit scenes on screen in any Graphics Mode! 'HELP' option included. Complete package of 5 programs. Disk version is recommended (add $5.00). Powerful and well documented! With listings. ATARI 800 only. On cassette--------$29.95 +$1.50 p&h

AT WEST COAST COMPUTER FAIRE- Booth # 1324

ATARI®

PAYROLL SOFTWARE FOR THE ATARI 800*

The MILES PAYROLL SYSTEM# is the first of a series of business software for the Atari 800*. Atari* graphics and sound have been greatly utilized and a detailed and comprehensive manual leads the user step by step enabling a person with little experience to easily operate the entire software package.

- Randon access file organization for fast updating of individual records
- Allows weekly, biweekly, semi-monthly, monthly, quarterly, semi-annual, or misc. payroll periods
- Calculates and prints payroll checks automatically
- Monthly, quarterly, and yearly cumulative totals maintained for each employee
- User-defined workman's compensation classifications
- Complete reporting, including W-2 at end of year
- User-defined earnings and deductions at end of year
- Automatic data error detection
- Packaged in a handsome three ring binder with diskettes and manual

To order, or for more information:

Miles# TM

MILES COMPUTING#
8941 Owensmouth Ave. #202
Canoga Park, CA 91304
(213) 700-1166

Special introductory price $179.95. Requires 32K and two Atari* 810 disc drives. Payment in U.S. funds required with order. Add $3.00 shipping/handling. California residents add 6% sales tax. Dealer inquiries welcome.

*Atari and Atari 800 are trademarks of Atari, Inc.
#Miles Computing and Miles Payroll System are trademarks of Miles Computing, Canoga Park, California. Not affiliated with Atari, Inc.

RAM
For ATARI

48K RAM BOARD FOR THE 400

- Increases memory capacity
- Reduces power consumption
- Reduces heat

48K Board (400) **$299**
32K Board (800) **$150**

INTEC
PERIPHERALS
CORP
3389 Del. Rosa Avenue
San Bernardino, CA 92404
(714) 864-5269

ATARI, 400, 800 are Trademarks of ATARI, Inc.

COMPUTER
Calligraphy?

Well, not really! But with **FONTEDIT** you can design your own character sets for the ATARI. You can create special graphic symbols for use in games, or even make a Russian alphabet. After you design a new set of characters, save them on disk or tape for later use. We provide a subroutine that you can use in your own programs to load a custom character set.

FONTEDIT is very easy to use: you use a joystick to design a large copy of the new character as you individually control whether each of the 64 dots that make-up a character are *on* or *off*. A menu of **FONTEDIT** commands on the screen shows you how to control this powerful yet simple-to-use character editor. The 56-page *User's Guide* provides clear, easy-to-understand instructions for **FONTEDIT**. There is also a detailed, line-by-line description that explains exactly how **FONTEDIT** works.

In addition to the **FONTEDIT** program, our IRIDIS #2 package includes three custom character sets, a routine to load the cstom characters for use in your own programs, and a program that shows the use of custom-designed graphic characters. IRIDIS #2 is an exceptional value: only $17.95 for the cassette, (requires 16K) or $20.95 for disk (requires 24K). California residents add 6% tax. VISA and MASTERCARD welcome.

THE CODE WORKS
Box 550
Goleta, CA 93116
805-683-1585

ATARI is a registered trademark of Atari, Inc.

Program 1.

```
;  RON MARCUSE, FREEHOLD NJ   11/29/81
;
;     CALLED FROM BASIC WITH:
;
;     A=USR(1568,ADR(X$),RC)
;
;  NOTE: X$ IS THE STRING THAT CONTAINS
THE FILE
;        RC IS THE NUMBER OF RECORDS
;
;  THE FOLLOWING ARE POKED BY BASIC PROG
RAM:
;
;      SS - BEGINNING OF SORT KEY (DECIM
AL- 203)
;      SE - END OF SORT KEY (DECIMAL - 2
04)
;      RL - RECORD LENGTH (DECIMAL - 205
)
;      TYPE - ASCENDING (0)  OR DESCENDI
NG (1)
;             (DECIMAL - 206)
;
; THE ROUTINE WILL LOOP THROUGH "FILE" S
WAPPING UNSORTED
; ADJOINING MEMBERS UNTIL THE "SWAP FLAG
" HAS NOT BEEN SET
; IN A GIVEN PASS. THE ZERO PAGE ADDRESS
ES "FST" AND "SEC"
; POINT AT THE INDIVIDUAL MEMBERS BEING
COMPARED. THE Y
; REGISTER IS USED AS AN INDEX POINTER F
OR TESTING OR
; MOVING BYTES WITHIN THE TWO RECORDS.
;
;
      *=   $0620   START AT PAGE 6
;     MEMBER n ADDRESS (LSB,MSB)
FST    =    $D4
;     MEMBER (n+1) ADDRESS (LSB,MSB)
SEC    =    $D6
;     BASE ADDRESS OF LIST (LSB,MSB)
BASE   =    $D8
;     FIRST POSITION OF SORT KEY
SS     =    $CB
;     LAST POSITION OF SORT KEY
SE     =    $CC
RL     =    $CD     ELEMENT LENGTH
SWAP   =    $DA     SWAP SWITCH
;     NUMBER OF ELEMENTS (LSB,MSB)
RC     =    $D0
;     RECORD COUNTER (MSB, X REG IS LSB)
CNTH   =    $CF
;     SORT TYPE, 0-ASC  1-DES
TYPE   =    $CE
;
;
;    POP # OF ARGUMENTS FROM STACK
      PLA
      PLA
      STA   BASE+1   SET BASE ADDRESS
      PLA
      STA   BASE
      PLA
      STA   RC+1     SET ELEMENT COUNT
      PLA
      STA   RC
;
;
;    START EACH PASS THROUGH FILE
BEGIN LDA   #$00
      STA   SWAP     SET SWAP TO 0
      STA   CNTH     SET HIGH COUNT TO 0
;    SET X REGISTER TO 1 (LOW COUNT)
      LDX   #$01
;    SET POINTER (n) TO BASE
      LDA   BASE
      STA   SEC
      LDA   BASE+1
      STA   SEC+1
;
CONT  CLC
      LDA   SEC      RESET POINTERS-
      STA   FST       (n) to (n+1)
      ADC   RL
      STA   SEC       (n+1) to (n+2)
      LDA   SEC+1
      STA   FST+1
      ADC   #$00
      STA   SEC+1
;    ASCII STRING COMPARISON
      LDY   SS
;
;    ASCENDING OR DESCENDING?
COMP  LDA   TYPE
      BEQ   ASC      SORT IS ASCENDING
      LDA   (SEC),Y TYPE = DESCENDING
;    COMPARE ADJOINING MEMBERS
      CMP   (FST),Y
      BCC   BACK     (n)>(n+1)
      BEQ   INCR     (n)=(n+1) TRY AGAIN
      BCS   FLIP     (n)<(n+1)
;
ASC   LDA   (SEC),Y TYPE = ASCENDING
;    COMPARE ADJOINING MEMBERS
      CMP   (FST),Y
      BCC   FLIP     (n)>(n+1)
      BEQ   INCR     (n)=(n+1) TRY AGAIN
      BCS   BACK     (n)<(n+1)
;
INCR  INY            ADD 1 TO POINTER
      CPY   SE       END OF SORT KEY?
      BEQ   COMP     NO
      BCS   BACK     YES, NEXT ELEMENT
```

```
       BCC   COMP     NO
;
;      SWAP ELEMENTS (n),(n+1)
FLIP   LDA   #$01
       STA   SWAP     SET SWAP SWITCH ON
       LDY   RL       LOAD LENGTH
;
MOVE   DEY            SET  DISPLACEMENT
       LDA   (SEC),Y EXCHANGE BYTES
       PHA
       LDA   (FST),Y
       STA   (SEC),Y
       PLA
       STA   (FST),Y
       CPY   #$00     MORE BYTES TO SWAP?
       BNE   MOVE     YES
;
;      INCREMENT RECORD COUNTER
BACK   INX
       CPX   #$00     CHECK FOR >255
       BNE   TEST
       INC   CNTH     ADD 1 TO HIGH COUNT
;
TEST   CPX   RC       END OF FILE?
       BNE   CONT     NO
       LDA   RC+1     CHECK HIGH EOF
       CMP   CNTH
       BNE   CONT     NOT END OF FILE
       LDA   SWAP     TEST FOR END OF SORT
       CMP   #$00     ANY SWAPS?
       BNE   BEGIN    YES, START OVER
      NO, RETURN TO CALLING PROGRAM
       RTS
       .END
```

Program 2.

```
100 FOR I=1568 TO 1693:READ A:POKE I,A:N
EXT I
1568 DATA 104,104,133,217,104,133
1574 DATA 216,104,133,209,104,133
1580 DATA 208,169,0,133,218,133
1586 DATA 207,162,1,165,216,133
1592 DATA 214,165,217,133,215,24
1598 DATA 165,214,133,212,101,205
1604 DATA 133,214,165,215,133,213
1610 DATA 105,0,133,215,164,203
1616 DATA 165,206,240,10,177,214
1622 DATA 209,212,144,44,240,12
1628 DATA 176,19,177,214,209,212
1634 DATA 144,13,240,2,176,30
1640 DATA 200,196,204,240,227,176
1646 DATA 23,144,223,169,1,133
1652 DATA 218,164,205,136,177,214
1658 DATA 72,177,212,145,214,104
1664 DATA 145,212,192,0,208,241
1670 DATA 232,224,0,208,2,230
1676 DATA 207,228,208,208,172,165
1682 DATA 209,197,207,208,166,165
1688 DATA 218,201,0,208,144,96
```

Program 3.

```
10 REM SORT LOAD PROGRAM  LYNN MARCUSE 1
1/27/81
11 REM
12 REM CALLING PROGRAM MUST:
13 REM
14 REM *  POKE RECORD LENGTH INTO LOCATI
ON 205
15 REM *  POKE BEGINNING OF SORT KEY INT
O LOC 203
16 REM *  POKE END OF SORT KEY INTO LOCA
TION 204
17 REM *  POKE TYPE (ASCENDING - 0 OR DE
SCENDING - 1) INTO LOC 206
18 REM
19 REM THIS PROGRAM WILL LOAD FILE INTO
MEMORY AND CALL MACHINE
20 REM LANGUAGE ROUTINE. WHEN COMPLETED,
 YOUR PROGRAM MAY BE
21 REM RE-CALLED BY EQUATING P$ TO YOUR
PROGRAM NAME.
22 REM
50 DIM X$(FRE(0)-600),R$(130),F$(15),P$(
15),I$(1)
58 REM
59 REM REPLACE X'S WITH YOUR FILE & PROG
RAM NAMES
60 P$="XXXXXX":F$="XXXXXX"
99 REM GET RECORD LENGTH
100 RET=100:R=PEEK(205)
109 REM OPEN FILE AND INPUT RECORDS
110 ? " LOADING ";F$:TRAP 600:OPEN #2,4,
0,F$:L=1
120 TRAP 140:INPUT #2,R$:TRAP 40000
130 X$(L,L+R-1)=R$:L=L+R:GOTO 120
140 CLOSE #2:L=L-1:N=L/R:? " RECORDS LOA
DED= ";N
149 REM CALL MACHINE LANGUAGE SORT ROUTI
NE
150 IF N>1 THEN ? " BEGIN SORT":A=USR(15
68,ADR(X$),N)
160 RET=170:? " COMPLETED  SAVING ";F$
169 REM ERASE OLD FILE AND SAVE NEW ONE
170 TRAP 600:XIO 36,#2,0,0,F$:OPEN #2,8,
0,F$
180 FOR I=1 TO L STEP R:R$=X$(I,I+R-1):?
 #2;R$:NEXT I
190 CLOSE #2:XIO 35,#2,0,0,F$
199 REM RETURN TO YOUR PROGRAM ?
200 RET=200:TRAP 600:IF P$(3,4)<>"XX" TH
EN ? " LOADING ";P$:RUN P$
210 END
600 ? " ERROR - ";PEEK(195):CLOSE #2
610 ? " PRESS RETURN TO CONTINUE";:INPUT
```

```
 I$:GOTO RET
```

Program 4.

```
10 REM SORT MERGE PROGRAM  RON MARCUSE 1
2/01/81
11 REM
12 REM THIS PROGRAM WILL LOAD FILE INTO
MEMORY AND CALL MACHINE
13 REM LANGUAGE ROUTINE. IF FILE IS TOO
LARGE, THE SORTED DATA
14 REM WILL BE SAVED AS "D:TEMP" AND BAL
ANCE OF FILE WILL BE
15 REM READ AND SORTED. WHEN THIS STEP I
S FINISHED, THE TEMPORARY
16 REM FILE  WILL BE MERGED WITH THE SOR
TED DATA IN MEMORY.
17 REM
20 GRAPHICS 0:DIM F$(15):? :? ,"SORT/MER
GE UTILITY":POKE 82,1
30 ? :? "ENTER:":? :? "FILENAME (D:name.
ext) ";:INPUT F$
40 ? "RECORD LENGTH ";:TRAP 40:INPUT R:T
RAP Q3:IF R<2 OR R>150 THEN 40
50 ? "SORT KEY (1st,2nd) ";:TRAP 50:INPU
T SS,SE:TRAP Q3
55 IF SS>=SE OR SS<0 OR SE>R THEN 50
60 ? "ASCENDING - 0  OR DESCENDING - 1 "
;:TRAP 60:INPUT T:TRAP Q3
65 IF T<0 OR T>1 THEN 60
70 POKE 205,R:POKE 203,SS-1:POKE 204,SE-
1:POKE 206,T
80 XL=FRE(0)-600:DIM X$(XL),R$(R),T$(R),
D$(6)
90 Q1=210:Q2=600:Q3=40000:D$="D:TEMP"
100 ? "LOADING ";F$:TRAP Q2:OPEN #2,4,0,
F$:M=0
120 L=1:? "PASS 1 - ";:GOSUB 500:IF M=0
THEN 160
140 ? "WRITING ";D$:OPEN #3,8,0,D$:GOSUB
 560
150 ? "PASS 2 - ";:L=1:GOSUB 500
160 CLOSE #2:? "DELETING ";F$
170 TRAP Q2:XIO 36,#3,0,0,F$:OPEN #3,8,0
,F$
180 ? "WRITING ";F$:IF M=0 THEN GOSUB 56
0:GOTO 400
200 TRAP Q2:OPEN #2,4,0,D$:J=1:A=1:B=1:A
E=1:BE=1
210 IF A=1 THEN TRAP 330:INPUT #2,R$:TRA
P Q3
220 IF B=1 THEN TRAP 340:T$=X$(J,J+R-1):
J=J+R:TRAP Q3
230 IF AE=0 AND BE=0 THEN 390
240 IF AE=1 AND BE=0 THEN 300
```

ATTENTION ATARI OWNERS

32K MEMORY BOARD FOR ATARI	$129.00
Atari 800 with 16K Memory	$679.00
Atari 800 with 48K Memory	$789.00
Atari 810 Disk Drive	$449.00

NEW PRO FOOTBALL FOR ATARI WITH 16K Cassette $9.95
Disk $13.95

20% Discount on All Atari Software

COMPUTER COUNTRY
909 North Main Street
Randolph, MA 02368
(617) 961-3285

Cash...Money Order...COD...Certified Check

```
245 IF AE=0 AND BE=1 THEN 310
250 IF T=1 THEN 280
260 IF R$(SS,SE)>T$(SS,SE) THEN 310
270 GOTO 300
280 IF R$(SS,SE)<T$(SS,SE) THEN 310
300 ? #3;R$:A=1:B=0:IF AE=0 THEN A=0:B=B
E
302 GOTO Q1
310 ? #3;T$:A=0:B=1:IF BE=0 THEN B=0:A=A
E
312 GOTO Q1
330 AE=0:GOTO 220
340 BE=0:GOTO 230
390 CLOSE #2:? "DELETING ";D$:XIO 33,#2,
0,0,D$
400 CLOSE #3:XIO 36,#3,0,0,F$
410 END
500 TRAP 530:INPUT #2,R$:TRAP Q3
510 X$(L)=R$:L=L+R:IF (L+R)<XL THEN 500
520 M=1
530 L=L-1:N=L/R:? "RECORDS LOADED = ";N
540 IF N>1 THEN ? "BEGIN SORT  ";:A=USR(
1568,ADR(X$),N)
550 ? "END SORT":RETURN
560 FOR I=1 TO L STEP R:R$=X$(I,I+R-1):?
 #3;R$:NEXT I:CLOSE #3:RETURN
600 ? "ERROR - ";PEEK(195):END
```

©

---

1 STOP SOFTWARE for the ATARI 400/800*

ACR: A POINT OF SALE PROGRAM

CASH RECEIPTS, INVOICING, INVENTORY, COST AVERAGING, PURCHASE ORDERS, RECEIVING RECORDS, 40 OR 80 COL. PRINTER.

48K DISK INTRO OFFER: $169.95

CASSETTE CHECKBOOK PROGRAM

MULTIPLE ACCOUNTS, 200 TRANSACTIONS PER PERIOD, USER DEFINED CATAGORIES, CREATIVE USE OF GRAPHICS AND SOUND.

16K CASS.: $14.95

SHADOW HAWK 1

THIS IS A MACH. LANG. SPACE ADVENTURE, WHICH WE KNOW WILL OFFER COMPETITION EVEN TO STAR RAIDERS.* OFFERS 3-AXIS ROTATION; HIRES GRAPHICS & ALL OF THE SPACE SHOOTEM UP YOU'LL EVER NEED (by HORIZON SIMULATIONS).

REQUIRES AT LEAST 48K, 1 DISK DRIVE: $48.95

DEALER INQUIRIES INVITED

High Country Microsystems
3489 W. 72nd Ave. Suite 108
Westminster, CO 80030
(303) 427-9036   426-5329

$2.00 Shipping & Handling. ADD 2.50 for COD.
Colo. residents add 6.5% sales tax.
Free catalogue available. MC and VISA accepted.

*ATARI is a registered trademark of ATARI, Inc.
SHADOW HAWK 1 is a trademark of HORIZON SIMULATIONS

Si **SYNCRO, INC.**

SOFTWARE DIVISION
31332 VIA COLINAS SUITE 107
WESTLAKE VILLAGE, CALIF. 91361
213-889-9508

SOFTWARE FOR ATARI T.M. 400/800

[illegible] © 1982

ALL GRAPHIC ADVENTURE. CAN YOU FREE YOURSELF FROM ALIEN HELL? IN THE SPIRIT OF THE "MAZE OF DEATH".

JOYSTICK REQUIRED

16K Tape $14.95 | 24K Disk $19.95

MAR TESORO © 1981

A NEW GRAPHIC ADVENTURE, SIT BACK WITH YOUR JOYSTICK AND DIVE THE DEEP BLUE SEA. RENT RIG AND CREW TO PLUNDER THE STRANGE TREASURES THAT AWAIT YOU.

JOYSTICK REQUIRED

24K Tape $16.95 | 32K Disk $21.95

MAZE OF DEATH © 1981

ARE YOU A SUPERAGENT? A FAST ACTION, ALL GRAPHIC GAME WITH PLENTY OF SPECIAL EFFECTS IN A MAZE LIKE UNDERGROUND COMPLEX. FIVE LEVELS OF PLAY IN REAL TIME.

JOYSTICK REQUIRED

16K Tape $12.95 | 24K Disk $17.95

VISA AND MASTERCARD ACCEPTED.
CALIF. RES. PLEASE ADD 6% SALES TAX
PLEASE INCLUDE $1.50 SHIPPING
DEALER INQUIRIES WELCOME

Sunrise Software®

BUDGETMASTER ($34.95)

- is a professional budgeting tool geared to develop and maintain even the most complex budget. At the same time, the easy to read manual and the friendly prompts allow even the beginning budgeter without computer experience to develop a budget to meet their needs.
- includes the option for hardcopy printout of any of the 39 different displays as well as an option to save your data to cassette or diskette.
- requires 32K RAM

NOW ON CASSETTE

CHECKBALANCER ($16.95)

- although the perfect companion program to the Budgetmaster, stands alone as a great tool to balance your checking account(s), with a monthly breakdown of your spending pattern.
- allows you to store your spending analysis to tape or diskette, or use **Checkbalancer** in conjunction with the **Budgemaster**, and automatically rewrite the expense data of the latter program process.
- requires 16K RAM

BUY BOTH PROGRAMS TOGETHER FOR $39.95 AND SAVE $11.95!!!

Send check or money order plus $1.50 for shipping and handling and save C.O.D. fee (Sorry, no credit cards)

Ohio Residents Add Sales Tax

(DEALER INQUIRIES WELCOME)

Sunrise Software® • **P.O. Box 25621 • Garfield Hts., Ohio 44125**
(216) 581-9359

*Atari is a Trademark of Atari, Inc.

Dynamic Renumber

R. D. Young
Ottawa, Ontario

Program line renumbering is often more than just cosmetic. Afterthoughts, frequently called *bugs*, invariably use up all those spaces left between original program lines. There are a number of line renumbering programs/utilities available for PET (and other computer) owners. Unfortunately, those that I have seen, including Toolkit, renumber the entire program, once invoked. It is therefore impossible to retain blocks of subroutines, as might be initially intended.

Blocks of subroutines, 1000-1999 or 2000-2999 for example, are particularly helpful during program development. It is easier to remember a thousand-line block while debugging (and leaving lots of space between blocks) than, for example, something like 760-790. At the same time, the mainline program or a subroutine block of lines may require renumbering during the debugging stage. A segment of the program can now be renumbered with Dynamic Renumber.

This program is a modified version of Resequencer by Joe Trimble from PET User Notes, Issue 5, July-August 1978, which was modified by Jim Russo and Henry Chow in PET User Notes, Issue 7, November-December 1978.

Dynamic Renumber will renumber the selected range of lines beginning with the desired new line number and using the desired increments. It will abort if the highest renumbered line overlaps a line not selected for renumbering, but it will give erroneous line numbers if the overlap occurs at the beginning of the renumbered segment. The program will then locate all GOTO's, GOSUB's, THEN's, ON...'s, and RUN's, and insert the new target line number if required. If, however, the new target line number is longer than the old line number, only part of the new line number will be inserted. When such an event occurs, the line number of the line in which the shortened insertion is being made and the proper target line will be printed side-by-side on the screen. An asterisk is printed as each program line is being analyzed for required changes.

This program will function quite nicely as a utility stored in and run from a 4K memory partition. The program to be renumbered must, of course, reside in the normal low end of memory. Alternatively, this program can be readily appended to a program already in memory.

Dynamic Renumber can be easily converted to other than PET BASIC, provided that line numbers are stored in the same manner (see also "Program Compactor," **COMPUTE!** #11). The first four bytes of each line are defined as follows:

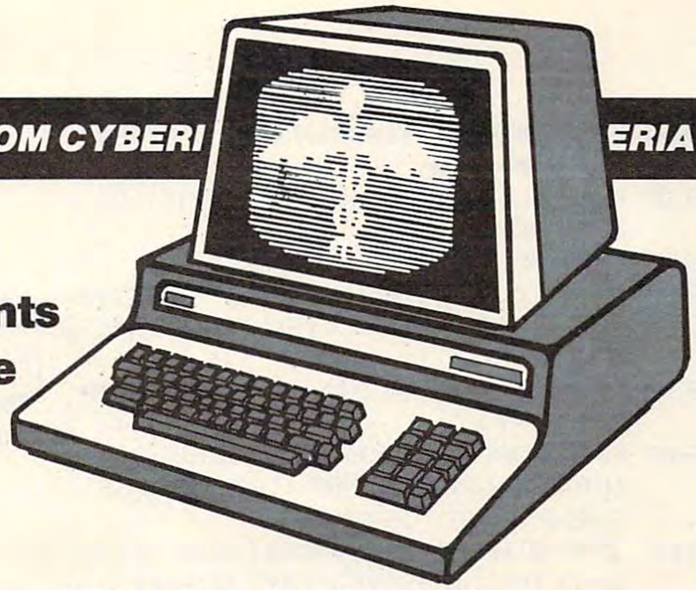
Pointer to next line – low byte
Pointer to next line – high byte
Line number – low byte
Line number – high byte

Changes to Dynamic Renumber, required before implementation with other BASIC's, are the start-of-BASIC pointer and the GOTO, GOSUB, etc. token values. The start-of-BASIC in the PET is 1025 decimal; this is the number that must be changed in lines 63895, 63933, and 63937. The applicable statement tokens are in line 63940 (assigned to variable P).

As one last precaution, you may wish to retain the space between the variable LE and the statement THEN in the associated IF...THEN statements, thus avoiding BASIC confusion with the LET statement.

```
63776 REM END RENUMBER
63887 REM LINE RENUMBER - RUN63888
63888 PRINT"RENUMBER":INPUT"START AT LINE #";LS
63889 INPUT"END AT LINE #";LE:IFLE>=63776THENLE=63775
63890 IF LS>= LE THEN63888
63891 INPUT"FIRST NEW LINE #";Z
63892 INPUT"INCREMENT NEW LINES BY";K
63895 DIML(500):L=1025:DEFFNR(X)=PEEK(X)+256*PEEK(X+1):REM*OLD ROM DIM L
      (255)*
63900 DEFFNM(X)=INT((K*X-K+Z)/256)
63902 N=FNR(L):X=FNR(L+2):IFX<LSTHENL=N:GOTO63902
63904 L1=L
63910 N=FNR(L):X=FNR(L+2):IFX<= LE THENA=A+1:L(A)=X:L=N:IFN=0THEN63920
63912 IFX<=LE THEN63910
63915 Y=INT(K*A-K+Z):IFX<=YTHENPRINT"MAX. LINE OVERLAP - CK. PGM":END
```

NEW FROM CYBERIA ★ NEW FROM CYBERI ERIA



Mari

A comprehensive accounts receivable and insurance billing system for modern health care offices and clinics

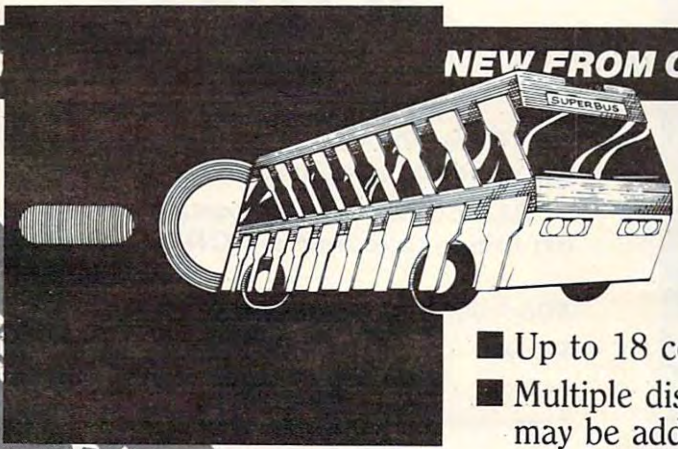
DESIGNED FOR COMMODORE 8000 SERIES COMPUTERS AND DISK DRIVES

- Help functions are always on-line
- Supports CPT, ICD and RVS medical coding
- Supports open item or balance forward accounting systems
- Interfaces to popular word processing programs
- Multiple terminals may be added with Superbus 4.0
- Multiple disk drives may be used—no limit on number of patients or accounts
- Includes a data base and forms generator to fill out any insurance form
- Includes a computer aided instruction program to train new users

Includes these standard reports:

- A/R aging
- A/R transactions
- General ledger
- Instant cash receipt
- Income analysis by physician
- Doctor referral report
- Patient/account cross reference
- Standard SuperBill insurance form

N NEW FROM CYBERIA ★ NEW FROM CYBERIA



SUPERBUS 4.0

Commodore Computer Networking System

- Up to 18 computers can be interconnected
- Multiple disk drives, printers and other devices may be added to the network
- Sophisticated security system prevents unauthorized use
- WordPro and Wordcraft programs are supported to allow multiple terminal word processing

NEW ★ NEW ★ NEW

*WordPro is a trademark of Professional Software, Inc.

CYBERIA inc. 2330 LINCOLN WAY, AMES, IA 50010 • 515-292-7634

```
63920 L=L1:FORB=1TOA:N=FNR(L):POKE(
      L+3),FNM(B)
63930 POKE(L+2),K*B-K+Z-256*FNM(B):
      L=N:NEXT
63933 L=1025
63935 N=FNR(L):X=FNR(L+2):IFX<63776
      THENAA=AA+1:L=N:IFN<>0THEN
      63935
63937 L=1025:FORB=1TOAA:N=FNR(L):X=
      FNR(L+2)
63940 F=0:FORC=L+4TON-1:P=PEEK(C):
      IFP=137ORP=141ORP=167ORP=138
      THENF=1:GOTO63999
63950 IFF>0THENF=0:IFP<58THENF=1:G=
      G+1:IFP>47THEND=10*D+P-48:
      GOTO63999
63960 IFD=0GOTO63999
63970 FORE=1TOA:IFD=L(E)GOTO63990
63980 NEXTE:D=0:G=0:GOTO63999
63990 D=0:E$="         "+STR$(E*K-K+Z):
      H=LEN(E$):C=C-G:IFP<48THENG=
      G-1:C=C+1
63995 IFH-6>GTHENPRINTX;E*K-K+Z;
63997 FORI=1TOG:POKEC,ASC(MID$(E$,I
      +H-G,1)):C=C+1:NEXTI:G=0
63999 NEXTC:L=N:PRINT"*";:NEXTB:END
READY.
```
©

NEW COMMODORE ADD-ONS
RAM-ROM:
GETS RID OF SAFING ROM

***MX-910 CBM/PET RAM/ROM;**
Allows multi ROM protected programs using the same socket to be put onto diskette/cassette, no need to insert protect ROM in socket after initial load, eliminates need for ROM switch box, write protect in software, decoded for dual ROM socket usage, 4K expandable to 8K, easy internal CBM installation: $119.95

***MX-232 CBM/PET TO RS-232C INTERFACE:**
Low cost, bidirectional, 50 to 19,200 baud rate, full modem controls, parity allows for two RS-232C CBM ports, installs easily inside CBM: $199.95

SX-100 IEEE-488/PET MODEM SOFTWARE:
Best 810 modem software, by 8010 developer, works with Source/Micronet/CBM to disk/CBM to CBM: $79.95

***MX-200 IEEE-488/PET PARITY MODEM/SOFTWARE**
Talk to a host computer requiring parity, all features of SX232: $399.95

***MX-903 CBM/PET AUTORESET:**
Easily installed reset circuit: $19.95

ECX COMPUTER COMPANY
2678 NORTH MAIN ST.
WALNUT CREEK, CA 94596
(415)944-9277

PETTERM

INTELLIGENT TERMINAL

Turn your PET into an intelligent terminal with one of our terminal packages. These are complete assembled hardware and software packages. All include line editing/resend, repeat key, shift lock, output to CBM printer, and more . . . Delivered on PET cassette with manuals.

PETTERM I	All features above $75.00
PETTERM II	All features of I, plus local text editor with down-loading capability . . $90.00
PETTERM III	All features of II, plus 80/132 column scrolling window for viewing formatted outputs wider than 40 columns. $100.00

forth

for PET/CBM

FORTH is a new concept in programming, with the speed of compilers and interactive ease of BASIC. Programs become a part of FORTH extending the power of FORTH and your PET.

8050,4040 disk, cassette all PET-CBMs 16k+

Starter	fig-FORTH w editor assembler	$35
Personal	floating point; strings; source	$75
Professional	turnkey development/data base	$259

F S S
software for small computers

1903 Rio Grande
Austin, Texas
78705

1-512-477-2207

P.O. Box 8403
Austin, Texas
78712

master charge

DEALERS INQUIRE

VISA

ASERT yourself . . . with CFI's new Database Retrieval System

WHO CAN USE ASERT?

libraries
personnel departments
dating services

OFFICIALLY APPROVED BY COMMODORE

schools
employment agencies
accountants

ANY BUSINESS THAT KEEPS RECORDS CAN USE ASERT TO:

- Create up to 21 fields per record
- Restructure fields at any time
- Sort on any field at any time
- Use FREE-TEXT area for comments
- Create up to 90 searchwords
- Search & retrieve on any combination of 90 searchwords
- Search with MUST HAVE, MAY NOT HAVE and OPTIONAL operators
- Print out hardcopy including labels
- Output to any word processor
- Compile summary statistics
- Maintain 1900 records per disk with "virtual" 5K record length

ASERT — Aid for Search & Retrieval of Text — $495 complete
For the 8032 CBM and 8050 disk drive — Commodore Approved Software

OTHER CFI SOFTWARE

Federal Income Tax Preparation System*
Personal Tax Calculator*
Emergency Control Program*
VIC Animation Tutorial
*Distributed for CFI under the Commodore label

ALL CFI SOFTWARE AVAILABLE
from your local Commodore dealer
or direct from CFI

CFI . . . *Computer Solutions,* 201 West 92 St., New York, NY 10025

NOW COMMODORE TALKS. AND LISTENS.

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

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

Some specifications.
COGNIVOX can be trained to recognize words or short phrases drawn from a vocabulary of up to 32 entries chosen by the user. To train COGNIVOX to your vocabulary, all you have to do is repeat the entries three times at the prompting of the computer. The voice output vocabulary can also have up to 32 words or phrases of your choice. Data rate is approximately 700 bytes per word. Vocabularies can be stored and recalled from disk, giving in effect unlimited selection of entries.
COGNIVOX VIO-1002 will work with all Commodore computers (old, new and newer ROMs) with at least 16K of RAM. It comes complete with assembled and tested hardware in a quality instrument case, speaker/amplifier, power supply, microphone, cassette with software and detailed user manual.

Easy to use
All you need to get COGNIVOX up and running is to plug it in the user port and load one of the programs supplied. Load the demon program and start talking to your computer right away. Or load one of the games and discover the magic of voice control.
It is easy to write your own talking and listening programs too. A single statement in BASIC is all that you need to say or to recognize a word. Full instructions on how to do it are given in the manual.

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

computer
COGNIVOX
L.R. Rabiner R.W. Schafer Digital Processing of Speech Signals
ANTONIOU digital filters: analysis and design
THEORY AND APPLICATION OF DIGITAL SIGNAL PROCESSING

VOICETEK

P.O. Box 388, Goleta, CA 93116

Disk Data Structures: An Interactive Tutorial

David Young
Richardson, TX

The floppy disk is a marvelous and yet mysterious medium for mass storage of data. Indeed, understanding exactly how a bit of data is stored and retrieved from the surface of the disk requires a good knowledge of physics. However, to learn about the data structures found on a disk requires mathematics no more complex than hexadecimal arithmetic. The manual supplied with the computer usually does an adequate job of supplying all the technical details, but wouldn't it sink in better if the actual data on the media could be viewed while it is being described?

The program that is presented here, Diskpeek, was created just for that purpose. Though this program was written for the Atari Personal Computer (DOS 2.0S), the interactive tutorial which follows contains information which should apply, in one form or another, to most other disk based computer systems. Those with a disk based Atari computer should type in Diskpeek before proceeding. This program is used to demonstrate the disk data structures as they are being described. The instructions integrated into the program should make its use self-explanatory.

The Disk Medium

The first disk structure to be aware of is the sector which, on any computer system, consists of a group of contiguous bits recorded at a specific location on the disk. The disk drive hardware always operates on whole sectors, that is to say, it is not possible to read or write partial sectors. Groups of sectors are organized into tracks forming concentric rings about the center of the disk.

The Atari system divides the disk into 40 tracks with 18 sectors per track for a total of 720 sectors. This is best visualized by taking the lid off of the disk drive and watching the read/write head move as certain sectors are addressed. On the Atari 810 disk drive this is accomplished by removing the four phillips head screws hidden under gummed tabs at each corner of the lid. While inside the case, a bit of lubrication on the 2 cylindrical guide rails supporting the head will make the drive less noisy.

If sectors 1 through 18 are read with Diskpeek, the head remains fixed on the outermost track. When sector 720 is read, the head moves in to the innermost track. When a disk is formatted, the head can be seen to bump sequentially through all 40 tracks. It is laying down the patterns on the oxide surface which will be recognized by the drive hardware as the sectors. The sectors are all initially empty (128 bytes of 0), but at the end of the formatting routine, as described in the next section, the Atari DOS records special data into certain sectors. The top of the drive can now be resecured. No more information about the hardware is needed to underatand the higher level disk data structures of the software.

Boot Sector

At the end of the formatting process, DOS reserves and initializes certain sectors for special tasks. Into sectors 1 through 3 is stored the bootstrap for DOS. On power-up the Atari operating system reads sector 1 to determine how many sectors to read and where into memory to load them. After it has loaded in the specified number of sectors, DOS starts executing the new code at the load address + 6. Put Diskpeek into the hex mode and read sector 1 of any DOS disk. Byte 0 says that 3 sectors are read (sequentially) and bytes 1 and 2 specify a load address of $700. (A 2 byte number is always specified with the least significant byte first.) Byte 6 is the first instruction to be executed (a $4C1407 is a JMP $714). In this case the code which follows sets up to load the File Management System of DOS into memory. This is called the second stage of the boot. Look at the first sector of any other boot disk available (any game or program which loads in from disk on power-up). It might be seen that the program loads in entirely during the first stage of the boot, i.e. byte 1 of sector 1 has a sector count which represents the entire program. For more details on the disk boot process, see the Atari *Operating System User's Manual.*

Volume Table Of Contents

Besides the first three boot sectors, DOS sets up sectors 360 to 368 as the directory of the disk. DOS uses the directory to keep track of where files are stored on disk and how much disk space remains. Read sector 360 of a DOS disk with Diskpeek in the hex mode and view a part of the directory called the Volume Table of Contents (VTOC). Information pertaining to the availability of every sector on the disk is stored in this sector. Bytes 1 and 2 specify the maximum number of user data sectors on the disk ($2C3 = 707) and bytes 3 and 4 specify the number of free sectors remaining on

Announcing

ATARI™

software

from the
authors of
An Invitation to Programming

exciting games
and educational programs
for kids,
teenagers
and
adults
featuring sound
and color graphics.

available on
guaranteed-to-load
cassettes
at fine
computer dealers in your
area or,
write us directly for
descriptive materials

PDI

Program Design, Inc.
Department CA
11 Idar Court
Greenwich, CT 06830

203-661-8799

See us at the West Coast Computer Faire — Booth 1348.

MICROWORLD, an adventure within your computer, is available on the ATARI and the TRS-80. You are transformed into an electroid, and must explore the circuits of your computer. Over 80 locations and many original problems exist within the maze of transformers and transistors. We dare you to explore the maze of bit cells! Each version of Microworld explores the workings of its respective computer, Atari or TRS-80. Microworld comes with a booklet defining terms and describing the function of the mystifying inner workings of home computers. Come face to face with a staticon! Explore the Microworld!

SATISFACTION GUARANTEED!
If for any reason you are not satisfied with our products, return your order within 14 days for a prompt and cheerful refund.

ORDERING INFORMATION
Orders are processed within five working days. Shipping and handling charge of $1.00 will be added to all orders within the U.S. and Canada. Overseas orders please add $3.00 for air post.

Atari Microworld Atari 400 and 800 32K Cassette	$19.95
Atari 400 and 800 32K Diskette	$22.95
TRS-80 Microwrold TRS-80 Model I and Model III Level II 16K Cassette	$19.95
TRS-80 Model I and Model III Level II 32K Diskette	$22.95

■ MICROWORLD $ ______
SHIPPING 1.00
TOTAL $ ______

■ TRS-80 ■ ATARI
■ CASSETTE ■ DISKETTE

NAME ______
STREET ______
CITY ______ STATE ______ ZIP ______
■ CHECK ■ VISA ■ MASTERCARD
______ EXP. DATE ______

MED SYSTEMS SOFTWARE
P.O. Box 2674, Chapel Hill, NC 27514
1-800-334-5470

the disk (707 for an empty disk, 0 for a full one). Starting in bit 6 (the second to highest order bit) of byte $0A, each bit up through byte $63 corresponds to a sector. A 1 corresponds to a free sector while a 0 means the sector is being used.

When a file is stored on the disk, the bits corresponding to the sectors used are set to 0. When the file is erased, the bits are set back to 1. That is why DOS, when it deletes a file, can be heard reading the entire file. It is determining which sectors were being used by the file so that it can free them back up. Notice that even on a newly formatted disk, sector bits 1, 2 and 3 (bits 6, 5 and 4 of byte $0A) are set to 0. These correspond to the 3 boot sectors. Likewise, the nine bits starting in byte $37 are 0 because they correspond to the sectors of the directory. These 12 sectors are thus kept from being overlaid by user files.

If the VTOC is viewed on an older disk which has had many file additions and deletions, it may be noted that the VTOC has become quite fragmented. Any file added to the disk may get stored into sectors scattered about the disk. How DOS keeps track of files spread over multiple sectors will be discussed shortly. By the way, even though the operating system recognizes sector 720 (try reading it; should be all zeroes), DOS never makes use of it. True to Murphy's Law, it adopted the number scheme of 0 to 719 instead of 1 to 720. No need to bother trying to read sector 0!

The Directory

Of all the disk data structures, probably the most important one to be acquainted with is the directory. The eight sectors following the VTOC (361-368) contain a list of all the files on the disk along with their size, starting sector, and status. Put Diskpeek into character mode and read sector 361 of the DOS disk that has several files on it. It can be seen that the name of the first file starts in byte $05 and the extension (if any) starts in byte $0D. If any of the 11 character positions of the filespec are unused, it contains a blank. Notice that the filenames start every 16 bytes, allowing eight directory entries per 128 byte sector. Thus, the maximum number of entries for the eight sectors of the directory is 64.

Now put Diskpeek in hex mode and read sector 361. The first byte of each 16 byte entry contains the status of the file. For a normal file that byte is $42, unless it is locked, in which case it has a status of $62. A deleted file has a status of $80. An anomaly occurs whenever a file is opened for output (from BASIC, perhaps) but is not closed before the computer is powered down or glitched. Since the status of an open file is $43, DOS will neither recognize the entry as "in use" nor "deleted." Even the sectors which may have been written out will not really exist on disk because the VTOC is not updated until the file is closed. The only harm done is that this bogus entry will take up space in the directory until the disk is reformatted. The second and third bytes of each entry contain the size in sectors of the file (low order byte first) while the fourth and fifth bytes specify the first sector of the file. DOS only needs to know the first sector of a file because each sector points to the next sector of the file in a process called "linking."

Linking

At this point it would be best to explain how DOS forms a data file on disk. First, the user must open an I/O channel for output to the disk, perhaps with the BASIC "OPEN" command. DOS responds by creating an entry in the directory with the specified filename and a status of $43. DOS reads the VTOC into memory and searches the disk map for the first free sector. If a free sector is found, its number is used as the starting sector in the directory entry. Now, when the user begins to output data via this I/O channel, perhaps with the BASIC "PUT" command, DOS waits until it has collected 125 bytes of user data in a buffer. Then DOS adds three special bytes of its own and outputs the sector to the disk. I call these three bytes the "sector link."

The sector link, bytes 125 to 127 of the sector, contains three pieces of information. The high order six bits of byte 125 contain a number which represents the position of the file's entry within the directory (0 to 63). DOS uses this number to check the integrity of the file. If ever this number should fail to match the position of the file's directory entry, DOS generates an error. The low order two bits of byte 125 and all of byte 126 form a pointer to the next sector of the file. A pointer is the address of a record in the computer's memory or, in this case, the address of a record on disk, the sector number.

The next sector of the file is determined by scanning the bit map of the VTOC for the next free sector, which may or may not be the next sequential sector of the disk. Thanks to the link pointers, all sectors of a file need not be contiguous sectors on the disk. The last byte of the sector link (byte 127 of the sector) contains the number of bytes used within the sector. This byte will always be $7D (125) except for the last sector of a file, which will probably be only partially filled. DOS writes out this partial sector only when the user closes the file, perhaps with the BASIC "CLOSE" command.

When an output disk file is closed, DOS writes the newly updated VTOC back out to sector 360. It then updates the file's directory entry by changing the status to $42 and filling in the file size (bytes 1 and 2) with the number of sectors used by the file. This completes the process of creating a file on disk. Now, when DOS is requested to read a file from disk, it finds the directory entry of the specified file to determine the start sector. Then, following the link pointers, it reads the file, sector by sector, until EOF (end of file) is reached, indicated by a link pointer of 0.

Equipped with a basic understanding of how a file is stored on disk, try looking at a file with Diskpeek. In character mode, first locate the name of the desired file in the directory (sectors 361-368). Then put Diskpeek in hex mode and look at the fourth and fifth byte of the entry to determine the start sector. For example, if these two bytes were "01 02" then type "$201" to read the first sector.

Observe the last three bytes of the sector and verify that the high order six bits of byte 125 correspond to the directory entry position and that byte 127 is the number of bytes used (probably $7D). Then determine the next sector of the file from the low order two bits of byte 125 and byte 126. For example, if bytes 125 and 126 are "06 02" then the next sector of the file is $202 and the file is the second entry of the directory (the first entry being entry zero). If the file is not too long, it would be instructive to follow the sector links to EOF. Once the ability of finding a file on disk and following the sector links is mastered, all that remains is to become familiar with the three types of files used by DOS.

File Types

The first type of file is not a true file, per se, because there is no entry in the directory for it. This file type includes the boot record and the directory itself. And, since the sectors which make up these files are not linked, but, instead, are related to each other sequentially, I call these records "sequentially linked files." When examining a sector of the boot record or directory, merely increase the sector number by one to get to the next sector of the record.

An example of the second type of file is that which is created with the BASIC LIST or SAVE command. This file consists of ASCII characters which either represent straight text, as in a LISTed file, or a sort of condensed text, as in a tokenized or SAVEd file. Except when viewing the sector links, the character mode of Diskpeek is best suited for examining this type of file. At this point it would be instructive to locate (in the directory of a DOS disk) a file created with the BASIC LIST command.

Upon determining the start sector, observe the file in the character mode. The BASIC program can be easily recognized. It may be noted that the carriage return-line feed character (CRLF) is displayed in its ATASCII representation (an inverse escape character) instead of being executed. Now observe a file that consists of a program that was SAVEd from BASIC. Since the text has been tokenized, the program is harder to recognize. However, certain parts of the program are not altered during the tokenization process, notably text following REM and PRINT statements. Now, having investigated ASCII files, it is time to discuss the last file type, the *binary load* file.

The binary load file is primarily used to load 6502 machine code into memory for execution. However, its format is so general that it can be used just as easily to load any type of data, including ASCII text. Locate a game or other program which is run with the BINARY LOAD option of DOS. Alternatively, create a binary load file by saving any part of memory (except ROM) with the BINARY SAVE option. Now observe the first sector of the file with Diskpeek in the hex mode.

First, notice that all binary load files start with two bytes of $FF. The next four bytes are the start and end addresses, respectively, where the data to follow will be loaded into memory. If these four bytes were "00 A0 FF BF" then the data would be loaded between the addresses of $A000 and $BFFF. I call these four bytes a *load vector*. After DOS has loaded in enough bytes to satisfy the load vector, it assumes (unless EOF is reached) that the next four bytes specify another load vector. DOS will continue inputting the file at this new address.

Upon completion of a BINARY LOAD, control will normally be passed back to the DOS menu. However, DOS can be forced to pass control to any address in memory by storing that two byte address at location $2E0. To store the two bytes, it is necessary to specify another load vector as part of the file. If, for example, it were desired to execute the program loaded in at $A000, the following load vector would be part of the file: E0 02 E1 02 00 A0. I call this specialized load vector an *autorun vector*. It achieves the same result as the RUN AT ADDRESS option of DOS. Try to find the autorun vector in the file being viewed. Although it could be at the beginning, it is most likely located at the very end of the file.

```
10 REM DISKPEEK: David Young 11/10/81
20 SETCOLOR 1,0,4:SETCOLOR 2,10,10
30 DIM HEXCHAR$(16),HEXBYTE$(2)
40 DIM HEXNUM$(113),SECTRW$(68)
50 DIM TEMP$(3),DFORM$(1)
60 ? CHR$(125):? "WAIT A FEW SECONDS..."

70 GOSUB 1130:GOSUB 970
80 GOSUB 660:RESTORE 90
```

```
90 DATA 0123456789ABCDEF
100 READ HEXCHAR$:OPEN #1,4,0,"K"
110 DFORM$="H"
120 ? CHR$(125):? "        DISKPEEK by Da
vid Young":?
130 ? "This is a disk utility for viewin
g"
140 ? "individual sectors of a disk. It"

150 ? "reads the sector specified by the
"
160 ? "user and then displays it's conte
nts"
170 ? "as a matrix of hex bytes or ATASC
II"
180 ? "characters.":?
190 ? "The sector number can be specifie
d in"
200 ? "decimal ('361') or hex ('$169').
Type"
210 ? "RETURN to toggle from one display
"
220 ? "format to the other."
230 POSITION 2,20:? CHR$(156);:? "Sector
 #";
240 INPUT HEXNUM$:IF LEN(HEXNUM$)<>0 THE
N 280
250 IF DFORM$="C" THEN DFORM$="H":GOTO 2
70
260 DFORM$="C"
270 GOSUB 770:GOTO 230
280 GOSUB 500:IF BYTE<0 OR BYTE>720 THEN
 GOSUB 350:GOTO 230
290 SECNUM=BYTE
300 GOSUB 880:IF X=1 THEN GOSUB 770
310 GOTO 230
320 REM
330 REM *** PRINT ERROR MESSAGE ***
340 REM
350 POSITION 2,19:? CHR$(156);CHR$(156);
CHR$(156);"NOT LEGAL NUMBER!":RETURN
360 REM
370 REM **** PRINT HEX BYTE ******
380 REM
390 GOSUB 430:PRINT HEXBYTE$;:RETURN
400 REM
410 REM *** HEX CONVERSION ***
420 REM
430 TEMPB=BYTE:BYTE=INT(BYTE/16)+1
440 HEXBYTE$(1,1)=HEXCHAR$(BYTE,BYTE)
450 BYTE=(TEMPB-(BYTE-1)*16)+1
460 HEXBYTE$(2,2)=HEXCHAR$(BYTE,BYTE)
470 BYTE=TEMPB:RETURN
480 REM
490 REM *** NUMBER CONVERSION ***
500 REM
510 TRAP 630:IF HEXNUM$(1,1)<>"$" THEN G
OTO 620
520 HEXNUM$=HEXNUM$(2)
530 IF LEN(HEXNUM$)=3 THEN HEXNUM$(4)=HE
XNUM$(3):HEXNUM$(3,3)=HEXNUM$(2,2):HEXNU
M$(2,2)=HEXNUM$(1,1):HEXNUM$(1,1)="0"
540 IF LEN(HEXNUM$)=2 THEN HEXNUM$(4)=HE
XNUM$(2):HEXNUM$(3,3)=HEXNUM$(1,1):HEXNU
M$(1,2)="00"
550 IF LEN(HEXNUM$)=1 THEN HEXNUM$(4)=HE
XNUM$(1):HEXNUM$(1,3)="000"
560 IF ASC(HEXNUM$(1,1))>64 THEN HEXNUM$
(1,1)=CHR$(ASC(HEXNUM$(1,1))-7)
570 IF ASC(HEXNUM$(2,2))>64 THEN HEXNUM$
(2,2)=CHR$(ASC(HEXNUM$(2,2))-7)
580 IF ASC(HEXNUM$(3,3))>64 THEN HEXNUM$
(3,3)=CHR$(ASC(HEXNUM$(3,3))-7)
590 IF ASC(HEXNUM$(4,4))>64 THEN HEXNUM$
(4,4)=CHR$(ASC(HEXNUM$(4,4))-7)
600 BYTE=(ASC(HEXNUM$(4,4))-48)+16*(ASC(
HEXNUM$(3,3))-48)+256*(ASC(HEXNUM$(2,2))
-48)+4096*(ASC(HEXNUM$(1,1))-48)
610 TRAP 40000:RETURN
620 TRAP 630:BYTE=VAL(HEXNUM$):GOTO 610
630 GOSUB 350:BYTE=-1:GOTO 610
640 REM
650 REM *** DISK READ/WRITE ***
660 REM
670 RESTORE 680:FOR K=1 TO 68:READ Q:SEC
TRW$(K,K)=CHR$(Q):NEXT K:RETURN
680 DATA 104,104,104,201,83,169,82,144
690 DATA 2,169,87,72,169,0,72,169
700 DATA 1,72,169,0,72,169,128,72
710 DATA 169,6,72,72,104,104,141,5
720 DATA 3,104,141,4,3,104,104,141
730 DATA 1,3,104,104,141,2,3,104
740 DATA 141,11,3,104,141,10,3,32
750 DATA 83,228,173,3,3,133,212,169
760 DATA 0,133,213,96
770 REM
780 REM *** DISPLAY SECTOR ***
790 REM
800 BYTE=INT(SECNUM/256):? CHR$(125)
810 ? "SECTOR # = ";SECNUM;
820 ? " ($";:GOSUB 370
830 BYTE=SECNUM-256*INT(SECNUM/256)
840 GOSUB 370:? ")"
850 IF DFORM$="H" THEN GOTO 870
860 X=USR(ADR(MEMCHAR$),1536+128):RETURN

870 X=USR(ADR(MEMHEX$),1536+128):RETURN

880 REM
890 REM *** READ SECTOR ***
900 REM
910 X=USR(ADR(SECTRW$),82,SECNUM)
```

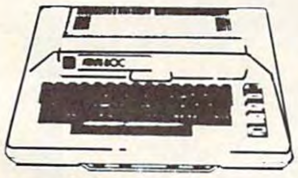
```
920 IF X=1 THEN 950
930 POSITION 2,19
940 ? "CAN'T READ SECTOR ";SECNUM;"!"
950 RETURN
960 REM
970 REM *** DISPLAY MEM IN HEX ***
980 REM
990 DIM MEMHEX$(122)
1000 RESTORE 1010:FOR K=1 TO 122:READ Q:
MEMHEX$(K,K)=CHR$(Q):NEXT K:RETURN
1010 DATA 104,104,133,229,104,133,228,16
9
1020 DATA 0,72,104,72,16,7,169,155
1030 DATA 32,164,246,104,96,169,155,32
1040 DATA 164,246,104,72,74,74,74,74
1050 DATA 201,10,48,2,105,6,105,48
1060 DATA 32,164,246,104,72,41,15,201
1070 DATA 10,48,2,105,6,105,48,32
1080 DATA 164,246,169,32,32,164,246,169
1090 DATA 32,32,164,246,104,72,168,177
1100 DATA 228,74,74,74,74,201,10,48
1110 DATA 2,105,6,105,48,32,164,246
1120 DATA 104,72,168,177,228,41,15,201
1130 DATA 10,48,2,105,6,105,48,32
1140 DATA 164,246,169,32,32,164,246,104
1150 DATA 24,105,1,72,41,7,208,204
1160 DATA 240,144
1170 REM
1180 REM *** DISPLAY MEM IN CHAR FORMAT
***
1190 REM
1200 DIM MEMCHAR$(122)
1210 RESTORE 1220:FOR K=1 TO 122:READ Q:
MEMCHAR$(K,K)=CHR$(Q):NEXT K:RETURN
1220 DATA 104,104,133,229,104,133,228,16
9
1230 DATA 0,72,104,72,16,7,169,155
1240 DATA 32,164,246,104,96,169,155,32
1250 DATA 164,246,104,72,74,74,74,74
1260 DATA 201,10,48,2,105,6,105,48
1270 DATA 32,164,246,104,72,41,15,201
1280 DATA 10,48,2,105,6,105,48,32
1290 DATA 164,246,169,32,32,164,246,169
1300 DATA 32,32,164,246,169,1,141,254
1310 DATA 2,104,72,168,177,228,201,155
1320 DATA 208,11,169,0,141,254,2,169
1330 DATA 219,133,93,169,31,32,164,246
1340 DATA 169,32,32,164,246,169,32,32
1350 DATA 164,246,169,0,141,254,2,104
1360 DATA 24,105,1,72,41,7,208,204
1370 DATA 240,144
```

©

ATARI

from

SOFTWARE STREET


$729.00

$429.00

$65.00

We carry a wide variety of hardware and software to accommodate all ATARI needs.

Call or write for FREE catalog

SOFTWARE STREET

3392 Clipper Dr.
Chino, CA 91710
(714) 591—3061
call for shipping charges

ATARI
A Warner Communications Company

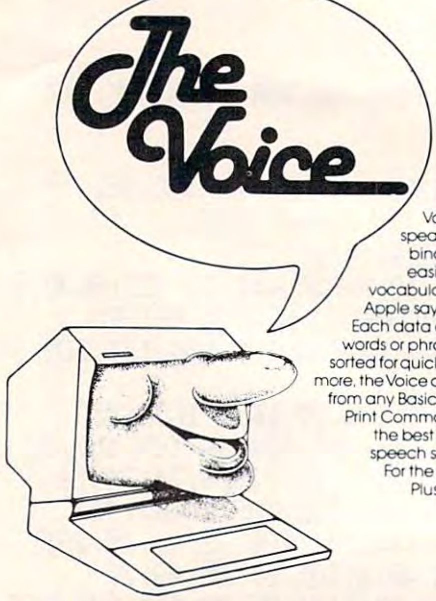
Single Board Computers	Bare Board	Kit	Assm.
80-153 1k RAM 2k EPROM	$19.95	$ 89.95	$110.95
80-280 1k RAM 2k EPROM	$19.95	$119.95	$129.95
81-260 2k RAM 4k EPROM	$39.95	$149.95	$199.95
81-030 4k RAM 16k EPROM	$89.95	*$249.95	$349.95
8085 3 chip system	$24.95		
8088 5 chip system	$29.95		
Apple II boards	**Bare Board**	**Kit**	**Assm.**
79-295 Parallel I/O	$19.95	$59.95	$ 69.95
81-132 A-D 16 Channel	$29.95	$69.95	$ 89.95
80-085 12k EPROM (2716)	$39.95	$49.95	$ 59.95
80-244 EPROM PROGRAMMER	$24.95	$39.95	$ 49.95
80-144 Display add + data	$25.95	$42.95	$ 49.95
80-297 Parallel Printer Interface		$69.95	$ 79.95
Extender Board		$12.95	
81-088 Speech Synthesizer (Votrax SC-01)			$139.90
Other Products	**Bare Board**	**Kit**	**Assm.**
CRT Controller	$39.95		
79-287 A-D and D-A	$19.95	$59.95	$ 79.95
79-282 S S Switch 1 ch.	$ 6.95	$10.95	$ 13.95
79-282 S S Switch 4 ch.	$24.95	$39.95	$ 49.95
80-160 +5 Volt 500 Ma.	$ 8.95	$16.95	$ 20.95
80-161 ± 12 Volt 120 Ma.	$ 8.95	$18.95	$ 22.95
81-120 Parallel Input Speech Synthesizer			$149.90

Send for complete Catalog and OEM pricing.
*Partially Populated price.

JOHN BELL ENGINEERING, INC.

ALL PRODUCTS ARE AVAILABLE FROM **JOHN BELL ENGINEERING**
MC P.O. BOX 338 • REDWOOD CITY, CA 94064 VISA
ADD SALES TAX IN CALIFORNIA
ADD 5% SHIPPING & HANDLING • 3% FOR ORDERS OVER $100
10% OUTSIDE U.S.A.

SEND FOR CATALOG **(415) 367-1137**



The Voice gives your Apple the power of speech! Use the standard Voice vocabulary to speak an endless combination of phrases, or easily record your own vocabulary and make your Apple say anything you like. Each data disk stores up to 80 words or phrases which can be sorted for quick reference. What's more, the Voice allows you to speak from any Basic program by using Print Commands. Guaranteed the best and easiest to use speech software available. For the Apple II or Apple II Plus with 48K. ($39.95)

from the leader in quality software

MUSE SOFTWARE™

Apple II is a trademark of Apple Computer Corp

330 N. CHARLES STREET
BALTIMORE, MD 21201
(301) 659-7212

Call or write for information and the name of your nearest **MUSE** dealer

ECHO][™ *SPEECH SYNTHESIZER*

UNLIMITED VOCABULARY AT YOUR FINGERTIPS

Give your Apple* something to talk about with an ECHO][Speech Synthesizer. The ECHO][offers intelligible voice-output while using a minimum of RAM. The ECHO][uses LPC technology pioneered by Texas Instruments, coupled with a phoneme-based operating system allowing you to create any vocabulary desired. Variable stress, pitch and volume let your Apple ask questions or make exclamations while also allowing for optimal quality. Speech can easily be added to BASIC programs with PRINT statements.

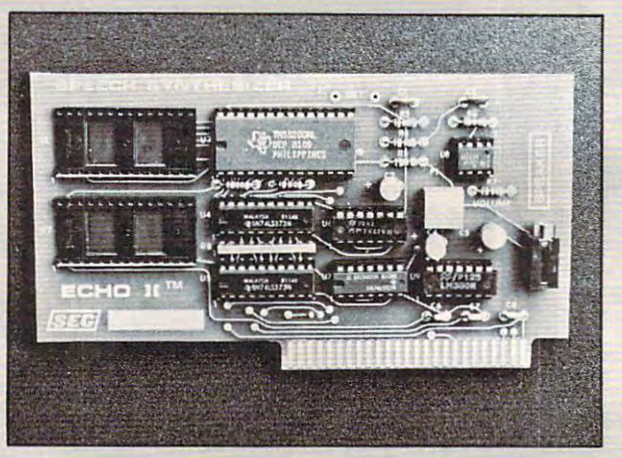
The applications of synthesized speech are limited only by your imagination. From educational programs to games to business applications, speech-output adds a new dimension to your Apple's capabilities.

The ECHO][Speech Synthesizer comes complete with speaker and cable, instruction manual, speech editing features and a sample vocabulary. The ECHO][requires 48K, Applesoft and at least one disk drive. Suggested list price is $225.

For further information, contact your dealer or Street Electronics Corporation.

Dealer inquiries welcome.

*** Trademark of Apple Computer Company**



SEC **STREET ELECTRONICS CORPORATION**

3152 E. La Palma Ave., Suite D
Anaheim, CA 92806 **(714) 632-9950**

Apple Addresses

Bill Grimm
Mountain View, CA

The Apple II uses three types of addressing depending upon the language being used. Apple's machine language uses hexadecimal addresses in the range from $0000 to $FFFF. Its Floating Point BASIC language uses decimal addresses in the range from 0 to 65535. Its Integer BASIC uses decimal addresses in the range from 0 to 32767 to -32767 to -1. This means that, if you want to address a particular memory location, you must choose the correct address for the language you are using. Since I program in all three languages and my references are a mixture from all three, I needed an address cross-reference program. So I wrote "Apple Addresses."

"Apple Addresses" can be used "as is" to convert one language's address to another's, and to give the high and low byte values which need to be poked into a BASIC program to store that address. Alternatively, you could extract the subroutines in Apple Addresses which convert between hex and decimal numbers and insert them in your own program. See the last paragraph of this article for more details.

The program begins by asking the user which of the six possible conversions he would like to make. This is followed by a request to select the way the results of the conversions are to be displayed. There are four possible displays:

1. single conversions displayed on the monitor one at a time.

2. Single conversions printed out on a Silentype printer* one at a time.

3. a range of conversions displayed on the monitor.

4. a range of conversions printed out on a Silentype printer*.

*With slight program modifications other printers could be used.

Subroutines

"Apple Addresses" makes extensive use of subroutines. This helps in organizing the program as well as making it shorter and easier to debug. The controlling or EXECutive routine is called Apple Addresses – Exec. It starts on line 100 and goes to line 310. Since a picture is worth a thousand words, I made what I call a *balloon diagram* (Figure 1) to show how data flows through the program. These are the conventions I used to make the diagram;

1. Each balloon represents a subroutine. The name of the subroutine and the line numbers where it is located are placed in the balloon.

2. Data flows through a subroutine in the direction of the arrows on the outside of the balloon.

3. Data flows between subroutines in the direction of the arrows on the *strings*.

4. If conditions are placed on what data flows through a subroutine, these conditions are written in along the *strings*.

As an additional aid for understanding how the program works I have included the following variable descriptions list:

A( ) — each A(I) holds the decimal equivalent value of the Ith hexadecimal numeral in the hex number being created from a decimal number — appropriate numbers are then added to convert these to ASCII codes.

A$( ) — holds the characters represented by the ASCII codes in A( ).

CHOICE — holds the number of the conversion chosen — see lines 120 to 178.

DVL — holds the decimal value of the number being converted — may be either FP or INT decimal.

DVL$ — is the string equivalent of DVL and is used in the output routines.

FLAG — if flag = 1 then an invalid number was entered and the program returns to get a new number.

FRST — holds the FP Basic address equivalent of the lowest address in the selected range.

FRST$ — holds the smallest address chosen — this address is then processed and stored in FRST.

HVL$ — holds the hex number selected or the hex number resulting from the conversion — if no hex numbers are involved then it holds the converted decimal number.

LST — holds the FP Basic address equivalent of the largest address in the selected range.

LST$ — holds the largest address chosen — this address is then processed and stored in LST.

N — holds the decimal equivalent of each hex numeral in a hex number being converted to a decimal number.

PHI% — holds the number that would be poked into the high byte when placing the address into memory.

PLO% — holds the number that would be poked into the low byte when placing the address into memory.

POK — holds the address from which PLO% and PHI% are derived.

SELECT — holds the type of output selected — see lines 462 to 470.

STP — holds the positive decimal stepping interval chosen.

STP$ — holds the stepping interval chosen which is later changed and stored in STP.

TB — the horizontal tab value desired.

TN — holds the intermediate numbers of the decimal address that is being converted into a hex address.

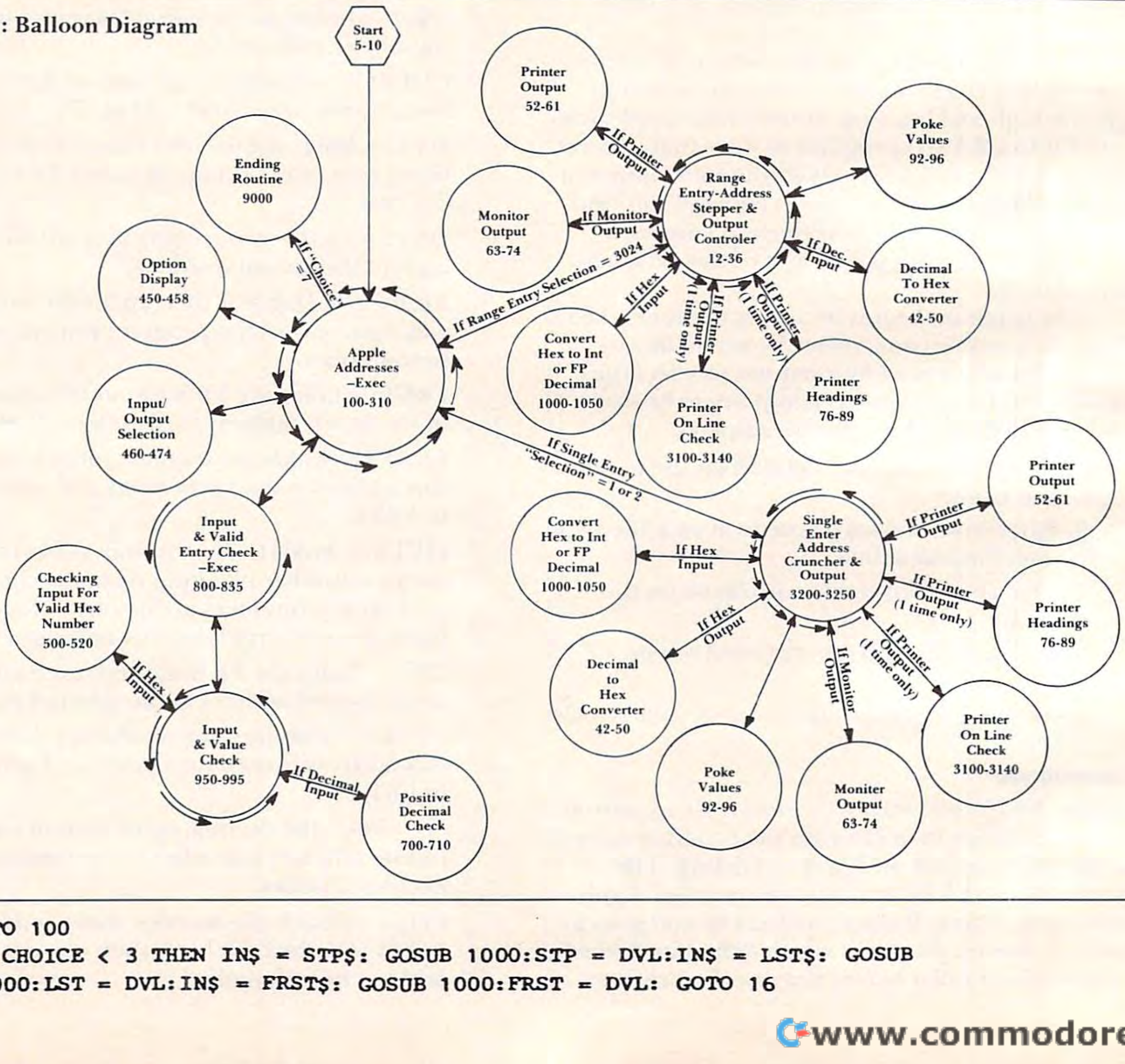
VTB — used to control the vertical tabbing of the monitor output.

Some Suggestions

I have found that the easiest way to debug a program while I am entering it is to first type in the EXEC program. Then, if I place return statements at all the branching locations, I can check the EXEC for bugs. Once the EXEC is free of bugs, I add one subroutine at a time in the order that the EXEC uses them, checking for bugs as I go.

If you have a need for subroutines which convert numbers from hex to decimal or from decimal to hex, two subroutines in this program may be of help. The first is called "decimal to hex converter" (lines 42 to 50). The input to this routine is TN which must hold a positive decimal number <65536. The output is HVL$ which holds the hex equivalent to the number in TN. The second is called "convert hex to INT or FP decimal" (lines 1000 to 1050). The input to this routine is HVL$ which must hold a hex number <=$FFFF and choice. If choice = 1 then you get the positive decimal equivalent. Otherwise you get Int BASIC's equivalent. The output is a decimal number in DVL.

Figure 1: Balloon Diagram


Start 5-10
Ending Routine 9000
Option Display 450-458
Apple Addresses –Exec 100-310
Input/ Output Selection 460-474
If "Choice" =7
If Range Entry Selection = 3024
Printer Output 52-61
If Printer Output
Monitor Output 63-74
If Monitor Output
Range Entry-Address Stepper & Output Controler 12-36
Poke Values 92-96
If Dec. Input
Decimal To Hex Converter 42-50
If Hex Input
Convert Hex to Int or FP Decimal 1000-1050
If Printer Output (1 time only)
If Printer Output (1 time only)
Printer On Line Check 3100-3140
Printer Headings 76-89
If Single Entry "Selection" = 1 or 2
Input & Valid Entry Check –Exec 800-835
Checking Input For Valid Hex Number 500-520
If Hex Input
Input & Value Check 950-995
If Decimal Input
Positive Decimal Check 700-710
Convert Hex to Int or FP Decimal 1000-1050
If Hex Input
Single Enter Address Cruncher & Output 3200-3250
If Printer Output
Printer Output 52-61
If Printer Output (1 time only)
Printer Headings 76-89
If Hex Output
Decimal to Hex Converter 42-50
If Monitor Output
If Printer Output (1 time only)
Printer On Line Check 3100-3140
Poke Values 92-96
Moniter Output 63-74

```
10  GOTO 100
12  IF CHOICE < 3 THEN IN$ = STP$: GOSUB 1000:STP = DVL:IN$ = LST$: GOSUB
     1000:LST = DVL:IN$ = FRST$: GOSUB 1000:FRST = DVL: GOTO 16
```

```
14 STP =  VAL (STP$):LST =  VAL (LST$):FRST =  VAL (FRST$)
16 VTB = 7:TB = 1: IF SELECT = 4 THEN  GOSUB 3100: POKE  - 12526,83: PR#
   1: PRINT : PRINT "CONVERTING FROM ";: ON CHOICE GOSUB 76,78,80,82,84
   ,86: POKE  - 12526,80
18  IF LST < 0 THEN LST = LST + 65536: IF FRST < 0 THEN FRST = FRST + 655
   36
19  FOR DVL = FRST TO LST STEP STP: IF CHOICE <  > 4 OR CHOICE <  > 6 THEN
   TN = DVL: GOSUB 42
20  IF CHOICE = 3 AND DVL > 32767 OR CHOICE = 4 AND DVL > 32767 OR CHOICE
    = 2 AND DVL > 32767 OR CHOICE = 6 AND DVL > 32767 THEN DVL = DVL -
   65536
22  IF CHOICE = 4 THEN HVL$ =  STR$ (DVL): IF DVL < 0 THEN HVL$ =  STR$ (
   DVL + 65536)
24  IF CHOICE = 6 THEN HVL$ =  STR$ (DVL): IF DVL < 0 THEN DVL = DVL + 65
   536
26  GOSUB 92
28  IF SELECT = 4 THEN  GOSUB 52: GOTO 32
30  GOSUB 62
32  IF DVL < 0 THEN DVL = DVL + 65536
34  NEXT DVL: IF SELECT = 4 THEN  PRINT : PR# 0
36  RETURN
42 HVL$ = "": FOR I = 4 TO 1 STEP  - 1:A(5 - I) =  INT (TN / (16 ^ (I - 1
   ))):TN = TN - (A(5 - I) * (16 ^ (I - 1))): NEXT I
44  FOR I = 1 TO 4: IF A(I) < 10 THEN A(I) = A(I) + 48: GOTO 48
46 A(I) = A(I) + 55
48 A$(I) =  CHR$ (A(I)):HVL$ = HVL$ + A$(I): NEXT I
50  RETURN
52 DVL$ =  STR$ (DVL): IF CHOICE < 3 THEN 58
54  PRINT  SPC( 6 -  LEN (DVL$));DVL$;: IF CHOICE = 5 OR CHOICE = 3 THEN
    PRINT ">$";HVL$; SPC( 1);: GOTO 59
56  PRINT ">"; SPC( 6 -  LEN (HVL$));HVL$;: GOTO 59
58  PRINT " $"; SPC( 4 -  LEN (HVL$));HVL$;">"; SPC( 6 -  LEN (DVL$));DVL
   $;
59  PRINT  SPC( 9 -  LEN (PLO$));PLO$; SPC( 14 -  LEN (PHI$));PHI$;:TB =
   TB + 39: IF TB > 42 OR SELECT = 2 THEN TB = 1: PRINT
60  HTAB TB: IF TB = 40 THEN  PRINT  SPC( 3);
61  RETURN
62  REM
63 DVL$ =  STR$ (DVL): VTAB VTB: HTAB TB: IF CHOICE < 3 THEN 68
64  PRINT  SPC( 6 -  LEN (DVL$));DVL$;: IF CHOICE = 5 OR CHOICE = 3 THEN
    PRINT ">$";HVL$; SPC( 2);: GOTO 70
66  PRINT ">"; SPC( 6 -  LEN (HVL$));HVL$; SPC( 1);: GOTO 70
68  PRINT "$0000>";: HTAB TB + 5 -  LEN (HVL$): PRINT HVL$;: HTAB TB + 12
    -  LEN (DVL$): PRINT DVL$; SPC( 2);
70  PRINT  SPC( 8 -  LEN (PLO$));PLO$; SPC( 14 -  LEN (PHI$));PHI$:VTB =
   VTB + 1: IF VTB > 23 THEN  HTAB 3: INPUT "PRESS <RETURN> TO CLEAR SC
   REEN";IN$: HOME :VTB = 6:TB = 1: GOTO 72
71  GOTO 74
72  IF IN$ = "Q" THEN  POP : GOTO 100
73  IF SELECT = 3 THEN VTB = 7
74  RETURN
76  PRINT "HEX TO FP DECIMAL": GOSUB 88: RETURN
78  PRINT "HEX TO INT DECIMAL": GOSUB 88: RETURN
80  PRINT "INT DECIMAL TO HEX": GOSUB 88: RETURN
82  PRINT "INT DECIMAL TO FP DECIMAL": GOSUB 88: RETURN
84  PRINT "FP DECIMAL TO HEX": GOSUB 88: RETURN
86  PRINT "FP DECIMAL TO INT DECIMAL": GOSUB 88: RETURN
88  IF SELECT = 2 THEN  PRINT : PRINT " CONVERSION   POKE LO BYTE  POKE H
   I BYTE": RETURN
```

```
89  PRINT : PRINT " CONVERSION   POKE LO BYTE  POKE HI BYTE  CONVERSION
     POKE LO BYTE  POKE HI BYTE": RETURN
92 POK = DVL: IF POK < 0 THEN POK = POK + 65536
94 PHI% = POK / 256:PLO% = POK - PHI% * 256
96 PHI$ =  STR$ (PHI%):PLO$ =  STR$ (PLO%): RETURN
100  POKE  - 16298,0: TEXT : HOME :FLAG = 0
110  VTAB 7
120  PRINT " 1. CONVERT HEX ADDRESSES TO FP BASIC": PRINT
130  PRINT " 2. CONVERT HEX ADDRESSES TO INT BASIC": PRINT
135  PRINT " 3. CONVERT INT BASIC ADDRESSES TO HEX": PRINT
140  PRINT " 4. CONVERT INT BASIC ADDRESSES TO FP": PRINT
150  PRINT " 5. CONVERT FP BASIC ADDRESSES TO HEX": PRINT
160  PRINT " 6. CONVERT FP BASIC ADDRESSES TO INT": PRINT
162  PRINT " 7. QUIT": PRINT
165  PRINT : PRINT "NOTE: ENTERING A 'Q' AT ANY POINT           RETURNS
      YOU TO THIS MENU."
170  VTAB 4: INPUT "CHOOSE ONE:";IN$
175  IF IN$ = "7" THEN 9000
178 CHOICE =  VAL (IN$): IF CHOICE < 1 OR CHOICE > 6 THEN 100
180  GOSUB 450: GOSUB 460: HOME : VTAB 1: HTAB 13: ON SELECT GOTO 190,195
     ,200,210
190  PRINT ": SINGLE ENTRY : MONITOR": GOTO 220
195  PRINT ": SINGLE ENTRY : PRINTER": GOTO 220
200  PRINT ": RANGE ENTRY : MONITOR": GOTO 220
210  PRINT ": RANGE ENTRY : PRINTER"
220  HOME : IF SELECT < 3 THEN  PRINT "ENTER NUMBER": GOTO 250
230  PRINT "FIRST NUMBER";: HTAB 22: PRINT "LAST NUMBER"
240  PRINT "STEPPING INTERVAL"
250  FOR I = 0 TO 39: PRINT  CHR$ (45);: NEXT I: PRINT " CONVERSION   POK
     E LO BYTE  POKE HI BYTE": POKE 34,6: IF SELECT < 3 THEN  POKE 34,5
260  HOME
280 CNT = 0:TB = 1:VTB = 7: IF SELECT < 3 THEN VTB = 6
290  GOSUB 800
300  ON SELECT GOSUB 3200,3200,12,12: IF SELECT < 3 THEN 290
310  VTAB 24: HTAB 5: CALL  - 868: INPUT "PRESS <RETURN> TO CONTINUE.";IN
     $: GOTO 100
450  HOME : HTAB 4: ON CHOICE GOSUB 452,456,458,455,454,457: FOR I = 0 TO
     39: PRINT  CHR$ (45);: NEXT I: POKE 34,2: RETURN
452  PRINT "HEX->FP": RETURN
454  PRINT "FP->HEX": RETURN
455  PRINT "INT->FP": RETURN
456  PRINT "HEX->INT ": RETURN
457  PRINT "FP->INT": RETURN
458  PRINT "INT->HEX": RETURN
460  HOME : VTAB 8
462  PRINT "  1. SINGLE ENTRY - MONITOR OUTPUT": PRINT
463  PRINT "  2. SINGLE ENTRY - PRINTER OUTPUT": PRINT
464  PRINT "  3. RANGE  ENTRY - MONITOR OUTPUT": PRINT
466  PRINT "  4. RANGE  ENTRY - PRINTER OUTPUT": PRINT
468  VTAB 6: INPUT "CHOOSE ONE:";IN$: IF IN$ = "Q" THEN  POP : GOTO 100
470 SELECT =  VAL (IN$)
472  IF SELECT < 1 OR SELECT > 4 THEN 460
474  RETURN
500  FOR I = 1 TO  LEN (IN$): IF  ASC ( MID$ (IN$,I,1)) > 70 OR  ASC ( MID$
     (IN$,I,1)) < 48 THEN 520
510  IF  ASC ( MID$ (IN$,I,1)) > 57 AND  ASC ( MID$ (IN$,I,1)) < 65 THEN  520
512  NEXT I: RETURN
520 FLAG = 1: RETURN
700  FOR I = 1 TO  LEN (IN$)
```

```
705  IF  ASC ( MID$ (IN$,I)) > 57 OR . ASC ( MID$ (IN$,I)) < 48 THEN 710
709  NEXT I: RETURN
710 FLAG = 1: RETURN
800  IF SELECT > 2 THEN 815
805  VTAB 3: HTAB 13: CALL  - 868: GOSUB 950: IF FLAG = 1 THEN FLAG = 0: GOTO
     805
810  GOTO 835
815  VTAB 3: HTAB 13: POKE 33,21: CALL  - 868: GOSUB 950:FRST$ = IN$: POKE
     33,40: IF FLAG = 1 THEN FLAG = 0: GOTO 815
820  VTAB 3: HTAB 33: CALL  - 868: GOSUB 950:LST$ = IN$: IF FLAG = 1 THEN
     FLAG = 0: GOTO 820
825  VTAB 4: HTAB 18: CALL  - 868: GOSUB 950:STP$ = IN$: IF DVL < 0 THEN
     FLAG = 1
830  IF FLAG = 1 THEN FLAG = 0: GOTO 825
835  RETURN
950  IF CHOICE > 2 THEN 970
955  INPUT "=$";IN$: IF IN$ = "Q" THEN  POP : POP : GOTO 100
957  IF IN$ = "" THEN FLAG = 1: GOTO 995
960  IF  LEN (IN$) > 4 THEN FLAG = 1: GOTO 995
965  GOSUB 500: GOTO 995
970  INPUT "=";IN$: IF IN$ = "Q" THEN  POP : POP : GOTO 100
972  IF IN$ = "" THEN FLAG = 1: GOTO 995
975  IF CHOICE < 5 AND  VAL (IN$) <  - 32767 THEN FLAG = 1: GOTO 995
977  IF CHOICE < 5 AND  VAL (IN$) > 32767 THEN FLAG = 1: GOTO 995
980  IF CHOICE > 4 AND  VAL (IN$) < 0 THEN FLAG = 1: GOTO 995
983  IF CHOICE > 4 AND  VAL (IN$) > 65535 THEN FLAG = 1: GOTO 995
985 DVL =  VAL (IN$): IF DVL < 0 THEN IN$ =  MID$ (IN$,2): GOSUB 700:IN$ =
      STR$ (DVL + 65536): GOTO 995
990  GOSUB 700
995  RETURN
1000 HVL$ = IN$
1010 DVL = 0: FOR I = 1 TO  LEN (IN$): IF  ASC ( MID$ (IN$,I,1)) > 64 THEN
     N =  ASC ( MID$ (IN$,I,1)) - 55
1018  IF  ASC ( MID$ (IN$,I,1)) < 64 THEN N =  ASC ( MID$ (IN$,I,1)) - 48
1020 DVL = DVL + N * 16 ^ ( LEN (IN$) - I): NEXT I
1030  IF CHOICE = 1 THEN 1050
1040  IF DVL > 32767 THEN DVL = DVL - 65536
1050  RETURN
3100  FOR I = 1 TO 7
3110 J =  - 16384 + 256 * I
3120  IF  PEEK (J + 23) = 201 AND  PEEK (J + 55) = 207 AND  PEEK (J + 76)
      = 234 THEN  RETURN
3130  NEXT I
3140  HOME : VTAB 10: PRINT "NO SILENTYPE PRINTER INSTALLED.": PRINT "SEL
     ECTION ABORTED!": FOR K = 1 TO 3000: NEXT K: POP : RETURN
3200  IF CHOICE < 3 THEN  GOSUB 1000: GOSUB 92: GOSUB 62: GOTO 3230
3210  IF CHOICE = 3 OR CHOICE = 5 THEN TN =  VAL (IN$): GOSUB 42: GOSUB 9
     2: GOSUB 62: GOTO 3230
3220 HVL$ = IN$: IF CHOICE = 6 AND  VAL (IN$) > 32767 THEN HVL$ =  STR$ (
     DVL - 65536)
3225  GOSUB 92: GOSUB 62
3230  IF SELECT = 2 AND CNT = 0 THEN  GOSUB 3100: POKE  - 12526,83: PR# 1
     : PRINT : PRINT "CONVERTING FROM ";: ON CHOICE GOSUB 76,78,80,82,84,
     86:CNT = CNT + 1
3240  IF SELECT = 2 THEN  PR# 1: GOSUB 52: PR# 0
3250  RETURN
9000  POKE  - 16300,0: POKE  - 16298,0: TEXT : CALL  - 936: POKE  - 16368
     ,0: END
```

©

More VIC Maps

Jim Butterfield
Toronto, Canada

Editor's Note: For more, see Jim's VIC maps in last month's issue, **COMPUTE!** *#20. — RTM*

It's interesting to look at the innards of the VIC. In some ways, it's much like the PET/CBM and many things are quite recognizable. But new things have crept in, too: some are associated with new features such as color, others are there to implement advanced ideas such as an improved INPUT statement. Inner-space explorers will recognize many familiar landmarks.

The most noticeable new feature is the massive tables of vectors and links that have been implemented in page three. In hopes of explaining things better, I am using the terms rather carefully. Both vectors and links are addresses in RAM. An advanced application program can use these addresses, or even change them; and this gives the VIC remarkable programming flexibility. The term "Link" is used when the address is normally used to connect adjacent code; in this case, it doesn't affect the program flow until the link is broken with a new address. A vector, on the other hand, is used as a jump point, and the normal program jumps somewhere else through the vector. In other words, a ROM program hits a link point and normally keeps going; it hits a vector point and branches.

I wish Commodore had chosen to keep VIC addresses compatible with those in the PET/CBM. If they had done so, many programs would have been portable between machines with no coding changes at all. But that's wishful thinking and, since many things are still the same style, it's not a serious hardship to trim up the PEEK and POKE addresses for transfer to the VIC.

I have inserted the "normal" address contents of many of the links/vectors in the brackets behind the description; they may not be valid for current machines, but a serious user can easily PEEK them himself.

The input and output ports are somewhat congested. There are almost as many I/O bits available as on the PET/CBM, but extra features such as joysticks and RS232 have caused a bit of a crunch.

The Video Interface Chip (VIC) itself is a remarkable piece of electronics. I hope my chart helps; but a full description can only be obtained in Commodore's technical reference.

I haven't noted the standard Jump Table in this map. Near the top of both the PET and the VIC are a series of standard locations to allow inputting, outputting, checking the stop key, and other jobs. Users familiar with their use in the PET/CBM will be pleased to know that the Jump Table is exactly the same in the VIC. All of the old favorites, such as FFD2 for PRINT and FFE4 for GET are still there.

Beginners shouldn't be scared by the mass of technical detail given here. The VIC can be used effectively without any of this information. But for those who love to tinker with the innards of the machine, there's a lifetime of experimental PEEKing and POKEing to be done; this map will help direct your efforts.

VIC Zero Page Memory Map

Hex	Decimal	Description
0000-0002	0-2	USR jump
0003-0004	3-4	Float-Fixed vector
0005-0006	5-6	Fixed-Float vector
0007	7	Search character
0008	8	Scan-quotes flag
0009	9	TAB column save
000A	10	0=LOAD, 1=VERIFY
000B	11	Input buffer pointer/# subscrpt
000C	12	Default DIM flag
000D	13	Type: FF=string, 00=numeric
000E	14	Type: 80=integer, 00=floating point
000F	15	DATA scan/LIST quote/memry flag
0010	16	Subscript/FNx flag
0011	17	0=INPUT;$40=GET;$98=READ
0012	18	ATN sign/Comparison eval flag
0013	19	Current I/O prompt flag
0014-0015	20-21	Integer value

Introducing

nüFEKOP

SOFTWARE PEOPLE

Games for VIC 20 available now

ALL 5 FOR $35.00 OR $9.99 APIECE

VIKMAN - you guessed it - you won't believe it

ALIEN PANIC - dig holes through brick floors of three story laddered maze to trap alien

DODGE CARS - fast fun hard to quit playing - keeps top 5 scores and compares with last 50

BLOW UP - you and a live mine field filled with booty - get as much as you can in two minutes

BANK ROBBERS - just try to make crime pay - before the police come

ALSO - Search, Krazy Kong and many more send $.50 for more information.

***nüFEKOP**

P.O. BOX 156, SHADY COVE, OREGON 97539-0156

C.O.D. ORDERS CALL (503) 878-2113

SPECIFY KEYBOARD OR JOYSTICKS

VIC is a registered trademark for Commodore Bus. Machines

*NUFEKOP: Druid origin, concerned with the putting of a extrordinarily large amount in a small pocket or enclosure; possible mystical connotations.

VIC=20 SOFTWARE

PIPER-THE MUSIC MACHINE

Here's the simplest way yet to compose, conduct and play music on your VIC. Complete music capability — play all note values, list and edit your music, easily handle repeats and rests. Written in fast machine language. Includes sample composition and manual. Does not require additional memory.

VIGIL INTERACTIVE GRAPHICS/GAME LANGUAGE FOR THE VIC

Create your own games with easy to learn, powerful language. Color & enhanced sound. With manual and 9 games including Breakout, SpaceWar, AntiAircraft, U.F.O., Concentration, Maze. Requires 3K memory expander.

HiRes/Multi-color Graphics

Add HiRes & Multi-Color grahics to VIC BASIC. Set colors, plot points, draw lines and boxes and write text to HiRes or Multi-Color screen. HiRes 104 x 152 points and Multi-Color 52 x 76. Does not require additional memory.

	U.S. & CANADA	FOREIGN
PIPER on cassett	$25	$30
PIPER User's Manual (refundable with software order)	$ 5	$ 7
PIPER 6502 Assembly Listing	$25	$30
VIGIL (requires a 3K memory expander) on cassette	$35	$40
VIGIL User's Manual (refundable with software order)	$10	$12
VIGIL 6502 Assembly Listing	$25	$30
HiRes/Multi-Color package for the VIC on cassette	$15	$20
HiRes/Multi-Color documentation	$ 3	$ 5
HiRes/Multi-Color 6502 Assembly Listing	$10	$15

Order prepaid, MC or VISA only (U.S. dollars). Add $3.00 for diskette.

ABACUS SOFTWARE
P. O. Box 7211
Grand Rapids, Michigan 49510
616 / 241-5510

VISA

master charge

Verbatim

flexible disks

Call Free **(800) 235-4137** for prices and information. Dealer inquiries invited. C.O.D. and charge cards accepted.

PACIFIC EXCHANGES
100 Foothill Blvd.
San Luis Obispo, CA
93401. In Cal. call
(800) 592-5935 or
(805) 543-1037.

VISA

ATTENTION

***VIC 20 OWNERS**

QUANTUM DATA INCORPORATED

"THE PERIPHERAL PEOPLE"

Makers of the following quality add-ons for the VIC 20*

RAM Memory Expansion from 0K to 64K
64 or 80 Column Video Expansion
PROM Simulator for saving programs in power-off mode
21-Key Numeric Pad for high speed data entry
3 Position Expansion Chassis
PROM Programmers and PROM/ROM Expansion
AND MORE!!!

Ask your local Commodore Dealer about our products or phone or write for Free Catalogue:

QUANTUM DATA, INC.

3001 Redhill Ave.
Bldg. 4, Suite 105
Costa Mesa, CA 92626

(714) 754-1945

*a Commodore Trademark

Dealer inquiries welcomed

MALA

Educational Programs For Your Pet For Only

$48

For 8 Issue Subscriptions

Each "MALA" comes to you on cassette with 4 brand new programs to load and run on your Pet.

• PET • COMMODORE • VIC •

COMM*DATA SYSTEMS, INC.

P.O. BOX 325
MILFORD, MICHIGAN 48042
(313) 685-0113

• PET • COMMODORE • VIC •

GAMES FOR YOUR VIC

VIC TREK • HANGMAN
TANK • YAHTZEE
DAMSEL • NAB PLUS
AND MANY MORE

only **$7.95** Each

FREE Program Summaries available

0016	22	Pointer: temporary strg stack
0017-0018	23-24	Last temp string vector
0019-0021	25-33	Stack for temporary strings
0022-0025	34-37	Utility pointer area
0026-002A	38-42	Product area for multiplication
002B-002C	43-44	Pointer: Start-of-Basic
002D-002E	45-46	Pointer: Start-of-Variables
002F-0030	47-48	Pointer: Start-of-Arrays
0031-0032	49-50	Pointer: End-of-Arrays
0033-0034	51-52	Pointer: String-storage(moving down)
0035-0036	53-54	Utility string pointer
0037-0038	55-56	Pointer: Limit-of-memory
0039-003A	57-58	Current Basic line number
003B-003C	59-60	Previous Basic line number
003D-003E	61-62	Pointer: Basic statement for CONT
003F-0040	63-64	Current DATA line number
0041-0042	65-66	Current DATA address
0043-0044	67-68	Input vector
0045-0046	69-70	Current variable name
0047-0048	71-72	Current variable address
0049-004A	73-74	Variable pointer for FOR/NEXT
004B-004C	75-76	Y-save; op-save; Basic pointer save
004D	77	Comparison symbol accumulator
004E-0053	78-83	Misc work area, pointers, etc
0054-0056	84-86	Jump vector for functions
0057-0060	87-96	Misc numeric work area
0061	97	Accum#1: Exponent
0062-0065	98-101	Accum#1: Mantissa
0066	102	Accum#1: Sign
0067	103	Series evaluation constant pointer
0068	104	Accum#1 hi-order (overflow)
0069-006E	105-110	Accum#2: Exponent, etc.
006F	111	Sign comparison, Acc#1 vs #2
0070	112	Accum#1 lo-order (rounding)
0071-0072	113-114	Cassette buff len/Series pointer
0073-008A	115-138	CHRGET subroutine; get Basic char
007A-007B	122-123	Basic pointer (within subrtn)
008B-008F	139-143	RND seed value
0090	144	Status word ST
0091	145	Keyswitch PIA: STOP and RVS flags
0092	146	Timing constant for tape
0093	147	Load=0, Verify=1
0094	148	Serial output: deferred char flag
0095	149	Serial deferred character
0096	150	Tape EOT received
0097	151	Register save
0098	152	How many open files
0099	153	Input device, normally 0
009A	154	Output CMD device, normally 3
009B	155	Tape character parity
009C	156	Byte-received flag
009D	157	Direct=$80/RUN=0 output control
009E	158	Tp Pass 1 error log/char buffer
009F	159	Tp Pass 2 err log corrected
00A0-00A2	160-162	Jiffy Clock HML
00A3	163	Serial bit count/EOI flag

00A4	164	Cycle count
00A5	165	Countdown,tape write/bit count
00A6	166	Tape buffer pointer
00A7	167	Tp Wrt ldr count/Rd pass/inbit
00A8	168	Tp Wrt new byte/Rd error/inbit cnt
00A9	169	Wrt start bit/Rd bit err/stbit
00AA	170	Tp Scan;Cnt;Ld;End/byte assy
00AB	171	Wr lead length/Rd checksum/parity
00AC-00AD	172-173	Pointer: tape bufr, scrolling
00AE-00AF	174-175	Tape end adds/End of program
00B0-00B1	176-177	Tape timing constants
00B2-00B3	178-179	Pntr: start of tape buffer
00B4	180	1=Tp timer enabled; bit cnt
00B5	181	Tp EOT/RS232 next bit to send
00B6	182	Read character error/outbyte buf
00B7	183	# characters in file name
00B8	184	Current logical file
00B9	185	Current secndy address
00BA	186	Current device
00BB-00BC	187-188	Pointer to file name
00BD	189	Wr shift word/Rd input char
00BE	190	# blocks remaining to Wr/Rd
00BF	191	Serial word buffer
00C0	192	Tape motor interlock
00C1-00C2	193-194	I/O start adds
00C3-00C4	195-196	Kernel setup pointer
00C5	197	Last key pressed
00C6	198	# chars in keybd buffer
00C7	199	Screen reverse flag
00C8	200	End-of-line for input pointer
00C9-00CA	201-202	Input cursor log (row, column)
00CB	203	Which key: 64 if no key
00CC	204	0=flash cursor
00CD	205	Cursor timing countdown
00CE	206	Character under cursor
00CF	207	Cursor in blink phase
00D0	208	Input from screen/from keyboard
00D1-00D2	209-210	Pointer to screen line
00D3	211	Position of cursor on above line
00D4	212	0=direct cursor, else programmed
00D5	213	Current screen line length
00D6	214	Row where curosr lives
00D7	215	Last inkey/checksum/buffer
00D8	216	# of INSERTs outstanding
00D9-00F0	217-240	Screen line link table
00F1	241	Dummy screen link
00F2	242	Screen row marker
00F3-00F4	243-244	Screen color pointer
00F5-00F6	245-246	Keyboard pointer
00F7-00F8	247-248	RS-232 Rcv pntr
00F9-00FA	249-250	RS-232 Tx pntr
00FF-010A	256-266	Floating to ASCII work area

[Additional VIC Maps appeared in **COMPUTE!**, *January, 1982, #20, pgs. 181-3. —Ed]*

```
FF8A-FFF5 65418-65525 Jump Table, Including:
  FFC6 - Set Input channel
  FFC9 - Set Output channel
  FFCC - Restore default I/O channels
  FFCF - INPUT
  FFD2 - PRINT
  FFE1 - Test Stop key
  FFE4 - GET
```

c000	ROM control vectors	cb1e	Print message from (y,a)
c00c	Keyword action vectors	cb3b	Print format character
c052	Function vectors	cb4d	Bad-input routines
c080	Operator vectors	cb7b	Perform [GET]
c09e	Keywords	cba5	Perform [INPUT#]
c19e	Error messages	cbbf	Perform [INPUT]
c328	Error message vectors	cbf9	Prompt & input
c365	Miscellaneous messages	cc06	Perform [READ]
c38a	Scan stack for FOR/GOSUB	ccfc	Input error messages
c3b8	Move memory	cd1e	Perform [NEXT]
c3fb	Check stack depth	cd78	Type-match check
c408	Check memory space	cd9e	Evaluate expression
c435	'OUT OF MEMORY'	cea8	Constant - PI
c437	Error routine	cef1	Evaluate within brackets
c469	Break entry	cef7	Check for ')'
c474	'READY.'	ceff	Check for comma
c480	Ready for Basic	cf08	Syntax error
c49c	Handle new line	cf14	Check range
c533	Re-chain lines	cf28	Search for variable
c560	Receive input line	cfa7	Set up FN reference
c579	Crunch tokens	cfe6	Perform [OR]
c613	Find Basic line	cfe9	Perform [AND]
c642	Perform [NEW]	d016	Compare
c65e	Perform [CLR]	d081	Perform [DIM]
c68e	Back up text pointer	d08b	Locate variable
c69c	Perform [LIST]	d113	Check alphabetic
c742	Perform [FOR]	d11d	Create variable
c7ed	Execute statement	d194	Array pointer subroutine
c81d	Perform [RESTORE]	d1a5	Value 32768
c82c	Break	d1b2	Float-fixed conversion
c82f	Perform [STOP]	d1d1	Set up array
c831	Perform [END]	d245	'BAD SUBSCRIPT'
c857	Perform [CONT]	d248	'ILLEGAL QUANTITY'
c871	Perform [RUN]	d34c	Compute array size
c883	Perform [GOSUB]	d37d	Perform [FRE]
c8a0	Perform [GOTO]	d391	Fixed-float conversion
c8d2	Perform [RETURN]	d39e	Perform [POS]
c8f8	Perform [DATA]	d3a6	Check direct
c906	Scan for next statement	d3b3	Perform [DEF]
c928	Perform [IF]	d3e1	Check FN syntax
c93b	Perform [REM]	d3f4	Perform [FN]
c94b	Perform [ON]	d465	Perform [STR$]
c96b	Get fixed point number	d475	Calculate string vector
c9a5	Perform [LET]	d487	Set up string
ca80	Perform [PRINT#]	d4f4	Make room for string
ca86	Perform [CMD]	d526	Garbage collection
caa0	Perform [PRINT]	d5bd	Check salvageability

d606	Collect string
d63d	Concatenate
d67a	Build string to memory
d6a3	Discard unwanted string
d6db	Clean descriptor stack
d6ec	Perform [CHR$]
d700	Perform [LEFT$]
d72c	Perform [RIGHT$]
d737	Perform [MID$]
d761	Pull string parameters
d77c	Perform [LEN]
d782	Exit string-mode
d78b	Perform [ASC]
d79b	Input byte parameter
d7ad	Perform [VAL]
d7eb	Get params for poke/wait
d7f7	Float-fixed
d80d	Perform [PEEK]
d824	Perform [POKE]
d82d	Perform [WAIT]
d849	Add 0.5
d850	Subtract-from
d853	Perform [SUBTRACT]
d86a	Perform [ADD]
d947	Complement fac#1
d97e	'OVERFLOW'
d983	Multiply by zero byte
d9ea	Perform [LOG]
da2b	Perform [MULTIPLY]
da59	Multiply-a-bit
da8c	Memory to FAC#2
dab7	Adjust FAC#1/#2
dad4	Underflow/overflow
dae2	Multiply by 10
daf9	+10 in floating pt
dafe	Divide by 10
db12	Perform [DIVIDE]
dba2	Memory to fac#1
dbc7	FAC#1 to memory
dbfc	FAC#2 to fac#1
dc0c	FAC#1 to FAC#2
dc1b	Round FAC#1
dc2b	Get sign
dc39	Perform [SGN]
dc58	Perform [ABS]
dc5b	Compare FAC#1 to mem
dc9b	Float-fixed
dccc	Perform [INT]
dcf3	String to fac
dd7e	Get ascii digit
dddd	Float to ascii
df16	Decimal constants
df3a	TI constants
df71	Perform [SQR]
df7b	Perform [POWER]
dfb4	Perform [NEGATIVE]
dfed	Perform [EXP]
e040	Series evaluate 1
e056	Series evaluate 2
e094	Perform [RND]
e0f6	?? Breakpoints ??
e127	Perform [SYS]
e153	Perform [SAVE]
e162	Perform [VERIFY]
e165	Perform [LOAD]
e1bb	Perform [OPEN]
e1c4	Perform [CLOSE]
e1d1	Parameters for load/save
e203	Check default parameters
e20b	Check for comma
e216	Parameters for open/close
e261	Perform [COS]
e268	Perform [SIN]
e2b1	Perform [TAN]
e30b	Perform [ATN]
e378	Initialize
e387	CHRGET for zero page
e3a4	Initialize Basic
e429	Power-up message
e44f	Vectors for $300
e45b	Initialize vectors
e467	Warm restart
e476	Program patch area
e4a0	Serial output '1'
e4a9	Serial output '0'
e4b2	Get serial input & clock
e4bc	Program patch area
e500	Set 6522 addrs
e505	Set screen limits
e50a	Track cursor location
e518	Initalize I/O
e54c	Normalize screen
e55f	Clear screen
e581	Home cursor
e587	Set screen pointers
e5bb	Set I/o defaults
e5c3	Set vic chip defaults
e5cf	Input from keyboard
e64f	Input from screen
e6b8	Quote mark test
e6c5	Set up screen print
e6ea	Advance cursor
e715	Retreat cursor
e72d	Back into previous line
e742	Output to screen
e8c3	Go to next line
e8d8	Do 'RETURN'
e8e8	Check line decrement
e8fa	Check line increment
e912	Set colour code
e921	Colour code table
e929	Code conversion

e975	Scroll screen
e9ee	Open space on screen
ea56	Move screen line
ea6e	Synch colour transfer
ea7e	Set start-of-line
ea8d	Clear screen line
eaa1	Print to screen
eaaa	Store on screen
eab2	Synch colour to char
eabf	Interrupt (IRQ)
eb1e	Check keyboard
ec00	Set text mode
ec46	Keyboard vectors
ec5e	Keyboard maps
ed21	Graphics/text control
ed30	Set graphics mode
ed5b	Wrap up screen line
ed6a	Shifted key matrix
eda3	Control key matrix
ede4	Vic chip defaults
edfd	Screen line adds low
ee14	Send 'talk'
ee17	Send 'listen'
ee1c	Send control char
ee49	Send to serial bus
eeb7	Timeout on serial
eec0	Send listen SA
eec5	Clear ATN
eece	Send talk SA
eee4	Send serial deferred
eef6	Send 'untalk'
ef04	Send 'unlisten'
ef19	Receive from serial bus
ef84	Clock line on
ef8d	Clock line off
ef96	Delay 1 ms
efa3	RS232 send (NMI)
efee	New RS232 byte send
f016	Error or quit
f027	Compute bit count
f036	RS232 receive (NMI)
f05b	Setup to receive
f09d	Receive parity error
f0a2	Receive overrun error
f0a5	Receive break error
f0a8	Receive frame error
f0b9	Bad device
f0bc	File to RS232
f0ed	Send to RS232 buffer
f116	Input from RS232 buffer
f14f	Get from RS232 buffer
f160	Check serial bus idle
f174	Messages
f1e2	Print if direct
f1f5	Get..
f205	..from RS232
f20e	Input
f250	Get.. tape/serial/RS232
f27a	Output..
f290	..to tape
f2c7	Set input device
f309	Set output device
f34a	Close
f3cf	Find file
f3df	Set file values
f3ef	Abort all files
f3f3	Restore default I/O
f40a	Do file opening
f495	Send SA
f4c7	Open RS232
f542	Load program
f647	'SEARCHING'
f659	Print file name
f66a	'LOADING/VERIFYING'
f675	Save program
f728	'SAVING'
f734	Bump clock
f760	Get time
f767	Set time
f770	Action stop key
f77e	File Error Messages
f7af	Find any tape header
f7e7	Write tape header
f84d	Get buffer address
f854	Set buffer start, end pointers
f867	Find specific header
f88a	Bump tape pointer
f894	'PRESS PLAY .. '
f8ab	Check cassette status
f8b7	'PRESS RECORD ..'
f8c0	Initiate tape read
f8e3	Initiate tape write
f8f4	Common tape read/write
f94b	Check tape stop
f95d	Set timing
f98e	Read bits (IRQ)
faad	Store characters
fbd2	Reset pointer
fbdb	New tape character setup
fbea	Toggle tape
fc06	Data write
fc0b	Tape write (IRQ)
fc95	Leader write (IRQ)
fccf	Restore vectors
fcf6	Set vector
fd08	Kill motor
fd11	Check read/write pointer
fd1b	Bump read/write pointer
fd22	Powerup entry
fd3f	Check A-rom
fd52	Set kernal2

```
fd8d   Initialize system constants
fdf1   IRQ vectors
fdf9   Initialize I/O regs
fe49   Save data name
fe50   Save file details
fe57   Get status
fe66   Flag ST
fe6f   Set timeout
fe73   Read/set top of memory
fe82   Read/set bottom of memory
fe91   Test memory location
fea9   NMI interrupt entry
fed2   RESET/STOP warm start
fede   NMI RS232 sequences
ff56   Restore & exit
ff5c   RS232 timing table
ff72   Main IRQ entry
ff8a   Jumbo jump table
fffa   Hardware vectors
```

©

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

80/40 COLUMN ALIEN INVADERS
$29.95

Exactly duplicates the famous arcade game. Enjoy hours of challenging fun. For Commodore Computers with Release 4 of BASIC, either 40 or 80 Column Screens

COMMODORE VIC SOFTWARE

Quality Software which incorporates exciting Sound, Color, Graphics and Annimation. Show off all of the many facets and capabilities of your computer.

Lunar Lander	Home Finance
Wrap	Math Drill
Breakout	Hangman
Blackjack	Yahtze
Pong	Othello
Slots	Target Moon
Alien Raiders	Shooting Gallery
Bombs Away	Capture

All of the above are priced at $9.95 or select any three for $27.50. Please add $3.00 and specify the DOS release if you wish disk based software.
Send SASE to be included on our mailing list.
Add .50 shipping and handling. Thank You.

MasterCard and VISA Accepted.

MICRO SPEC LTD
2905 Ports O'Call Ct. Plano, Tx 75075 (214) 867-1333

mis VIC-20 SOFTWARE

EGG ATTACK JOYSTICK

This seemingly innocent electronic game of catch turns gradually into a harried race of joystick control. The player must maneuver a nuclear reactor in order to catch radioactive eggs as they are dropped from advancing "monsters." If the player misses three eggs, the Earth is no longer habitable and their turn of play is over. **12.95**

TUNNEL PATROL JOYSTICK

The player is required in this arcade game to maneuver an android through a maze in order to harvest "Energy Crystals" and score points. As an obstacle, the program randombly throws the android into the fourth dimension where it must battle the dreaded "Batwing" in order to survive. **12.95**

AVAILABLE EARLY APRIL: OUR COMPREHENSIVE CHECKBOOK PROGRAM.

See this and other MIS software at your local computer store, or order direct from MIS (408) 338-9546. California residents add sales tax.

DEALERS INQUIRIES WELCOME

250 FERN ROCK WAY/BOULDER CREEK CA./95006

Great VIC Software

New Programs for your VIC Computer On Cassette!

COLOR & SOUND

• Cattle Roundup	• Adventure
• Artillery Shoot	• Head On
• Micro Maze	• Target
• Snake Out	• Hang In
• Trapper	• Chase

$9.95 ea.

(Includes shipping, Arizona residents add sales tax).

Ready To Load — Hours of Fun!
Write For Free Catalog

ComputerMat ***Software***
Box 1664 Dept. V
Lake Havasu City, AZ 86403

EPROM Reliability

Michael E. Day
West Linn, OR

Although EPROMs are in widespread use, there are continuing problems with the use of the device affecting their overall reliability.

The following report describes how to obtain the maximum performance and reliability from the 2708 EPROM. The concepts involved, however, may be applied to most of the ultra-violet erasable PROMs on the market to date.

The EPROM 'cell' consists, basically, of a capacitor which either has a charge on it or does not. The charge is created by applying a high voltage pulse to the device, and is removed by exposing the device to high intensity ultra-violet light.

The cell is programmed by injection of high energy electrons through the oxide onto the floating gate. Once there, the charge is trapped, as there are no electrical connections to this floating gate. This action is similar to the action of a zener diode in that, as the voltage increases, it finally passes a point where it can overcome the barrier presented by the silicon oxide surrounding the gate and allows the electrons to flow to the gate and collect there. As the voltage is removed it finally drops to a point where it can no longer maintain the bridge through the oxide, and it again becomes isolated. However, the gate now has a charge of electrons on it.

The charge is removed from the cell by exposure with ultra-violet light of the correct wave length (2537A) and energy (10 watt seconds/cm^2) which will impart sufficient photon energy to the trapped electrons to allow the floating gate to be fully discharged.

The presence of charge on the floating gate causes a shift of the cell threshold. In the discharged state (no charge on the floating gate) the cell has a low threshold, and selection of the cell turns on the transistor. Storing a charge on the gate shifts the threshold of the cell above the select voltage so that the transistor will not turn on when it is selected. The amount of charge on the gate determines the level of select voltage at which the transistor will change from a non-conducting to a conducting state. The cell is designed so that the discharged threshold and charged threshold are equally above and below the select voltage. This provides for maximum immunity against marginal cells.

Data retention can be measured by baking the device at an elevated temperature (250°C). 168 hours at this temperature is equivalent to 10 years at 70°C. Test samplings have shown that the time to 5% batch failure is 100 years.

Experiments have been made to determine the effects of prolonged exposure to UV light. Through the first 20 hours the threshold voltage increased slightly after which it stabilized out to 30 days at which time the test was terminated. Although no study has been made to determine what is causing the initial change, it is thought to be caused by some radiation damage caused by the UV.

It is believed that UV lamps with short wavelengths (less than 1800A) and high intensity can ionize oxide with long exposure. The theory is that this will shift the threshold until the part will not function properly. This is not a permanent shift and a bake at 150° for 24 hours should correct the problem.

Some EPROMs exhibit a sensitivity to ambient light. This does not erase them, but they may not function properly. This is a common phenomenon with most semiconductors. Covering the lid with some sort of opaque material will prevent this.

For a given device, given that the programming equipment is operating at factory specifications, the failure to take a charge is device-related, and attempts to bring the charge level higher by reprogramming will seldom be successful. Failure to erase is the most common problem. There are many factors which can cause inadequate erasure; among them are weak UV lamp due to age, dirt on the IC (both internal and external), dirt on the UV lamp, erase requirements outside of normal specifications, or a defective component.

The EPROM is read by determining if the charge on the capacitor of the cell is above or below the threshold of the sensing transistor (the threshold being that level of applied voltage which causes the transistor to change from a non-conducting state to a conducting state). This threshold can be affected by shifts in the -12 volt and -5 volt supplies at the device and temperature. Due to this, if the charge on the cell is near the threshold of the sensing transistor, a shift in the supply voltage or temperature can cause the cell to appear to change state, have an excessive access time, or be intermittent. A cell which is sufficiently near the threshold of the sensing transistor so that it can be affected by temperature or voltage shifts is called "marginally programmed" or "marginal."

One failure of the EPROM is a "leaky cell" (a cell that loses its charge after a short period of time). A leaky cell can be found several ways. One way is to bake the device at 250° after programming it, and then test for lost data.

monitor analog data quickly and inexpensively

ADAM

provides analog data monitoring for your microcomputer. With proper transducers it allows you to sense temperature, wind direction, electrical current, or joysticks and convert the sensors analog voltages into digital readings your microcomputer can process.

SPECS

8 bit resolution A/D
0-5 volt input range
16 Analog Input channels

MODELS

ADAM with TRS-80 interface	$190.00
ADAM with R5-232/RS-423 interface	$250.00
ADAM with RS-422 interface	$275.00

Dealer inquiries welcomed.

small system design

P. O. BOX 4546 • MANCHESTER, NH 03108

Representatives Pacific Northwest Data Systems Marketing (503) 297-8444

New England American Data Systems Marketing (617) 341-0171

T.M.

Oryx software

What does the Oryx, an African antelope, have in common with reliable service? They are both almost extinct.
Oryx Software wants to bring trustworthy service back to life. We're here to stay!

GUARANTEED LOWEST PRICES!
To celebrate our opening, we are matching any advertised price. Just show us the ad.

ORDER TOLL FREE · Outside WI 1-800-826-1589

SOFTWARE

Graphpower	299.00
Tax Preparer '82	127.00
Real Estate Analyzer	127.00
Creative Financing	127.00
Word Processor	425.00
Financial Projection	550.00
Mail Management	250.00
Micropro Wordstar	248.00
Mailmerge	90.00
Spellstar	169.00
Supersort II	159.00
Context Connector	180.00
Easywriter (Pro)	195.00
Easymover	45.00
Tellstar II	70.00
Easymailer	160.00
Datadex	249.00
Microplan Basic	419.00
Executive Secretary	212.00
Nevada CoBol	129.00

ACCESSORIES

Apple II Dual Thermometer w/software	260.00
Mountain Expansion Chassis	625.00
CPS Multifunction	199.00
Mountain Music System	395.00
100,000 Day Clock	375.00
Mountain Clock	252.00
Supertalker SD 2000	199.00
A/D + D/A	299.00
CCS 12K ROM/PROM Board	99.00
CCS A/D Converter Board	99.00
CCS Serial Asynch. Board	139.00
CCS Calendar/Clock Board	99.00
VOTRAX Type-n-Talk	375.00
ADALAB Package	495.00
DOS Dater	49.00
ABT Numeric Keypad	125.00
ABT Softkey	150.00
ABT Barwand	195.00
ABT Cash Box	295.00

. . . AND MANY MORE!

Call for price information:

Printers - Epson, Okidata, Microline, NEC, C. ITOH, Qume, Diablo, Anadex, IDS, MPI, Microtek, Printek.

Monitors - Amdek, Sanyo, Hitachi, NEC, Kaga/TSK.

Modems - Novation, Hayes.

Disk Drives - Percom, Corvus, Morrow, CCS, etc.

Plotters - Watanabe.

Quoted prices valid for stock on hand and subject to change without prior notice.

Ask for our **FREE** catalog.

Please:
- Wisconsin residents - add 4% sales tax
- Foreign orders - please add 15%
- Add $2.50 for shipping per software and small item. Call regarding others.

We Welcome:
- **Visa, MasterCharge** - (Add 4%)
- **Checks** (Allow 1-2 weeks for clearing)
- **COD** (Add 4%)

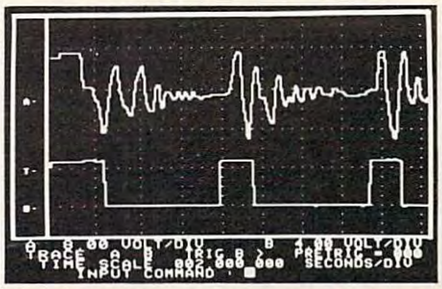
TOLL FREE - Outside Wisconsin: 1-800-826-1589

For Technical Info & in Wisconsin:
715-848-2322

Oryx Software
P.O. Box 1961 • Wausau, WI 54401

APPLESCOPE

DIGITAL STORAGE OSCILLOSCOPE
Interface for the Apple II Computer



```
A   8.00 VOLT/DIV          B   4.00 VOLT/DIV
TRACE  A  B   TRIG B >   PRETRIG = 000
 TIME SCALE  002.000.000   SECONDS/DIV
      INPUT COMMAND :
```

The APPLESCOPE system combines two high speed analog to digital converters and a digital control board with the high resolution graphics capabilities of the Apple II computer to create a digital storage oscillosocpe. Signal trace parameters are entered through the keyboard to operational software provided in PROM on the DI control board.

- DC to 3.5 Mhz sample rate with 1024 byte buffer memory
- Pretrigger Viewing
- Programmable Scale Select
- Continuous and Single Sweep Modes
- Single or Dual Channel Trace

Price for the two board Applescope system is **$595**
EXTERNAL TRIGGER ADDAPTER **$29**

APPLESCOPE ACCESSORIES

APPLESCOPE-HR12 High resolution 12 bit analog to digital converter with sample rates to 100 Khz. Requires 48K Apple II with disk drive. Software provide on floppy disk includes basic SCOPE DRIVER package.
Price per channel **$695**

APPLESCOPE-HRHS High Resolution AND High Speed. Circuit combines two 6 bit flash analog to digital converters to give a 10 bit dynamic range. The 10 bit converter resolution is maintained at sampling rates up to the 7 Mhz. maximum for signal slew rates less than .5 volts per microsecond. Larger inputs slew rates will reduce the converter resolution to 6 bits until the signal stabilizes within the .5 Volt per microsecond limit. Requires 48K Apple II with disk drive. Software provided on disk includes the basic SCOPE DRIVER package. Price per channel **$695**

APPLESCOPE-EXT External trigger adapter has a switch selectable external trigger input to a BNC connector mounted in a rear slot of the Apple II computer.
Price **$29.00**

APPLESCOPE-BNC BNC adapter connects the Berg stick connectors on the A1 circuit card to male BNC plugs mounted in a rear slot of the Apple II compter.
Price **$14.95**

BUS EXTENDERS Allow easy access to Apple II peripheral circuit cards. Price **$19.95**

SCOPE PROBES Oscilloscope probes for use with the APPLESCOPE - BNC adapter. Price**$29.95**

SCOPE DRIVER Advanced software for the Applescope system provided on 5¼" floppy disk. Available options include:
- Signal Averaging - Acquires 1 to 255 signal sweeps and displays the averaged result.
- Digital Volt Meter - Allows use as a real time DVM or use to measure points on an acquired sweep.
- Hard Copy - Uses graphics printer to produce hard-copy output of displayed traces.
- Disk Storage - Allows automatic storage and recover of acquired data on floppy disks.
- Spectrum Analyzer - Calculates and displays frequency spectrum of acquired data.

BUS RIDER
LOGIC ANALYZER for the APPLE II

The BUS RIDER circuit card silently rides the Apple II peripheral bus and allows real time tracking of program flow. Software provided on EPROM allows set up of trace parameters from the keyboard and read back of disassembled code after a program has been tracked.

- 32 bit by 1024 sample memory buffer
- Monitors Data and Address bus plus 8 external inputs
- Trigger on any 32 bit word or external trigger
- Pretrigger viewing

The BUS RIDER is an invaluable development tool for anyone working with Apple II or Apple II+ computers.
Price **$295**

RC ELECTRONICS INC.
7265 Tuolumne St., Goleta, CA 93117
Visa **(805) 968-6614** M/C

Another method of testing for leaky cells is to make an erase profile for the suspect EPROM. This is done by programming the device, and then erasing it in one to two minute increments, measuring the number of erased bits after each increment. Making a graph with this information will give you a profile of the erasure characteristics of the EPROM. Any cell that erases twice as fast as the overall average should be cosidered suspect.

Another failure mode of the EPROM is the "sticky cell" (a cell which is difficult to program or erase). Although a sticky cell can be overcome by a longer program or erase time, in a production environment it is not acceptable to adjust these times for each device. Therefore, any device which requires more than three times the normal time to program or erase should be considered defective.

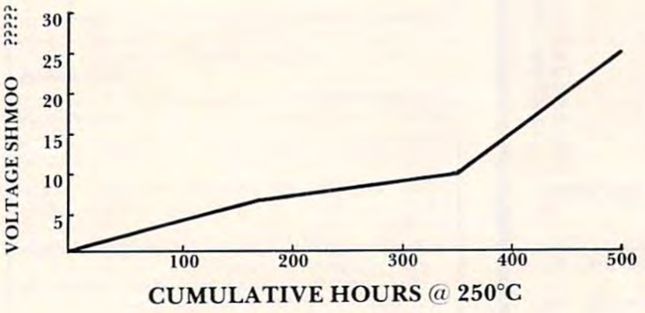
The major source of problems with the 2708 EPROM is inadequate erasure. In testing the EPROM to determine if it has been adequately programmed or erased, it is not acceptable to simply read the PROM and compare the information read against the true data, since marginal cells may not be found with this method. A more reliable method of verifying if an EPROM has been properly programmed or erased is to measure the depth of the charge at each cell. This can be done by shifting the threshold level of the sensing transistor above and below the normal level and by doing a normal read and compare.

In this way, a map of the charge level of the cells in the EPROM can be generated by observing the level at which the output changes state.

The threshold level of the 2708 EPROM can be shifted by adjusting the -5 volt supply (VBB). Causing the -5 volts to go more negative will determine how deep the cell has been charged; bringing it more positive will determine how much it has been erased.

The charge limits will vary greatly not only from manufacturer to manufacturer, but from device to device. Therefore, an acceptable limit must be determined at which the device may be considered good or bad. For the 2708 this is greater than twice the tolerance for the -5 volt supply. This can be simply generated by using the forward voltage drop across the diode (.7 volts) above and below the -5 volt level. In more critical applications a two-diode level drop (1.4 volts) might be considered.

More is not always better. Just because the charge on one device is deeper than on another does not mean that it will retain the charge longer. Data retention is related to cell isolation and not necessarily to the level of the charge.



TEMPERATURE	FAILURE RATE 60% CONFIDENCE (% / 1000 hours)	FAILURE RATE 90% CONFIDENCE (% / 1000 hours)
70°C	0.013	0.027
55°C	0.006	0.013

Operating Life Test Results

TEMPERATURE	SAMPLE SIZE	HOURS	EQUIVALENT DEVICE HOURS @ 70°C	FAILURES	FAILURE MODE
160°C	64	2243	39.9×10^6	1	Charge Loss
160°C	49	2028	27.6×10^6	0	
160°C	51	2028	28.7×10^6	1	Charge Loss
160°C	40	2830	31.4×10^6	2	Charge Loss
160°C	80	1176	26.1×10^6	1	Charge Loss
160°C	77	1176	25.1×10^6	4	Charge Loss
160°C	79	984	21.6×10^6	1	Charge Loss

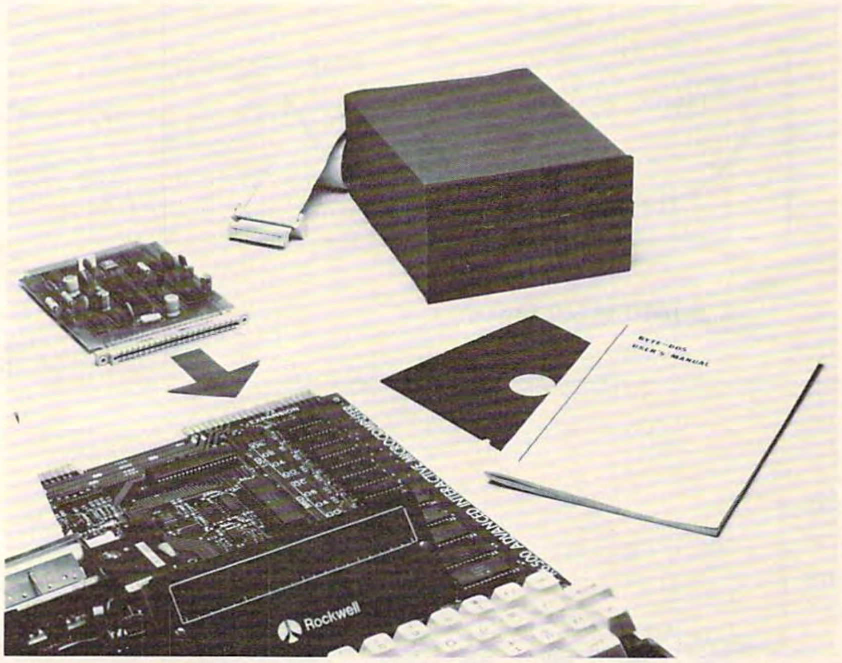
DOS FOR AIM-65®: $499.00 COMPLETE!

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

BYTE–DOS COMMANDS

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

AND MORE!



IT'S EASY AND FUN TO USE!

BYTE MICROSYSTEMS
C O R P O R A T I O N
1477 ELKA AVENUE, SAN JOSE, CA 95129 408/446/0559

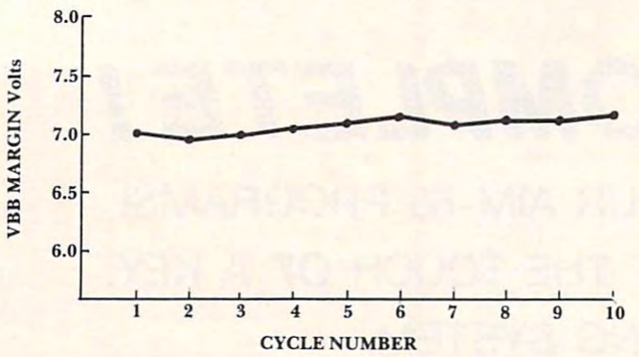
master charge THE INTERBANK CARD
VISA

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

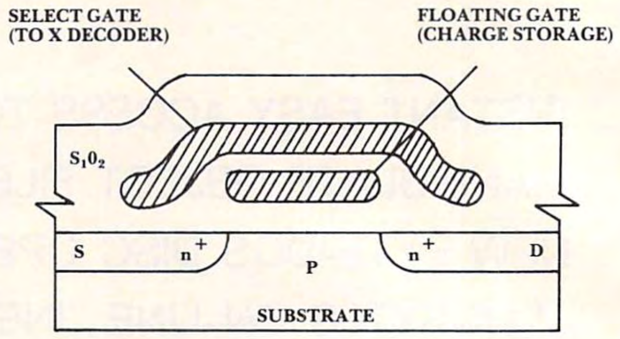
SEND ME THE FULL STORY!

Name ____________________
Company ____________________
Address ____________________
City __________ State ______ Zip ______
Telephone ______________ ☐ home ☐ work

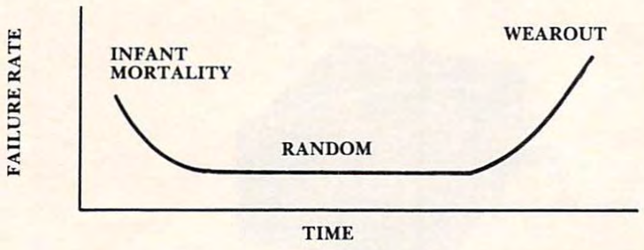
☐ I own an AIM-65.
☐ I'm thinking of buying an AIM-65.
☐ Send full BYTE-DOS Data.



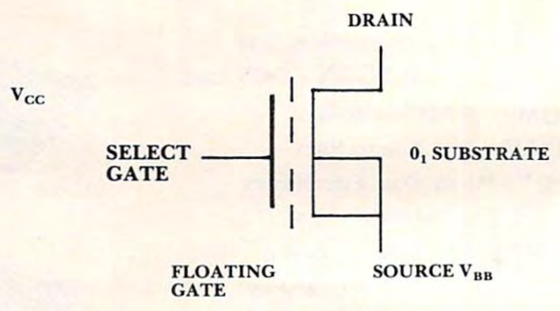
Multiple Program Erase Experiment



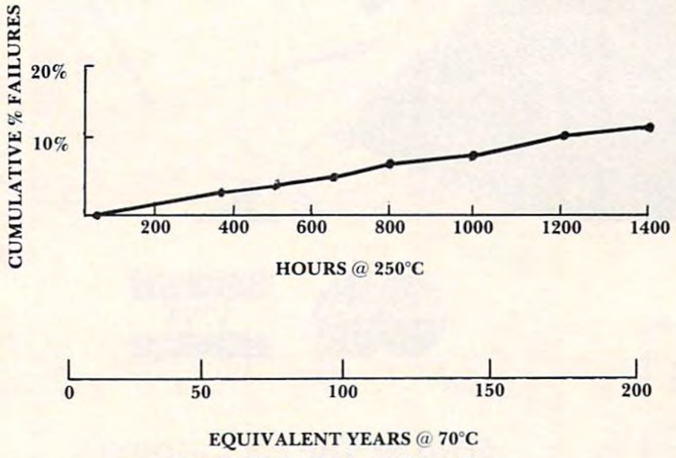
(a) CROSS SECTION



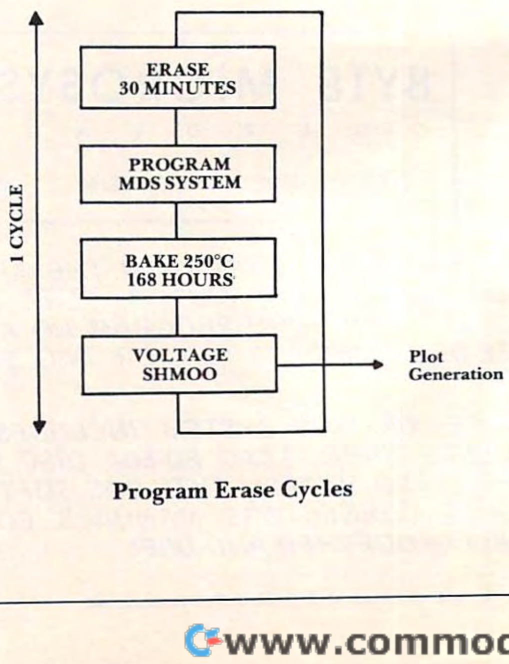
Reliability Life Curve



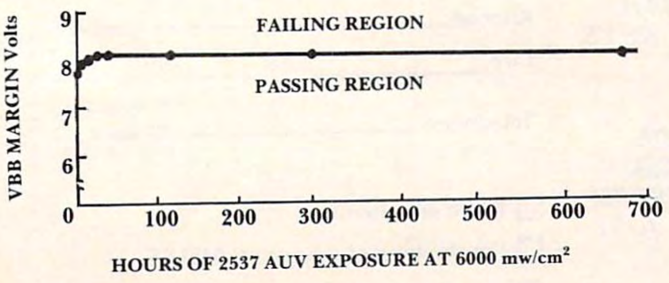
(b) SCHEMATIC SYMBOL



Intel® 250°C Bake Failure Rate



Program Erase Cycles



©

SPRING SPECIAL

No need to wait any longer for our controller development system;

The MMC/03S is back! For $250.00 you get:

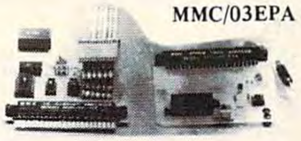
1 MMC/03D Microcomputer, IKRAM, 2-6522's ZIF sockets for CPU and EPROM, 20MA full duplex current loop, & crystal clock.

1 MMC/03ICE In circuit emulator, works off any 6502-based system including AIM, SYM, KIM, OSI. (Apple version slightly more, PET version a lot less)

1 MMC/03EPA EPROM programming adaptor; Programs - 2758's, 2716's, and 2532's.

MMC/03D MMC/03ICE

MMC/03EPA



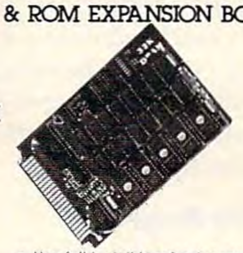
R. J. BRACHMAN ASSOCIATES, INC.
P.O. Box 1077
Havertown, PA 19083
(215) 622-5495

SEE COMPUTE!
April 1981
For Eric Rehnke's Review.

NEED MORE MEMORY?

32K BYTE DYNAMIC RAM & ROM EXPANSION BOARD

Expand Your 4K/8K PET
SYM/KIM/AIM –65 to 32K



- Easily connected to your computer via the expansion connector.
- Build huge and complex programs!
- Need 64K of RAM? Buy two boards, on board configuration circuitry will allow you to expand to 64K easily!
- New dynamic RAM technology brings you more memory in less space and at a lower cost!
- RAM chips are upgrade, compatible with the new 64K RAM chips for future expansion!
- Operates on +5 volts only, supplied from your computer power supply, no on board generators to go bad.
- Requires A LOT less power than static RAM!
- Has full invisible refresh operation, does not interfere with processor operation.
- 5 on board sockets for 2516/2532 (2K/4K) type EPROMS, addressable anywhere.
- Great for designing a two board computer system. (CPU, I/O-RAM, ROM).
- Other specifications:
  Disable any 4K block of RAM for I/O, place RAM above or below 8000 HEX, KIM-4 BUSS COMPATIBLE FOR CARD RACKS. Adapter cables available for non rack use.
- All these features on a 6 x 4.5" board!

**ASSEMBLED & TESTED BOARDS—GUARANTEED FOR 6 MONTHS
PURCHASE PRICE IS FULLY REFUNDABLE IF RETURNED
UNDAMAGED WITHIN 14 DAYS**

Price — **$289.88**

Include $3.00 for S&H within the U.S.
Allow 6 weeks for delivery.
Full informative documentation included with all our products.
50% deposit required on all C.O.D. Orders.
(213) 362-8156 Mail Order Only

PROTRONICS
COMPUTER INNOVATIONS

**17537 Chatsworth St.
Granada Hills, CA. 91344**

ANNOUNCING TWO NEW TERMINALS

Smart • Fast • Graphics • Matching Modem and $295 Printer

Netronics announces a state of the art breakthrough in terminals, now at prices you can afford, you can go on-line with data-bank and computer phone-line services. It's all yours: "electronic newspapers," educational services, Dow-Jones stock reports, games, recipes, personal computing with any level language, program exchanges, electronic bulletin boards . . . and more every day!!!

Netronics offers two new terminals, both feature a full 56 key/128 character typewriter-style keyboard, baud rates to 19.2 kilobaud, a rugged steel cabinet and power supply. The simplest one, FASTERM-64, is a 16 line by 64 or 32 character per line unit, with a serial printer port for making hard copy of all incoming data, and optional provisions for block and special character graphics. The "smart" version, SMARTERM-80, features either 24 line by 80 characters per line or 16 by 40 characters per line, it offers on-screen editing with page-at-a-time printing, 12,000 pixel graphics, line graphics, absolute cursor addressing, underlining, reverse video, one-half intensity and much more . . . simply plug them into your computer or our phone modem and be on-line instantly. Use your TV set (RF modulator required) or our delux green-phosphor monitor pictured above. For hard copy just add our matched printer.

Price breakthrough!!! Own the FASTERM-64, a complete terminal kit, ready to plug in for just $199.95 or order the SMARTERM-80 kit for just $299.95, (both available wired and tested.) Be on-line with the million-dollar computers and data services today . . . we even supply the necessary subscription forms.

More good news: All the components in our terminals are available separately (see coupon), so you buy only what you need!!!

FASTERM-64 . . . DISPLAY FORMAT: 64 or 32 characters/line by 16 lines . . . 96 displayable ASCII characters (upper & lower case) . . . 8 baud rates: 150, 300, 600, 1200, 2400 4800, 9600, 19, 200, (switch sel.) . . . LINE OUTPUT: RS232/C or 20 ma current loop . . . VIDEO OUTPUT: 1V P/P (EIA RS-170) . . . CURSOR MODES: home & clear screen, erase to end of line, erase cursor line, cursor up & down, auto carriage return/line feed at end of line & auto scrolling . . . REVERSE VIDEO . . . BLINKING CURSOR . . . PARITY: off, even or odd . . . STOP BITS: 1, 1.5, 2 . . . DATA BITS PER CHARACTER: 5, 6, 7 or 8 . . . CHARACTER OUTPUT: 5 by 7 dot matrix in a 7 by 12 cell . . . PRINTER OUTPUT: prints all incoming data . . . 1K ON BOARD RAM . . . 2K ON BOARD ROM . . . CRYSTAL CONTROLLED . . . COMPLETE WITH POWER SUPPLY . . . OPTIONAL GRAPHICS MODE: includes 34 Greek & math characters plus 30 special graphics characters . . . ASCII ENCODED KEYBOARD: 56 key/128 characters.
SMARTERM-80 . . . DISPLAY FORMAT: 80 characters by 24 lines or 40 characters by 16 lines 128 displayable ASCII characters (upper & lower case) 8 baud rates: 110, 300, 600, 1200, 2400, 4800, 9600, 19, 200 . . . LINE OUTPUT: RS232/C or 20 ma current loop . . . VIDEO OUTPUT: 1V pp (EIA RS-170) . . . EDITING FEATURES: insert/delete line, insert/delete character, forward/back tab . . . LINE OR PAGE TRANSMIT . . . PAGE PRINT FUNCTION . . . CURSOR POSITIONING: up, down, right, left, plus absolute cursor positioning with read back . . . VISUAL ATTRIBUTES: underline, blink, reverse video, half intensity, & blank . . . GRAPHICS: 12,000 pixel resolution block plus line graphics . . . ON-SCREEN PARITY INDICATOR . . . PARITY: off, even or odd . . . STOP BITS: 110 baud 2, all others 1 . . . CHAR. OUTPUT: 7 by 11 character in a 9 by 12 block . . . PRINTER OUTPUT . . . 60 OR 50 Hz VERTICAL REFRESH . . . BLINKING BLOCK CURSOR . . . CRYSTAL CONTROLLED . . . 2K ON BOARD RAM . . . ASCII ENCODED KEYBOARD: 56 key/128 character . . . 4K ON BOARD ROM . . . COMPLETE WITH POWER SUPPLY.
TELEPHONE MODEM 103 O/A . . . FULL DUPLEX, FCC APPROVED . . . DATA RATE: 300 baud . . . INTERFACE: RS232/C and TTY . . . CONTROLS: talk/data switch (no need to connect and disconnect phone), originate/answer switch on rear panel . . . NO POWER SUPPLY REQUIRED.
ASCII KEYBOARD ASCII-3 . . . 56 KEY/128 CHARACTER ASCII ENCODED . . . UPPER & LOWER CASE . . . FULLY DEBOUNCED . . . 2 KEY ROLLOVER . . . POS OR NEG LOGIC WITH POS STROBE . . . REQUIRES +5 & -12V DC (SUPPLIED FROM VIDEO BOARDS)
PRINTER COMET I . . . SERIAL I/O TO 9600 BAUD . . . 80 CHARACTER COLUMN (132 COMPRESSED) . . . 10" TRACTOR FEED . . . UPPER/LOWER CASE . . . INDUSTRY STANDARD RIBBONS . . . 4 CHARACTER SIZES . . . 9 BY 7 DOT MATRIX . . . BI-DIRECTIONAL PRINTING

Continental U.S.A. Credit Card Buyers Outside Connecticut
CALL TOLL FREE 800-243-7428

To Order From Connecticut Or For Tech. Assist. Call (203) 354-9375

NETRONICS R&D LTD. Dept.
333 Litchfield Road, New Milford, CT 06776

Please send the items checked below:

☐ COMPLETE FASTERM-64 TERMINAL (includes FASTVID-64 video board ASCII-3 keyboard, steel cabinet and power supply) . . . kit $199.95 plus $3 P&I . . . wired & tested $249.95 plus $3 P&I . . . graphics option: add $19.95 to each of above
☐ COMPLETE SMARTERM-80 TERMINAL (includes SMARTVID-80 video board, ASCII-3 keyboard, steel cabinet and power supply) . . . kit $299.95 plus $3 P&I . . . wired and tested $369.95 plus $3 P&I
☐ FASTVID-64 VIDEO BOARD (requires +5 & -12V DC) . . . kit $99.95 plus $3 P&I . . . graphics option add $19.95 . . . wired & tested $129.95 plus $3 P&I . . . graphics option add $19.95
☐ SMARTVID-80 VIDEO BOARD (requires +5 & +/-12V DC) . . . kit $199.95 plus $3 P&I . . . wired & tested $249.95 plus $3 P&I
☐ DELUXE STEEL TERMINAL CABINET . . . $19.95 plus $3 P&I
☐ ASCII-3 KEYBOARD (requires +5 & -12VDC) . . . kit $69.95 plus $3 P&I . . . wired and tested $89.95 plus $3 P&I
☐ POWER SUPPLY (powers ASCII-3 keyboard & video boards) . . . kit only $19.95 plus $2 P&I
☐ ZENITH VIDEO MONITOR (high resolution green phosphor) . . . wired & tested $149.95 plus $6 P&I
☐ TELEPHONE MODEM MODEL 103 O/A . . . wired & tested $189.95 plus $3 P&I
☐ DOT MATRIX PRINTER Comet I . . . wired & tested $299.95 plus $10 P&I
☐ RF MODULATOR MOD RF-1 . . . kit only $8.95 plus $1 P&I
☐ 3FT-25 LEAD MODEM/TERMINAL OR PRINTER/TERMINAL CONNECTOR CABLE . . . $14.95 ea plus $2 P&I

For Canadian orders, double the postage . Conn. res. add sales tax.

Total Enclosed $ ______________________
☐ Personal Check ☐ Cashier's Check/Money Order
☐ VISA ☐ MasterCard (Bank No. ____________)
Acct. No. ____________ Exp. Date ____________

Signature ______________________
Print Name ______________________
Address ______________________
City ________ State ________ Zip ________

Random Music Composition On The PET

Alfred J. Bruey
Jackson, MI

This program, MUSICOMP, lets the PET computer compose and play music. MUSICOMP was written to provide the user with an introduction to computer generated music. The music is output using the CB2 method of music generation which is described many places in PET literature. Attachments A and B give descriptions of the hardware that you can use if you don't already have CB2 sound. Figure 1 shows the connections necessary to output sounds from the PET to an audio amplifier. Figure 2 shows a simple audio amplifier that you can make if you don't have one.

Program Description

MUSICOMP generates three kinds of music: white music, brown music, and 1/f music. For a complete description of these three types of music, see Martin Gardener's Mathematical Games column in the April, 1978, issue of *Scientific American* magazine.

A. White music: White music is a sequence of completely random sounds. In this program, you have your choice of two different types of white music:

1. Option 1 on the menu allows any of 256 different frequencies to be generated. The notes are not correlated with each other in any way. It is unlikely that you will want to go away humming the tunes you generate using this option.

2. Option 2 also generates random sounds, but these sounds are restricted to: the 25 piano notes (well-tempered scale) beginning with the B below middle C.

B. Brown Music: The second type of music is called brown music (Option 3). It is similar to the Brownian motion of particles. In brown music, each note can vary by only one tone (half-step) from the preceding note. The only randomness is in choosing the starting note and in determining whether each note is one tone higher or lower than its predecessor. You will probably find this music boring. It sounds something like a finger exercise for a violinist.

To get brown music, enter a 1 when you are asked for the maximum variation. Entering some other number, a 3 for example, will allow each note to vary three tones from its predecessor. True brown music allows only a one tone variation from note to note. The option of choosing a maximum variation is given so you can experiment with sounds.

C. 1/f Music: The final type of random music in this program is 1/f music. This music is somewhere between the randomness of white music and the boring regularity of brown music. 1/f music was discovered by an investigator who was trying to find music in nature. The algorithm used in this program is the same as the one described in the previously mentioned article except five different colored dice were used instead of three so that tunes 32 notes long could be created. Most listeners agree that 1/f music is much more musical than either white or brown music.

Extensions

I assume that anyone who knows BASIC and a minimum of music will want to change this program. That's why an annotated listing of the program is provided.

You might want to add options which impose different rules on the composition. You might also want to add the coding to save the composition on tape or disk. The place where you might do this is marked in the listing.

Using The Program

Load the program in the usual way. The main menu will be displayed on the screen as follows. Press the proper key from 1 to 5 to make your selection, but do not press RETURN. (If you press RETURN accidently and get the READY signal, type CONT and press RETURN and you'll be right back where you left off.)

```
COMPOSITION SELECTION
1 RANDOM TUNE
2 RANDOM TUNE, WELL-TEMPERED
3 RANDOM TUNE, WELL-TEMPERED
  WITH STEP SIZE LIMIT
4 1/F MUSIC
5 END PROGRAM
```

A brief description of each of the options follows:

Option 1: Random notes – This option will compose and play tunes based on 256 different tones, ranging from a tone slightly below the B below middle C to a tone that's probably even too high for your dog

The SM-KIT is a collection of machine language firmware programming and test aids for BASIC programmers. SM-KIT is a 4K ROM (twice the normal capacity) which you simply insert in a single ROM socket on any BASIC 4 CBM/PET–either 80 column or 40 column. Includes both programming aids and disk handling commands.

ERROR DETECTION: the SM-KIT automatically indicates the erroneous line and statement for any BASIC program error.
LINE NUMBERING: the SM-KIT automatically numbers BASIC statements until you turn the function off.
SCREEN OUTPUT: the commands FIND, DUMP, TRACE and DIRECTORY display on the CRT while you hold the RETURN key (display pauses when the key is released). Continuous output is selected with shift-lock.
OUTPUT CONTROL to DISK or PRINTER: in addition to displaying on the CRT, you can direct output to either disk or printer.
HARDCOPY: allows screen displays to be either printed or stored on disk.
FIND: searches all or any part of a program for text or command strings or variable names. Either exact search or wild card search supported.
RENUMBER: the SM-KIT can renumber all or any part of a program. The selective renumbering allows you to move blocks of code within your program.
VARIABLE DUMP: displays the contents of floating point, integer, and string variables (both simple and array). Can display all variables or any selected variables.
TRACE: SM-KIT can trace program execution either continuously or step by step starting with any line number. Selected program variables can be displayed while tracing.
DISK COMMANDS: as in DOS Support (Universal Wedge), the "shorthand" versions of disk commands may be used for displaying disk directory, initializing, copying, scratching files, load and run, etc.
LOAD: SM-KIT can load all or part of BASIC or machine language programs. It can append to a program in memory, overwrite any part of a program, load starting with any absolute memory location, and load without changing variable pointers.
MERGE: allows merging all or any part of a program on disk with a program in memory.
SAVE and VERIFY: SM-KIT provides one step program save and verification. It also allows you to save any part of a program, or any address range.

SM-KIT
for Commodore Computers

A Programming Productivity Tool

ONLY $40

A 4K ROM with both programming and disk handling aids.

Developed by (and available in Europe from) SM Softwareverbund-Microcomputer GmbH, Scherbaumstrasse 29, 8000 Munchen 83, Germany

252 Bethlehem Pike
Colmar, PA 18915

215-822-7727 **A B Computers**

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

NEW PRODUCT

SWARM-100

JUST PLUG IT IN
• No soldering • No messy wires

SOFTWARE SELECTABLE

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

$150.00

Master Charge and VISA accepted.

For 24 Pin ROM Machines Only.

BATTERIES INCLUDED

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

to hear. When you press the 1 key, a series of questions will be displayed (Press RETURN after each answer):

HOW LONG IS THE TUNE
(Answer with a number from 1 to 150)

DIFFERENT LENGTH NOTES (Y OR N)
(If you enter a Y, each note length will be one second long. If you answer M it will be ½ second long. If you answer F it will be ¼ second long. All other note lengths will be scaled accordingly.)

REPEAT NOTES (Y OR N)
(If you reply N, the tune will play one time and then the main menu will reappear. If you reply Y, the tune will repeat. In either case, you can stop the tune while it is playing by holding down the X key. You will return to the main menu.)

After you have answered these four questions, there will be a short pause while time values are being calculated for all the notes. Then the tune will begin to play.

Option 2: Random notes, well-tempered. This is the same as Option 1 except that all notes are chosen randomly from one of 25 tones. These tones are the 25 piano notes beginning with the B below middle C.

Option 3: Random notes, well-tempered, with step-size limit. You will be asked the same questions as in Options 1 and 2. After you answer them, you will receive an additional question:

MAX. VAR. FROM LAST NOTE

This question is asking you for the maximum variation in tone (half-steps) that are permissible from one note to the next. If you reply 1, you will get brown music. You may enter any other value just to see what kind of tune the PET will compose.

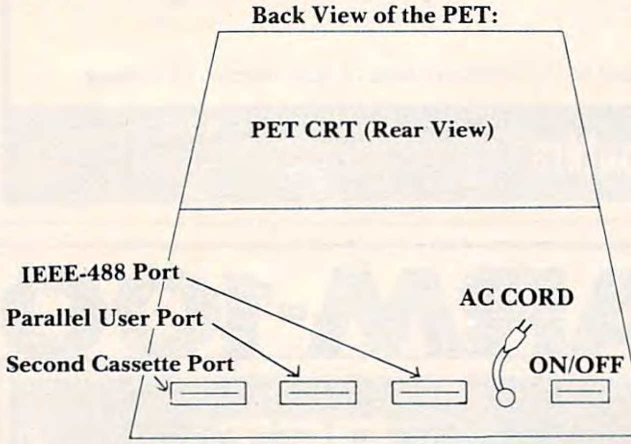
Option 4: 1/f Music. Pressing the 4 key will generate 1/f music. The 1/f tunes will all be 32 notes long, so you will not be asked for the length of the tune. Otherwise, you will be asked the same questions as in Options 1 and 2.

Option 5: End Program. Select Option 5 when you are ready to quit.

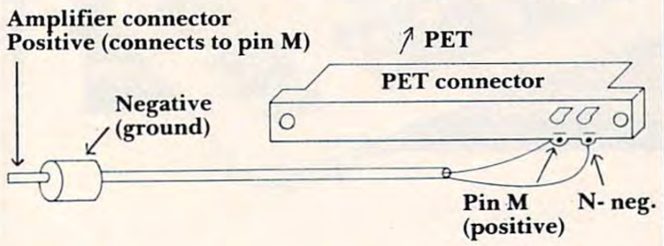
A Circuit For A PET Amplifier

Below is a circuit for a PET amplifier for making music or adding sound effects to your games. Use an RCA phono jack as the input and you'll be able to use the same connector cable as described previously.

Figure 1.

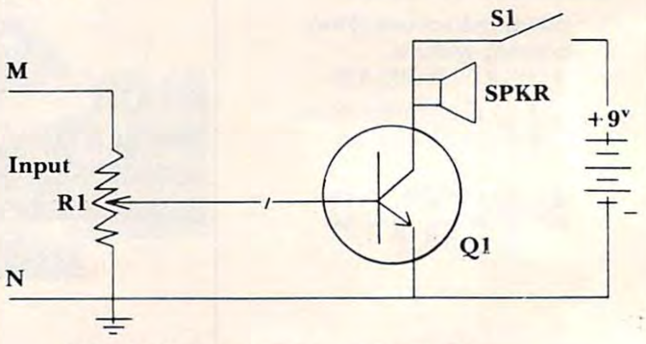


The edge connector that you need plugs into the Parallel user port of the PET. Do not attach it to the IEEE-488 port. (It's not a bad idea to put a strip of masking tape across the IEEE port so you don't accidently plug into it.) Here's what the completed cable should look like. The amplifier end might look different if your system doesn't use the RCA type jack.



You should use shielded cable for the line between the PET and the amplifier. *Be sure you don't put the PET connector on upside down!*

Figure 2.



R1 – 100K potentiometer with switch
S1 – Part of R1
Q1 – RS2031 (Radio Shack 276-2031)
SPKR – 8 ohm SPEAKER
SMALL PLASTIC BOX, 9 VOLT BATTERY, BATTERYHOLDER

• *THE MAGIC OF* MAGIS •

MAGIS is a fully integrated accounting system designed for both experienced and novice business users. No previous knowledge of computers is required. MAGIS offers a comprehensive solution to your business accounting needs. It covers General Ledger, Accounts Receivable, Accounts Payable, Inventory and Payroll. In addition, MAGIS has a Point of Sale or Sales Journal which updates all affected files, and fills in prices and descriptions.

Although a thorough manual is provided, you can operate the system without it, as MAGIS is completely **MENU DRIVEN.** That is, all of your choices are shown on the screen. You never have to look up a command. Default file sizes are:

375	General Ledger Accounts
500	Accounts Receivable (6000 open invoices)
50	Accounts Payable (1000 open invoices)
50	Payroll Records
3000	Inventory Items. (12000 w/2 drives)
10	Departments
40	Depreciable Assets. Automatically computed

How much will all this accounting power cost me? The answer is "Not NEARLY what you would expect to pay!" MAGIS sells for up to $1500.00 at other dealers, but now you can get it for **$499** from **C & O Micro Systems.** MAGIS requires a CBM 8032, 8050 disk drive and printer.

MAGIS IS COMMODORE APPROVED

Also - **ASTEROID MINER.** Uses graphics, music and sound effects. Requires 8032. $14.95 Cass. $16.95 Disk.

C & O Micro Systems Inc.
P.O. Box 14633 Lenexa, KS 66215

To order call toll free 800/854-4368
In California call 800/472-4378
For information call 913/888-4062 or 816/252-0216

80 × 25

PET/CBM™

2000/3000/4000 Series

not using a CRT, or display controller chip

$275.00*

Select either **80 × 25** or **40 × 25** On The Built-in Display

From the keyboard or program

Displays the full, original character set

Available from your local dealer or:
EXECOM CORP.
1901 Polaris Ave.
Racine, WI 53404
Ph. 414-632-1004

*Plus installation charge of $75.00

Available only for Basic 3.0 & Basic 4.0

PET & CBM™a trademark of Commodore Business Machines

NEW PET/CBM SOFTWARE

Let Computer Mat turn your Pet into a Home Arcade!

ASTEROIDZ — Its your ship vs. a swarm of killer gammaroidz. You are on a collision course and must destroy them before they blast you into the next galaxy. Four levels of play. Has hyperspace keys that move you around. Arcade style entertainment at its finest. Great graphics and sound.

Cass. 8K $9.95

MUNCHMAN — How many dots can you cover? It's you against the computer munchers ZIP and ZAP. Can you clear the maze first or will they get you? Number keys move you up, down, right and left. GREAT GRAPHICS AND SOUND.

Cass. 8K $9.95

TARGET COMMAND — Its you against a barrage of enemy lazers that are aimed at your ammo dumps. Sight in on the targets and score as many hits as you dare. As your skill increases so does the the difficulty — (5 levels to select). This is an arcade-style game with great graphics and sound effects. A must for your PET/CBM.

Cass. 8K $9.95

ALL OUR SOFTWARE RUNS IN 8K
OLD-NEW ROM — 40 CHR. SCREEN
WRITE FOR FREE CATALOG OF VIC/PET SOFTWARE
PLEASE ADD $1.00 PER ORDER FOR SHIPPING
COMPUTER MAT • BOX 1664C • LAKE HAVASU CITY, AZ. 86403

OLD PET OWNERS NEVER DIE

☆ ....THEY JUST CALL ON US! ☆

Our innovative line of hardware products protect the investment in your original version 4K/8K PET by extending its life and usefulness. Our software works on most PET/CBMs.

● 2114-TO-6550 RAM ADAPTER HUNDREDS SOLD!

Replaces up to eight 6550 RAMs with low cost, reliable 2114s—one at a time. Use two units for total 6550 replacement. Never worry about RAM availability again!

PHB-001	Bare Board	$11.95
PHK-001S	Full Kit (No 2114s)	$17.95
PH-001S	Assembled (No 2114s)	$24.95

($2.50 postage/handling per order)

● 4K MEMORY EXPANSION

Installs easily internally. Uses eight 2114Ls. Write protect provisions. Second expansion needs external +5V power. "Soft-ROM" applications. Bigger programs!

PHB-002	Bare Board	$16.00
PHK-002	Full Kit (No 2114s)	$29.00
PH-002S	Assembled (No 2114s)	$42.00
PH-002	Assembled with 2114s	$89.00

($3.00 postage/handling per order)

● "REAL WORLD" SOFTWARE

For most PET/CBMs. Word Processor, Mailing List, Catalog, Satellite Tracker, Morse Code Keyer, Frequency Counter. Professional results at low cost!

$17–$25

Write for Free Catalog

PROMPT SHIPMENT! SATISFACTION GUARANTEED
California Residents, add 6% Tax Foreign Shipping Higher

OPTIMIZED DATA SYSTEMS

Dept. C, P.O. Box 595 Placentia, California 92670

DEALER INQUIRIES INVITED

PET/CBM Trademark of Commodore Business Machines

Substitutions may be made for Q1, but a high-gain transistor should be used to be sure of sufficient volume. The transistor can be mounted directly on the leads of the potentiometer.

Program 1.

```
150 DIM SN(15Ø),ST(15Ø),PI(25),PN(1
    5Ø)
16Ø FORI=1TO25:READPI(I):NEXTI
17Ø DATA251,237,223,211,199,188,177
    ,167,157,148,14Ø,132,125,1
    17,111,1Ø4
18Ø DATA98,93,87,82,78,73,69,65,61
19Ø REM ****************************
    ****
2ØØ REM VARIABLE LIST:    *
21Ø REM T=TIME OF NOTE IN 6ØTHS OF
    SECOND
22Ø REM P=POKE NUMBER FOR NOTE
23Ø REM TY$=TYPE OF SONG
24Ø REM    1=RANDOM
25Ø REM    2=RANDOM, WELL-TEMPERED
26Ø REM    3=RANDOM, WELL-TEMPERED,
     LIMIT ON STEP SIZE
27Ø REM    4=1/F MUSIC
28Ø REM    5=STOP
29Ø REM L%=LENGTH OF SONG
3ØØ REM L$="Y" NOTES DIFFERENT LENG
    TH
31Ø REM    "N" NOTES SAME LENGTH
32Ø REM S$="S" SLOW SONG,S=1
33Ø REM    "M" MEDIUM SPEED SONG,S=
    2
34Ø REM    "F" FAST SONG,S=4
35Ø REM ***************************
    ***
36Ø PRINT"{CLEAR}{Ø3 RIGHT}{Ø3 DOWN
    DOWN}{REV}COMPOSITION SELE
    CTION"
37Ø PRINT"{DOWN}{Ø4 RIGHT}{REV}1{OF
    OFF} RANDOM TUNE
38Ø PRINT"{DOWN}{Ø4 RIGHT}{REV}2{OF
    OFF} RANDOM TUNE, WELL-TEM
    PERED"
39Ø PRINT"{DOWN}{Ø4 RIGHT}{REV}3{OF
    OFF} RANDOM TUNE, WELL-TEM
    PERED
4ØØ PRINT"{DOWN}{Ø4 RIGHT}    WITH
    STEP SIZE LIMIT"
41Ø PRINT"{DOWN}{Ø4 RIGHT}{REV}4{OF
    OFF} 1/F MUSIC
42Ø PRINT"{DOWN}{Ø4 RIGHT}{REV}5{OF
    OFF} END PROGRAM
43Ø GET TY$:IFTY$=""THEN43Ø
44Ø ONVAL(TY$)GOTO5ØØ,59Ø,69Ø,98Ø,4
    6Ø
45Ø GOTO43Ø
46Ø REM ****************************
    ***
47Ø REM EXIT ROUTINE **************
    ***
48Ø REM ****************************
    ***
49Ø PRINT"{CLEAR}{Ø3 RIGHT}{Ø4 DOWN
    DOWN}{REV}ROUTINE ENDED":E
    ND
5ØØ REM ********************
51Ø REM PLAY RANDOM ********
52Ø REM ********************
53Ø GOSUB 119Ø :REM GET SONG DATA
54Ø FORI=1TOL%
55Ø SN(I)=INT(RND(3)*255+1)
56Ø NEXTI
57Ø GOSUB141Ø:REM GENERATE NOTES AN
    D PLAY
58Ø GOTO36Ø
59Ø REM****************************
    ***
6ØØ REM RANDOM, WELL-TEMPERED *****
    ***
61Ø REM ***************************
    ***
62Ø GOSUB 119Ø :REM GET SONG DATA
63Ø FORI=1TOL%
64Ø SN(I)=INT(RND(5)*25+1)
65Ø SN(I)=PI(SN(I))
66Ø NEXTI
67Ø GOSUB 141Ø
68Ø GOTO36Ø
69Ø REM ***************************
    ***
7ØØ REM RANDOM,WELL-TEMP,STEP-SIZE
    ****
71Ø REM ***************************
    ***
72Ø GOSUB 119Ø :REM GET SONG DATA
73Ø SN(1)=INT(RND(6)*25+1):PN(1)=PI
    (SN(1))
74Ø IFMV>1THEN85Ø
75Ø REM BROWNIAN MOVEMENT
76Ø FORI=2TOL%
77Ø IFSN(I-1)=1THENSN(I)=2:PN(I)=PI
    (2):GOTO83Ø
78Ø IFSN(I-1)=25THENSN(I)=24:PN(I)=
    PI(24):GOTO83Ø
79Ø KR=RND(7)
8ØØ IFKR< .5THENSN(I)=SN(I-1)+1
81Ø IFKR>=.5THENSN(I)=SN(I-1)-1
```

HOME FINANCES
Commodore PET™

These programs were developed on a Commodore 2001 series, thirty-two kilobyte (32K required) personal electronic transactor with a single tape cassette.

Will categorize and review entries, give totals & tallies for all past-entered data (not a ledger; no dates or specific item names).

Great for use every 2-6 months in conjunction with check book records, stubs, bills, or your cancelled checks.

(a) Personal cash flow & tax deduction-related; 24 specific to general categories:
"tax accounts" $30
or "TAX ACCOUNTS" (No shift)

(b) Forty categories mostly specific for common home expenditures:
"house accounts" $50
or "HOUSE ACCOUNTS" (No shift)

(c) Both above programs; very slight overlap: $70

Choice desired . . . ________

Mail, with check or other payment and your address, to:

PractEd Tapes, Inc.
12162 S.E. 14th St.
Bellevue, Washington 98005
(206)747-8485

Programs not guaranteed against loss, misuse or system difficulties.

CBM/PET? SEE SKYLES ... CBM/PET?

SEE SKYLES ... CBM/PET? SEE SKYLES

... CBM/PET? SEE SKYLES ... CBM/

PET? SEE SKYLES ... CBM/PET? SEE

"Should we call it Command-O or Command-O-Pro?"

That's a problem because this popular ROM is called the Command-O-Pro in Europe. (Maybe Command-O smacks too much of the military.)

But whatever you call it, this 4K byte ROM will provide your CBM BASIC 4.0 (4016, 4032) and 8032 computers with 20 additional commands including 10 Toolkit program editing and debugging commands and 10 additional commands for screening, formatting and disc file manipulating. (And our manual writer dug up 39 additional commands in the course of doing a 78-page manual!)

The Command-O extends Commodore's 8032 advanced screen editing features to the ultimate. You can now SCROLL up and down, insert or delete entire lines, delete the characters to the left or right of the cursor, select TEXT or GRAPHICS modes or ring the 8032 bell. You can even redefine the window to adjust it by size and position on your screen. And you can define any key to equal a sequence of up to 90 key strokes.

The Command-O chip resides in hexadecimal address $9000, the rightmost empty socket in 4016 and 4032 or the rearmost in 8032. If there is a space conflict, we do have Socket-2-ME available at a very special price.

Skyles guarantees your satisfaction: if you are not absolutely happy with your new Command-O, return it to us within ten days for an immediate, full refund.

Command-O from Skyles Electric Works **$75.00**
Complete with Socket-2-Me **95.00**
Shipping and Handling .... *(USA/Canada)* **$2.50** *(Europe/Asia)* **$10.00**
California residents must add 6%/6½% sales tax, as required.

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

Visa/Mastercard orders: call tollfree **(800) 227-9998** (except California). California orders: please call **(415) 965-1735.**

IBM COMPATIBLE 8" FLOPPY DISK for CBM/PET,* Aim, Sym

- **HIGH SPEED** — at least FOUR times faster than the 488 bus disks.
- **HIGH CAPACITY** — up to 850K bytes of storage in a three drive system.
- **RELIABILITY** — wide timing margins insure long trouble-free operation.
- **SOFTWARE** — the PDOS II disk software provides a full set of BASIC commands with standard syntax.

5¼" MINI FLOPPY DISK SYSTEMS:	
Model 540-1 Single Drive, Double Density (143K)	$595.00
Model 540-2 Dual Drive, Double Density (286K)	$895.00
Model 580-1 Single Drive, Quad Density (286K)	$795.00
Model 580-2 Dual Drive, Quad Density (572K)	$1,195.00
8" MINI FLOPPY DISK SYSTEMS:	
Model 877-1 Single Drive, IBM standard (295K)	$1,095.00
Model 877-2 Dual Drive, IBM standard (590K)	$1,695.00

PEDISK II

fullFORTH +

FULL FEATURE "FORTH" FOR 6502 SYSTEMS
Interpreter — Cross-Compiler — Cond. Assembler —
Screen Editor — String Handling — Floating Point
SPECIFY PEDISK II, PET 2040 or 4040 DISK, OR APPLE $75.00

FOR INFORMATION, SEE YOUR DEALER OR:

ADD-ON DRIVES FOR TRS-80*	
Model I, 5¼", 40 Track (102K)	$329.00
Model I, 5¼", 80 Track (204K)	$429.00

cgrs **MICROTECH**

DEALER INQUIRIES INVITED

P.O. BOX 102 • LANGHORNE, PA 19047 • (215) 757-0284

*PET IS A TRADEMARK OF COMMODORE

```
820 PN(I)=PI(SN(I))
830 NEXTI
840 GOTO950
850 FORI=2TOL%
860 MX=SN(I-1)+MV
870 IFMX>25THENMX=25
880 MN=SN(I-1)-MV
890 IFMN<1THENMN=1
900 NO=MX-MN+1
910 CG=INT(RND(6)*NO)
920 SN(I)=MN+CG
930 PN(I)=PI(SN(I))
940 NEXTI
950 FORI=1TOL%:SN(I)=PN(I):NEXTI
960 GOSUB 1410:REM SET TIMES AND PL
    AY NOTES
970 GOTO360
980 REM ****************************
    ***
990 REM 1/F MUSIC *****************
    ***
1000 REM ****************************
    ***
1010 GOSUB 1190 :REM GET SONG DATA
1020 L%=32
1030 FORI=1TO5:D(I)=INT(RND(8)*6+1):
    NEXTI
1040 SN(1)=D(1)+D(2)+D(3)+D(4)+D(5)-
    5
1050 IFSN(1)<1THENSN(1)=1
1060 SN(1)=PI(SN(1))
1070 FORI=2TOL%
1080 IFI=17THEND(1)=INT(RND(8)*6+1)
1090 IFINT((I-1)/8)=(I-1)/8THEND(2)=
    INT(RND(8)*6+1)
1100 IFINT((I-1)/4)=(I-1)/4THEND(3)=
    INT(RND(8)*6+1)
1110 IFINT(I/2)<>I/2THEND(4)=INT(RND
    (8)*6+1)
1120 D(5)=INT(RND(8)*6+1)
1130 SN(I)=D(1)+D(2)+D(3)+D(4)+D(5)-
    5
1140 IFSN(I)<1THENSN(I)=1
1150 SN(I)=PI(SN(I))
1160 NEXTI
1170 GOSUB1410
1180 GOTO360
1190 REM *************************
1200 REM ASK FOR SONG DATA
1210 REM *************************
1220 PRINT"{CLEAR}{03 RIGHT}{03 DOWN
    DOWN}{REV}COMPOSITION DATA
    "
1230 IFTY$="4"THEN1270
1240 INPUT"{02 RIGHT}{DOWN}ENTER LEN
    GTH, IN NOTES";L%
1250 IFL%<=0THENPRINT"{DOWN}TOO SHOR
    T":GOTO1240
1260 IFL%>150THENPRINT"{DOWN}MAXIMUM
     LENGTH 150":GOTO1240
1270 INPUT"{02 RIGHT}{DOWN}DIFFERENT
     LENGTH NOTES (Y OR N) ";L
    $
1280 IFRIGHT$(L$,1)<>"Y"ANDRIGHT$(L$
    ,1)<>"N"THENPRINT"{DOWN}EN
    TER Y OR N":GOTO1270
1290 INPUT"{DOWN}{02 RIGHT}SLOW, MED
    IUM, FAST (S,M,F)";S$
1300 IFS$<>"S"AND S$<>"M"ANDS$<>"F"T
    HENPRINT"{DOWN}S,M, OR F":
    GOTO1290
1310 IFS$="S"THENS=1
1320 IFS$="M"THENS=2
1330 IFS$="F"THENS=4
1340 INPUT"{DOWN}{02 RIGHT}REPEAT NO
    TES (Y OR N) ";RP$
1350 IFRP$<>"Y"ANDRP$<>"N"THENPRINT"
    {DOWN}ENTER Y OR N":GOTO13
    40
1360 IFTY$<>"3"THEN1400
1370 INPUT"{02 RIGHT}{DOWN}MAX. VAR.
     FROM LAST NOTE ";MV
1380 MV=INT(MV)
1390 IFMV<=0THENPRINT"{DOWN}INVALID
    VALUE ":GOTO1370
1400 RETURN
1410 REM *************************
1420 REM GENERATE TIMES AND PLAY NOT
    ES
1430 REM ****************************
    ***
1440 IFL$="Y"THEN1490
1450 FORI=1TOL%
1460 ST(I)=16/S
1470 NEXTI
1480 GOTO1540
1490 W=64/S
1500 FORI=1TOL%
1510 R=INT(RND(4)*5+1)
1520 ST(I)=W/R
1530 NEXTI
1540 POKE59467,16:POKE59466,15
1550 FORI=1TOL%
1560 POKE59464,SN(I)
1570 T=TI
1580 IFTI-T<ST(I)THEN1580
1590 POKE59464,0
1600 GETA$:IFA$="X"THEN1630
1610 NEXTI
1620 IFRP$="Y"THEN1550
1630 POKE59467,0:POKE59466,0
1640 RETURN
```

©

P.I.E.-C

Interface CBM to Parallel Printer. Uses and Extends the IEEE-488 Bus.
Fully Compatible with ALL WORD-PROCESSING SOFTWARE!

Will Operate EPSON, IDS, OKIDATA, CENTRONICS, ANADEX, C. ITOH, NEC, PLUS ALL OTHER PARALLEL PRINTERS!

Switch Selectable Address 4 to 30. Switch for Direct/Converted Data. Professional Package and 6' Cable.

(301) 730-3257
$129.95 + $5 S&H (MD Res. +5 tax)

LemData Products
P.O. Box 1080, Columbia, MD 21044

AFFORDABLE — ONLY $50

MAILING LIST PROGRAM

FOR COMMODORE

16K or 32K — 40 or 80 column
2040 or 4040 Disk Drive

Write up to 1125 FILES, SORT, SEARCH, PRINT LISTS AND LABELS.
PLUS receive an UNDERSTANDABLE GUIDE!

ALSO AVAILABLE -- BIBLE HOCKEY GAME -- $10 --
ORDER GAME ON TAPE OR DISK

PSK EVALUATION SERVICE

1421 CAMERON LANE
MERCED, CA 95340
TELEPHONE: (209) 383-0405

WHEN ORDERING SPECIFY EQUIPMENT

Dealers – Reserve your copies of **COMPUTE**'s first Atari and PET/CBM books today. Call 919-275-9809 for ordering information.

GOLF

An instructional game for beginners. Realistic distances for each club (driver thru wedge). Perfect for the experienced golfer on a rainy day. 1 to 4 players. 9 or 18 holes. Accommodates for hooking and slicing. Graphics. Hazards-water, trees, sand traps, out-of-bounds. PGA rules. Computer scoring. Instructions.

For PET 16K: TRS-80 Mod. 1 Level II 16K (No Graphics)
Cassette $14.95 (N.Y.S. Residents Add 4 % Sales Tax)
Listing For Other Machines $4.95 (No Graphics)

Send Check or Money Order To:
R.L. HEFFNER
7 NEPTUNE LANE
MIDDLETOWN, N.Y. 10940

NEW SOFTWARE

VORTEX™

The enemy is at the center of swirling rings of energy. Your mission – penetrate and destroy.

PHANTOM FIGHTER™

Defend Superstition Mountain from attacking UFO's.

Two original arcade games for 40 column PETs. Machine language routines and graphics by Bob Carr and Kathy Higby, the originators of FROG, DANCER, OURANOS! and DROMEDA!

· · · Each 14.95* on cassette · · ·

Also Available

WASPS . . . . 9.95* Cassette
Defend yourself from killer insects!

GRAPHIC DUO . . . . 6.95* Cassette
Two terrific animated cartoons.

GAME PAC . . . . 6.95* Cassette
SUPERFROG! – AEROSOL! – WEATHER WAR!

MAGIC CARPET
P.O. Box 35115 Phoenix, AZ 85069

ARIZONA RESIDENTS ADD 5% SALES TAX.
Add $3 per order for shipping and handling. Foreign orders add 10% for air postage. Guarantee — if for any reason, one of our programs won't load, return it for free replacement.

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

Ghost Programming

Aric Wilmunder
Los Angeles, CA

I will show how it is possible for 16K Atari users to write and run BASIC programs normally requiring 24 or even 32K. This method is not at all like the method given to us in the BASIC manual where small programs simply call each other and passing of variables and arrays is difficult. Instead, this method is many times more powerful than chaining. Passing of variables is easy, and chaining is unnecessary.

In this article, I will explain how it is possible to write lines of code, subroutines, even entire programs without using any memory except for the space necessary for variables, arrays, and strings. How it is even possible to call and execute programs without changing or destroying the currently stored program. However, like every silver lining, mine too has a dark cloud – there are a number of restrictions involved. I will try to cover these restrictions thoroughly, but only after explaining the technique.

I should mention that, although all of the programming examples are disk oriented, all of the techniques used can be easily modified for cassette users.

After spending nearly four weeks trying to cram close to 40K worth of program into a 32K machine, I began to re-examine the problem of conserving memory. There are many ways to save memory space on the Atari, from removing I/O buffers on the DOS to complete recoding (of which I have done quite a bit). (A list of memory conservation techniques is included as part of *De Re Atari*, and anyone interested in writing large programs should become familiar with them.)

Instant Exec

What kept nagging me were the fifty or more lines of initialization code that were executed only once during my entire program. After their execution, these lines simply took up precious memory space which could be used for other purposes. Also, many of these lines are simply variable assignment statements like J = 12 or I = 1, or string assignments like A$ = "PHASERS." These statements must be executed at the beginning of each execution, but could be forgotten during execution.

Of the two types of assignments, variable and string, the string assignments concerned me the most. The statements DIM A$(26):A$ = "ABCD...Z" does not use only the 26 bytes for storing the string, but you are also using another 26 or more bytes for the assignment. The result is that your program is using more than twice the memory that is necessary in order to store a string. This may be no problem with smaller strings of up to fifty bytes, but, when using larger strings in a program where memory is already scarce, it can be quite alarming.

The method that has solved most of my problems goes something like this: create a file with all of the assignment statements used in the opening of the program in the same structure as a LIST file but minus the line numbers. For example: rather than having a LIST file that, when dumped, looks exactly like a program listing. You have the same line of code, but with commands only. The line:

```
1000 FOR I = 65 TO 90: ?CHR$(I);: NEXT I
```

would read:

```
FOR I = 65 TO 90: ?CHR$(I);: NEXT I
```

When entered, this line would act exactly as if it were typed on the keyboard by hand. At the beginning of my main program I use the command 'ENTER"D:<filename>" '. This command causes the system to enter each line of code from my Exec program and execute it using virtually no memory space.

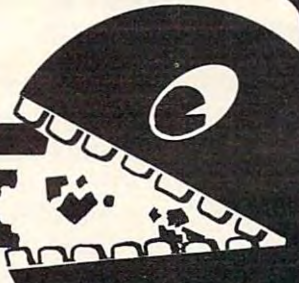
You can create a file with only an initialization routine, or go so far as to write an entire program with this method. To execute any of these programs you simply type 'ENTER' or 'E.', the extension and the file name. BASIC will treat this Exec Program exactly as if you were typing in each statement from the keyboard, thereby using no memory space for lines used only once. The amount of memory that can be saved from this method ranges from 5% to virtually an entire program's space.

One of the restrictions with this technique is that programs must be single step or step by step executable. The program must step one line at a time executing each line separately for the entire length of the program. Another restriction is that you cannot have multi-line FOR/NEXT loops (where both the FOR and the NEXT do not reside on the same line). The difficulty is in that, by the time the NEXT is encountered, BASIC will have discarded the FOR statement, giving the loop nowhere it can return to, and causing an error. The lines:

```
FOR I = 65 TO 90
?CHR$(I);
NEXT I
```

would have to be restructured into one single line. A simple test for writing and developing Exec Programs is to try to write the program by typing

TAKE A BYTE



OUT OF YOUR SOFTWARE COST!

SAVE 20% OFF RETAIL

JOIN **SOFTWARE GALORE!***
THE FASTEST GROWING MEMBERSHIP BUYING SERVICE FOR APPLE, TRS-80, ATARI, PET, IBM, CP/M SOFTWARE.

*Member: Better Business Bureau

We would like to give you an opportunity to come on board and become a **MEMBER** of **SOFTWARE GALORE.** The largest national **SOFTWARE BUYING SERVICE.** For a limited time only, **introductory** memberships are available to you at **50% off.** Now only **$7.50** (Reg. $15.00 value) and will be good for the period of one year. **Only members may buy from SOFTWARE GALORE.**

You'll find **we stock** the latest releases as well as older hard to find software products that most computer stores no longer carry. We feature these manufacturers:

Accent Software
Acorn Software
Activision
Avant-garde Creations
Adventure International
Arcade Plus
Astar International Co.
Automated Simulations
Avalon Hill Game Co.
Beagle Brothers
Big Five Software
Broderbund Software
Budgeco
California Pacific
Cavalier Computer
Computer Magic Ltd.
Context
Continental Software
Dankin 5 / Level 10
Datamost
Datasoft
Delta Software
Denver Software
Edu-ware
Edu-ware East
Gebelli
Glostronics
Hayden Software
Highlands Computer Service
Howard Software
Infocom
Innovative Design Software
ISA
IUS
Lazer Systems
LJK Enterprises
Micro/Lab International
Microsoft
Muse
On-Line Systems
Optimized Systems Software Inc.
Personal Software
Piccadilly
Phoenix Software
Quality Software
Riverbank Software
Sensible Software
Sentient Software
Sierra Software
Sirius Software
Sir-Tech Software
Softape
Software Emporium
Software Publishing Corp.
Spectrum Computer
Stoneware
Strategic Simulations
Sublogic
Synergistic Software
TG Products
The Book Company
The Cornsoft Group
The Logical Choice
Turnkey Software
United Software of America
Versa Computing
Videx Inc.
Voyager Software

Other "Discount" outlets "Blowout" a few items to get your business then charge you 100% list price for everything else.

At **SOFTWARE GALORE,** you'll save a full **20% OFF RETAIL** on everything you buy and be entitled to these additional services:

- An Additional 5% off on your first single item purchase.
- Access to the largest inventory of software!
- Low handling/shipping charges.
- Overnight shipments by **U.P.S.**
- 90 day guarantee on all software.
- Personalized, efficient service.

Software Galore backs its commitment of service and quality to its members with a 90 day unconditional warranty. You'll never get stuck with a lemon from us.

Catalogs and inquiries call our **Toll Free Numbers:**
1-800-772-2666 x 989 (in Calif.) 1-800-227-2634 x 989 (outside CA)
FOR IMMEDIATE ORDERS CALL 213-827-1851

Enclosed is ☐ 7.50 (for 1 yr.) ☐ 20.00 (for 2 yrs.) sign me up immediately and send a catalog, membership pack and order forms.
☐ Please send me a catalog and application only. I am under no obligation to buy.

Name: ☐ Mr. ☐ Ms.____________________
Street:____________________
City:__________ State:__________ Zip:__________
Type of Computer Used:____________________

Mail to: **SOFTWARE GALORE P.O. Box 10005, Marina del Rey, CA 90291**
Or Call: **1-800-772-2666 x 989 (in Calif.) 1-800-227-2634 x989 (outside Calif.)**

STATISTICS FOR RESEARCHERS
R-STAT

Friendly, easy to use, well documented.

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

DISKETTE & MANUALS **$350.00**

ATARI® DOS II
For a FREE Brochure or further information please call or write.

DSC CO: P.O. BOX 1035
CENTRALIA, WA 98531
(206) 736-7714

VISA & MASTERCHARGE ACCEPTED

ATARI™ 400/800*
SOFTWARE

AT LAST! A <u>NEW</u> GAME for your ATARI computer!!

Here is a game unlike any you have played before.

Requires a good memory for detail. Rewards strategy.

STACK ATTACK!!

For details write:

COLOR COMPUTER CONCEPTS
1275 Terry Drive,
Idaho Falls, ID 83401

or

Order a copy for $14.95.
Specify Disk or Cassette.
32K 16K
*Trademark of ATARI, INC.

each statement directly into the machine without using line numbers and then checking the results.

Another Restriction

Still another restriction is that, because EXEC programs have no line numbers, GOTOs and GOSUBs to points within the EXEC program are not allowed. However, if you currently have a program in memory, you can call outside routines that exist in your main program without affecting program control. Say you have a delay routine at line 100 in your main program; you can have your Exec Program GOSUB that line and then return to the next line of the EXEC program. If you want, you can even have a loop that will repeatedly call that routine. This technique is shown in Program 3.

In order to create EXEC program files like the one I described, I have written a simple demo program which will write them. In this demo, you write your own program starting at line 1000 and continuing anywhere up to line 9999. The program writes itself out to disk in a LIST file containing only the lines between 1000 and 9999. This LIST file is then opened as an input file and each line is read individually, the line numbers are removed, and the line is rewritten to a new file. When the program ends, you can test your file by typing: E."D:NOMEM.EXE.

If your Exec Program was properly written, the file should be executed and your original program will remain unchanged. If you tried the disk directory program, (Program 1), you would now have a program on disk which could be called at any time and would leave no leftover lines to be deleted later.

One feature which I should mention about this demo program is the ability to test your program before making a file. By typing GOTO and the line number of the first line of your program, you can follow the program execution and even make changes where necessary before creating an Exec Program. This is important because, if an error occurs anytime during execution, the EXEC program will stop and control will return to the monitor. For testing, type E."D:<Filename> and check for proper program flow. If problems arise, you can list the line numbers, make changes, and RUN the program again until all bugs are removed.

Transfer Of Control

Two aspects of using this method merit close attention. The first is that if you wish to enter this program from a running program, it is necessary to have a GOTO (next line in Main program) as the last statement. This will turn program control over from your Exec program to your Main program when the Exec is over. If this is forgotten, when the EXEC program is over, execution will stop. Since variables, arrays, and strings are passed on, the Main program can use variables from the Exec and vice versa.

The other interesting aspect is that keyboard input will be changed while the machine is reading from the file. The problem arises from the fact that, while the EXEC program is running, the machine acts as if all commands are being typed in directly on the keyboard. When a regular INPUT command is encountered, rather than inputing from the keyboard, the next piece of information will be read in from the disk. If a string is being input, that string will look like the next series of commands. The way around this is to open the keyboard as an input buffer. (OPEN #1,4,0,"K:") Strings and numeric values would then be entered in a loop using repeated GET commands and ending when a <CR> is encountered. The routine given will automatically terminate after a specified number of characters have been entered. (In the sample program, 20 characters are entered, but this can be changed by replacing both 20's in the routine with whatever you like.) The routine also tests for DELETE characters and modifies the string accordingly. For numeric values, you can simply let A = VAL(A$). This is shown in Program 2.

After you have tried a number of programs, you will notice that the prompt READY will appear after each line is executed. So far, I have no cure for this problem, but if one is found I'll be sure to let you know.

In a short period of time, you can build a substantial library of Exec functions. By changing the name of the output file, you can label the functions any way you find convenient. For example; E."D:DIR would display your current directory, and E."D:HEXDEC would convert hex values to decimal. Except for variable declarations, none of these would affect the current program in memory.

All in all, I have shown only a handful of the potential uses of Exec Programs. Other uses might include complex Batch jobbing and self-deleting line numbers. Any new ideas or feedback about this technique would be greatly appreciated. Like many aspects of the Atari, I feel that we are still only beginning to understand the full potential of this fantastic machine.

Main Program

```
100 DIM A$(500)
110 TRAP 200
120 LIST"D:XYZZY.TMP",1000,9999
130 OPEN#1,4,0,"D:XYZZY.TMP"
140 OPEN#2,8,0,"D:NOMEM.EXE"
150 INPUT#1;A$
160 PRINT#2;A$(6)
170 GOTO 150
200 IF PEEK(195)<>136 THEN ?"ERROR -";
```

```
     PEEK(195)
210  CLOSE#1
220  CLOSE#2
230  END
```

Program 1: Disk Directory

```
1000 GRAPHICS 0:CLOSE#1:OPEN#1,6,0,"D:*.*"
     : FOR I=1 TO 999:GET#1,A:?CHR$(A);:
     IF A<>155 OR B<>83 THEN B=A: NEXT I
```

Program 2: Input A Value

```
1000 CLOSE#1:?"ANSWER?";:OPEN#1,4,0,"K:":
     FOR I = 1 TO 20: GET#1,A: ?CHR$(A);:
     A$(I)= CHR$(A): I=I+20*(A=155)-2*(A=
     126): NEXT I
```

Program 3: Calling Outside Routines

```
500 FOR I=0 TO 127
510 PRINT CHR$(27);CHR$(I);
520 NEXT I
1000 FOR J = 1 TO 5: GOSUB 500: NEXT J
```

Program 4: List Program Variables

```
1000 J=PEEK(130)+256*PEEK(131)
1010 FOR J=J TO PEEK(132)+256*PEEK(133)-1:
     ?CHR$(PEEK(J)-128*(PEEK(J)>127));CHR$
     (27+128*(PEEK(J)>127));:NEXT J
1020 I=0: FOR J=PEEK(130)+256*PEEK(131) TO
     PEEK(132)+256 * PEEK(133)-1: I = I +
     (PEEK(J)<127):NEXT J: ? I;" VARIABLES"
```

©

Look Out!

MX 80

PRO/WRITER

IS HERE

ATARI D.D.
$460.00

ATARI 800
$735.00

PRO/WRITER OFFERS YOU MORE

- Tractor & Friction Feed
- Paper Cut Off
- The Tightest Dot Matrics You've Ever Seen
- Hi Res. Key & Dot Graphics
- 100 CPS
- Proportional Spacing
- Math Symbols
- And More For Less
- Atari to Pro/Writer Word Processor Free with Printer
- Atari
- Zenith
- Printers
- Paper
- Elephant Disks
- Software
- Atari to Pro/Writer Software

Poquette's
14538 Lynshar Road, Grass Valley, CA 95945
Dealer inquiries invited (916) 272-6808

STARTING YOUR OWN MICROCOMPUTER BUSINESS

Starting your own microcomputer business is easy if you know the right steps to take. Two volumes of the new book **Your Fortune In The Microcomputer Business** describe the things you should know to start right and to build your business successfully.

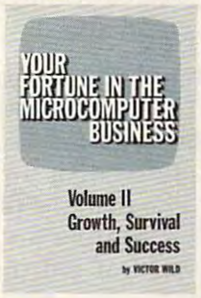
Many people have good ideas for a successful microcomputer business. But they don't know how to put their ideas into action. These people are plenty smart. That's not the problem. The problem is in knowing simple things: How to analyze the market. How to select the right product or service. How to get enough startup money. How to plan your growth and success. The answers to these problems are not hard to understand. All you need is the right information. **Your Fortune In The Microcomputer Business** gives you the knowledge tools to start right, grow and prosper.

This practical reference manual has no hocus-pocus. It **does not** tell you how to do astrology or dating services. It **does** give you clear, complete, step-by-step instructions on how to get started right and insure your success!

Volume I Includes:

- The hottest trends in the market
- The seven best test markets
- How to read between the lines at a trade show
- 107 ways to reach your market
- Examples of the best ads in the business
- How to get free advertising
- The 21 steps to set up your business
- How to start a manufacturing business
- How to write a newsletter
- How to give a seminar
- How to start a service business
- How to package software for the mass market
- How to be a highly paid consultant
- Franchising—the good, the bad and the ugly
- Interviews with six successful microcomputer businesses
- How to use the RLC factor to be street smart, lean, mean and successful

Volume II Includes:


YOUR FORTUNE IN THE MICROCOMPUTER BUSINESS
Volume II
Growth, Survival and Success
by VICTOR WILD

- Strategies for growth
- Straight facts on incorporation
- How to build your organization
- How to manage cash flow
- How to develop your accounting system
- How to manage employees, wages and salaries
- How to survive a cash crunch
- What to do if things get really bad
- How to maximize your profits
- How to grow by acquisition
- How to value your business
- How and when to sell your business

WRITE OR CALL—DON'T WAIT!

Wildfire Publishing, 326 DJ Toro Canyon
Carpinteria, CA 93013 Ph. (805) 684-1489

Please send the following book(s) by return mail. I understand if I want to return them **for any reason** within 90 days of receipt, I can do so and get a **prompt full refund.**

☐ **Your Fortune In The Microcomputer Business Vol. I, Getting Started, $20.00.**

☐ **Your Fortune In The Microcomputer Business Vol. II, Growth, Survival and Success, $20.00.**

Please add $1.50 per book for shipping, Calif. residents please add $1.20 each sales tax.

Name ______
Street ______
City ______ State ______ Zip ______
Enclosed is ______ or charge my MasterCharge ☐ or Visa ☐
Card # ______
Ex. Dt. ______ MCInterbank # ______
Signature ______ © 1982 Wildfire Pub.

BASIC 4.0 To Upgrade Conversion Kit

Elizabeth Deal
Malvern, PA

Q: When is a NEXT not a NEXT?
A: When it's a DCLOSE command from Basic 4, of course.

This article is intended primarily for users of the Upgrade PET/CBM systems. It discusses several BASIC 4 disk commands, as they appear on the Upgrade screen.

BASIC 4 programs can often run in the upgrade system with or without conversion. But, to convert, one must know the author's intent in the program and the Upgrade system obliterates the necessary information. Reflect on a three way analogy you might, some day, see on your screen:

NEXT = RETURN WITHOUT GOSUB = DCLOSE

It looks curious, but it makes sense.

A Bit Of History

Some time ago, I had the pleasure of using a BASIC 4 CBM. I was writing a relative file program. At one point I *had* to renumber the program, CBM couldn't do it for me, and the only sensible solution was to load the program into my trusty old Upgrade PET equipped with Toolkit™. I listed the program to see how the disk commands would behave in a new environment.

Assorted quotes from BASIC 4:

```
300  FOR I = 1TONF:RECORD#(DF),(CR),(FP%(I))
310  PRINT#DF,F$(I):GOSUB230:NEXT:RETURN
READY.
2020  DOPEN#(DF),(FF$),D(DD),L(RS)  :GOSUB230
2030  RECORD#(DF),)NR):GOSUB230:PRINT#(DF),
      CHR$(255):GOSUB230
2040  CLOSEDF:GOSUB230:OPENDF:GOSUB230:FR
      = 1:RETURN
READY.
3090  SCRATCH(KY$)
READY.
4020  DCLOSE
READY.
```

As seen by the Upgrade system:

```
300  FORI = 1TONF:DATA#(DF),(CR),(FP%(I))
310  PRINT#DF,F$(I):GOSUB230:NEXT:RETURN
READY.
2020  FOR#(DF),(FF$),D(DD),L(RS)  :GOSUB230
2030  DATA#(DF),(NR):GOSUB230:PRINT#(DF),
      CHR$(255):GOSUB230
2040  CLOSEDF:GOSUB230:OPENDF:GOSUB230:
      FR = 1:RETURN
READY.
3090 GOSUB(KY$)
READY.
4020  NEXT
READY.
```

The screen showed FOR# where DOPEN# should have been (line 2020), and DATA# where a relative file statement RECORD# should have been (lines 300 and 2020). Worse still, it translated SCRATCH(KY$) into GOSUB(KY$) in line 3090. Finally, a conversion of a simple DCLOSE into NEXT (line 4020) seemed incredible.

Both the Toolkit and the PET left those keyword tokens intact (I did not retype the BASIC 4 keywords, doing that would have destroyed them). The program worked fine after transfer to the BASIC 4 computer. And that was that.

Recently, I had to look at that undocumented mess of code. I remembered some of the nasties, but couldn't recall them all. Several of these commands leaped out in a listing as invalid ones, but I didn't catch NEXT, of course. It seemed to belong. However, Power didn't let this one slip by.

While scrolling through the program, back and forth, looking for additional trouble, I noticed that GOSUB(KY$) translated into STRING TOO LONG(KY$) and there appeared a strange looking 4020 RETURN WITHOUT GOSUB statement. That was my NEXT. (I cannot provide a printout, because to print we use the LIST command, whereas these two long sentences were not done by LIST, they resulted from scrolling.)

I was lucky in that I was looking at a program I had written and had a vague idea of what it did. But imagine, for an instant, that somebody sends you a program containing BASIC 4 disk commands. How can you go about finding out which are used? How can you distinguish the true Upgrade commands, like NEXT from BASIC 4 disk commands?

Solution

It always helps to understand the process. The Power manual was useful in solving this one for me, because it explained where and how Power, and the PET for that matter, pick up the keywords and error messages contained within ROM.

One way to get at the keywords is to look in ROM in both Upgrade and BASIC 4 systems and produce a side-by-side listing of tokens and messages. The search addresses were taken from memory maps.

DISKSHARE 9000

Multi User System

- Works with all combinations PET/CBM models.
- 100% Hardware. No software required!
- No special commands, User TRANSPARENT!
- Up to 8 users can share a disk &/or printer.
- 10, 15, 20, & 25 foot cables available.
- Connects in minutes, no tools necessary.
- Ideal for EDUCATIONAL & BUSINESS needs.
- Increases productivity – Reduces Costs!

DISKSHARE 9000-3	3 USER SYSTEM WITH 6ft. CABLES	$ 749.00
DISKSHARE 9000-8	8 USER SYSTEM WITH: – THREE 6ft. CABLES – THREE 10ft. CABLES – TWO 15ft. CABLES	$1149.00
DISKSHARE 9001-6	ADD-ON CABLE (6 FOOT LENGTH)	$ 79.95
DISKSHARE 9001-10	ADD-ON CABLE (10 FOOT LENGTH)	$ 99.95
DISKSHARE 9001-15	ADD-ON CABLE (15 FOOT LENGTH)	$ 119.95
DISKSHARE 9001-20	ADD-ON CABLE (20 FOOT LENGTH)	$ 139.95

For More Information Call:

Questar International Inc.

Suite 102
7270 Woodbine Avenue,
Markham, Ontario Canada
(416) 490-8044

dealer and
distributor inquiries
invited

I used this routine:

```
140 N=0:F=1:M=128:P=127:TP=PEEK(50003)
150 S=49298:E=49812:REM UPGR & ORIG
160 IFTP=160THENS=45234:E=45858:REM 4
    80 COLUMN
170 FORJ=STOE
180 IFFTHENF=0:PRINT:PRINTN;N+M;:N=N+1
190 V=PEEK(J):IFVANDMTHENV=VANDP:F=1
200 PRINTCHR$(V);
210 NEXTJ
READY.
```

The results are shown in Figure 1. A list nearly identical to the BASIC 4 listing was in **COMPUTE!** #15, and the list of the Upgrade tokens was in **COMPUTE!** #1. The list presented here also adds the messages which follow the list of tokens.

Note that tokens on the Upgrade PET range in number from 128 to 203. From 204 down we have the PET-people interface. On the BASIC 4 systems, tokens range from 128 to 218 with tokens 128-203 being common between the two systems. Messages follow the tokens and begin at number 219.

The tokens that give us trouble are the ones in BASIC 4 numbered 204-218. They line up with Upgrade PET's messages or with the beginning of the token list, depending who is doing the lining up, LIST or Power's scroller.

The Logic Of It All

The reason behind it goes like this (I think) : The program that runs the PET, the BASIC interpreter, takes a BASIC 4 token that was loaded in, for instance token 206 (DCLOSE). In order to print it on the screen, it scans the table looking for 206. But the Upgrade PET knows that the highest valid token number in its list is 203. When the list is exhausted, it wraps around and starts at the top of the list, goes down three more items and, consequently, returns an inconspicuous NEXT. Power, on the other hand, doesn't wrap around. When a token, invalid for the system, exists in the program, it goes down the list to number 206 and finds a clearly visible RETURN WITHOUT GOSUB message, equivalent to DCLOSE. All quite logical. And simple.

The conversion kit, therefore, consists of a list of tokens and messages. By some careful work on your part, BASIC 4 programs can be read on an Upgrade screen. If you see a strange looking command, you can find out what it means by aligning the tokens and messages.

Try to guess what BASIC 4 statement is intended when the LIST command says END and Power's scroller says NEXT WITHOUT FOR? How about LIST showing GOTO and the scroller showing REDIM'D ARRAY?

Subsequent to the disk commands having been decoded from their curious appearance, the only remaining job is to rewrite those commands into words Upgrade PET can understand (to achieve reverse compatibility). Relative file commands cannot be converted that easily. For this you might consult reference (4) below. If you see RECORD # scattered in the BASIC 4 program, you'll need to do some work. In any case, make sure that you add a semicolon at the end of all PRINT statements. Other commands can be translated with little difficulty by consulting the disk manual, once you know what they are supposed to be.

Don't Jump To Conclusions

WARNING: Trying to write a BASIC 4 program on an Upgrade PET cannot work easily. Writing FOR#4 will not result in DOPEN#4, unless you scan the program and add 75 to the selected FOR token value leaving intended FORs alone. It makes no sense to try to do it, because you couldn't debug your hybrid creation anyway.

REFERENCES:

1) Butterfield's Memory maps in **COMPUTE!** *issues 2 and 7.*
2) POWER Manual (Professional Software).
3) User's Manual for CBM 5¼-inch Dual Floppy Disk Drives, Commodore Business Machines, part # 320899.
4) Butterfield's Mixing and Matching Commodore disk system.

I am grateful to COMPUTER FORUM of FRAZER, PA for permitting me to use their BASIC 4 equipment.

For your Commodore CBM and VIC needs contact:

PETTED MICROSYSTEMS

4265 W. Loomis
Milwaukee, WI 53221
(414)282-4181

DR. DALEY Introduces...

THE WIZ

Data Management System

THE WIZ is here!!!

THE WIZ, a system powerful enough to manage most of your data storage and manipulation needs - yet is easy to use. A system we are so sure of that we have an offer you can't resist. First though, let's take a look at a few of the many features of this program.

Feature	Benefit
1. ON-LINE help	At your fingertips is the equivalent of a 60 page manual. At any time the computer is waiting for a response from you, you may press the 'h' key or type 'help'. THE WIZ will then provide you with an explanation of the function you are working with.
2. Plotting capability	This is a feature unique to THE WIZ. It can produce a bar graph with up to 18 bars or a histogram with up to 100 points plotted. Graphically presented data is easy to interpret.
3. Wordpro interface	This option is standard with THE WIZ. With many of the competing data managers, if available, it is an extra cost option.
4. Read a sequential file	You may reorganize your files or even read sequential files generated by other data management systems.
5. Search for keywords	Here you can search for a word in ANY field in your record. It can even ignore differences due to upper case and lower case characters.
6. Constants in data entry	You may store up to three separate sets of contant fields. Each set can have as many fields as you like filled with information. Then two keystrokes will call the appropriate set.

And there is more. There is not room enough to tell you all the features in a one page ad!

SPECIAL OFFER

You will like our system. In fact we are so convinced of this, that we are going to pay you to try it! If you have another commercially available data management program, you can receive TRADE-IN credit for your purchase of THE WIZ. Call us on our toll free number (800) 548-3289 for our offer on your present system. Remember that this offer expires March 15, 1982.

DR. DALEY'S SOFTWARE

Water St. Darby, MT. 59829

(800) 548-3289 (for Orders & Information)

(406) 821-3924 (Montana and technical assistance)

Call between 9 a.m. and 6 p.m. Mountain time Monday through Friday

UPGRADE			BASIC 4		
0	128	END	0	128	END
1	129	FOR	1	129	FOR
2	130	NEXT	2	130	NEXT
3	131	DATA	3	131	DATA
4	132	INPUT#	4	132	INPUT#
5	133	INPUT	5	133	INPUT
6	134	DIM	6	134	DIM
7	135	READ	7	135	READ
8	136	LET	8	136	LET
9	137	GOTO	9	137	GOTO
10	138	RUN	10	138	RUN
11	139	IF	11	139	IF
12	140	RESTORE	12	140	RESTORE
13	141	GOSUB	13	141	GOSUB
14	142	RETURN	14	142	RETURN
15	143	REM	15	143	REM
16	144	STOP	16	144	STOP
17	145	ON	17	145	ON
18	146	WAIT	18	146	WAIT
19	147	LOAD	19	147	LOAD
20	148	SAVE	20	148	SAVE
21	149	VERIFY	21	149	VERIFY
22	150	DEF	22	150	DEF
23	151	POKE	23	151	POKE
24	152	PRINT#	24	152	PRINT#
25	153	PRINT	25	153	PRINT
26	154	CONT	26	154	CONT
27	155	LIST	27	155	LIST
28	156	CLR	28	156	CLR
29	157	CMD	29	157	CMD
30	158	SYS	30	158	SYS
31	159	OPEN	31	159	OPEN
32	160	CLOSE	32	160	CLOSE
33	161	GET	33	161	GET
34	162	NEW	34	162	NEW
35	163	TAB(	35	163	TAB(
36	164	TO	36	164	TO
37	165	FN	37	165	FN
38	166	SPC(	38	166	SPC(
39	167	THEN	39	167	THEN
40	168	NOT	40	168	NOT
41	169	STEP	41	169	STEP
42	170	+	42	170	+
43	171	-	43	171	-
44	172	*	44	172	*
45	173	/	45	173	/
46	174	↑	46	174	↑
47	175	AND	47	175	AND
48	176	OR	48	176	OR
49	177	>	49	177	>
50	178	=	50	178	=
51	179	<	51	179	<
52	180	SGN	52	180	SGN
53	181	INT	53	181	INT
54	182	ABS	54	182	ABS