



## MALA

Educational Programs  
For Your Pet For Only

# \$48

For 8 Issue Subscriptions

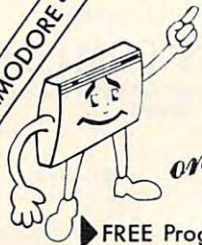
Each "MALA" comes to you on cassette with 4 brand new programs to load and run on your Pet.

### COMM\*DATA SYSTEMS, INC.

P.O. BOX 325  
MILFORD, MICHIGAN 48042  
(313) 685-0113

### GAMES FOR YOUR VIC

VIC TREK • HANGMAN  
TANK • YAHTZEE  
DAMSEL • NAB PLUS  
AND MANY MORE



# only \$7.95 Each

▶ FREE Program Summaries available

## Your VIC-20 Will Smile...

### SUPERFONT

Design programmable characters on your VIC-20 with this easy-to-use program. Especially useful when creating animations, since you can edit four characters at once as a 2x2 block.

### SAFARI

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

### QUIX

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

These three excellent programs are available on the VIXEL #2 cassette for the standard 5K VIC-20 for only \$12.95 in the US and Canada. Foreign orders add \$3.50 for shipping. CA residents add 6% tax. VISA and MasterCard welcome.

VIXEL is a trademark of The Code Works.  
VIC-20 is a trademark of Commodore Business Machines.

### The Code Works

Box 550, Goleta, CA 93116 805/683-1585

## nüFEKOP... VIC 20 ...

Sirs:

What a game! I loved it! Finally a company that sells a great product for a low price. I have ordered from other companies before and never have I got a product like this before!!!

Bryan Niderspohn

275 West Bartlett Rd

Lynden, Washington 98264

Thank You

Thank You

Thank You!!!!!!!!!!!!!!

## ...nüF-SAID?

**ANY 5 FOR \$42.50 : ANY 3 FOR \$27.00 : ANY 1 FOR \$9.99 : OUR CATALOG \$.50**

**SEARCH** - Very good! Currently the most played game in the office! Try this one.

**ALIEN PANIC** - Climb ladders, dig holes and drop aliens ... Quickly!

**KRAZY KONG** - You have to climb to the top but this crazy monkey keeps rolling barrels!

**VIKMAN** - Sweep up dots before monsters mop you up.

**QUIRK** - Fun, fun, fun. Leaves you wanting to play again and better your score. Fast Action.

**RESCUE from NUFON** - Explore 100 (5K, VIC?!). Rooms avoiding monsters and teleporting bipeds.

**DODGE CARS** - Fast fun on crowded freeway. This colorful game keeps top 5 & compares against last 50.

**INVASION** - Protect energy pods from being stolen by enemy ships. Simple and challenging.

GAMES FOR THE "VIC 20"

Registered Trademark COMMODORE BUSINESS MACH.

WRITE

## nüFEKOP

P.O. BOX 156

SHADY COVE, OREGON 97539

\*\*SPECIFY KEYBOARD OR JOYSTICK\*\*

OR CALL 503-878-2113 FOR  
MASTER CARD, VISA OR C.O.D.



```

1 REM AMORTIZE --BY
2 REM AMIHAI GLAZER
3 REM UNIV. OF CALIF.
4 REM IRVINE,CA.92717
5 DEF FNR(X)=INT(100*          X+.5)
    /100
10 PRINT "{CLEAR}{REV}AMORTIZE"
20 PRINT "{03 DOWN}"
30 PRINT "NO. OF PERIODS"
40 PRINT "    (IN MONTHS)"
50 PRINT "N= ";:GOSUB      63990
60 PRINT "ANNUAL %INTEREST RATE"
70 PRINT "AR=";:GOSUB      63990
80 MR=AR/1200
90 PRINT "PRINCIPAL"
95 PRINT "P=";:              GOSUB 63
    990
100 PMT=(P*MR)/(1-(1+      MR)^(-N)
    )
105 PMT =FNR(PMT)
110 PRINT "{02 DOWN}PMT=",      F
    NR(PMT)
111 PRINT "{DOWN}PRESS RETURN KEY"
112 PRINT "TO CONTINUE OR STOP"
113 GETA$:IF A$=""          THEN 113

120 PRINT "{02 DOWN}"
130 FOR I=1 TO N
132 GET A$:IF A$=""          THEN GOT
    O 140
134 GET A$: IF A$<>" "      THEN GOT
    O 134
136 GET A$:IF A$=""          THEN GOT
    O 136
140 RDUE =FNR(P*MR)
150 CUMR=FNR(CUMR+RDUE      )
160 P=P-PMT+RDUE
170 PRINT "{REV}MONTH=";I
180 PRINT " PRINCIPAL =" ;FNR(P)
190 PRINT " TOTAL INT.=" ; (CUMR)
200 PRINT " INT. DUE =" ; (RDUE)
210 NEXT I
220 END
63990 POKE 204,0:POKE  207,0:GET A$
63991 IF A$<>CHR$(13) THEN PRINT A$;
    :GOTO 63990
63992 PRINT " {WHT}"
63993 PRINT "GO63996"
63994 POKE 631,145:      POKE632,145:P
    OKE633, 145:POKE634,145:P
    OKE 635,13
63995 POKE 636,145:      POKE637,145:P
    OKE638, 13:POKE198,8:END
63996 PRINT"{02 UP}":FOR ZZ =1TO3:PR
    INT"{BLU}"          ":NEXT:PR
    INT"{03 UP}"
63997 RETURN

```

©



## 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.

- Random access file organization for fast updating of individual records
- Allows various 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 COMPUTING#**  
8941 Owensmouth Ave. #202  
Canoga Park, CA 91304  
(213) 700-1166

Special introductory price \$179.95. Requires 32K and two Atari\* 810 disk 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.



400 16K .....	\$319.00
400 YOURS to 32K or 48K .....	CALL
800 (16K) .....	659.00
410 RECORDER .....	84.00
810 DISK DRIVE .....	449.00
850 INTERFACE .....	169.00
830 MODEM .....	149.00
825 PRINTER .....	575.00
481 ENTERTAINER KIT .....	85.00
482 EDUCATOR KIT .....	125.00
483 PROGRAMMER'S KIT .....	60.00
484 COMMUNICATOR KIT .....	309.00

Prices subject to change without notice.  
Shipping extra. No tax out of state.  
Ca. residents add appropriate taxes.

WE ARE AN AUTHORIZED ATARI SALES AND  
SERVICE CENTER



**COMPUTERTIME, INC.**

P.O. Box 216  
Kentfield, CA 94914

CALL TOLL-FREE  
In California

800-227-2520  
800-772-4064



*The Atari permits an excellent graphics animation technique — Player/Missile Graphics. However, smooth horizontal motion is easier to achieve than vertical; this article shows how to solve this problem using USR.*

# No Commotion Motion

Tina Halcomb  
Carrollton, TX

In this article I will cover a fast motion routine for Player/Missile graphics.

The Atari has a built-in hardware register called the horizontal position register, which applies only to Player/Missile graphics. When the value in this register is changed, the Player/Missile moves to it's new position. If you were to change the number continuously by small amounts (such as in a program loop) you can obtain a smooth, sweeping motion.

Unfortunately, there is not a register available to us that pertains to the vertical position of the Player/Missile. Vertical movement can be achieved easily, however, by adjusting the RAM which represents this position. In routines that I have used in the past, the Player/Missile image was erased from the old position and redrawn in the new position. This technique works fine; however, any time you draw a picture you have to define its shape. This means that for each different shape, you need a separate drawing routine. Moving an image in this manner also produces a crawling effect.

I found that by rotating all 256 bytes of the Player-Missile directly in memory, the movement looked much like that of the horizontal move. And since we are moving all 256 bytes of the image, the shape is not important.

A program of this nature written entirely in BASIC would run very slow, and there would be no advantage to it. If I had attempted to write this program in assembly language, it would be obsolete before it was finished.

The USR function in the Atari allows you to add assembler subroutines to your BASIC programs. By using this function I was able to draw the Player/Missiles with BASIC and move them with my assembler routine. Line 90 of the BASIC

program shows the parameters that the USR function operates from. The first number, 1536, is the address of the Assembly language routine. The second number is the actual Player/Missile number. The third and fourth numbers are the X and Y offsets (or how far you want to move the image), respectively.

In Player/Missile graphics RAM positioning, the first section of RAM is unused. The size of the unused or "free" RAM area is dependent upon the

Figure 1.

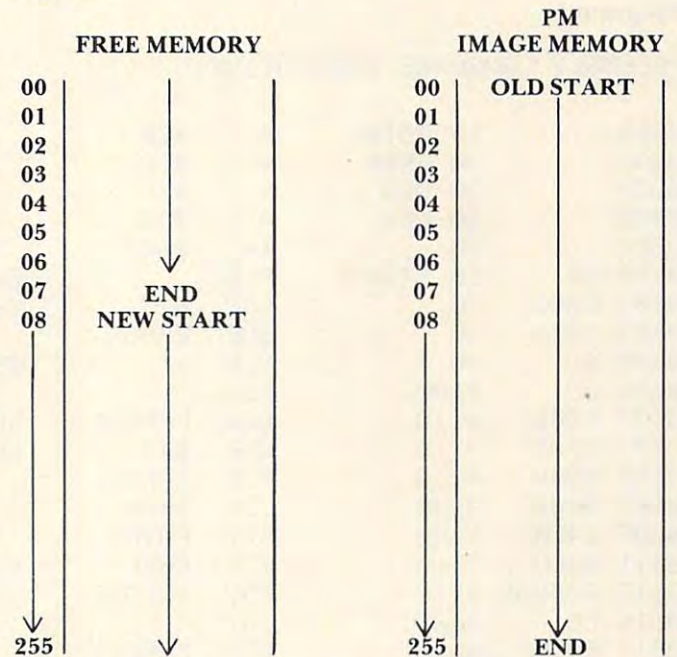
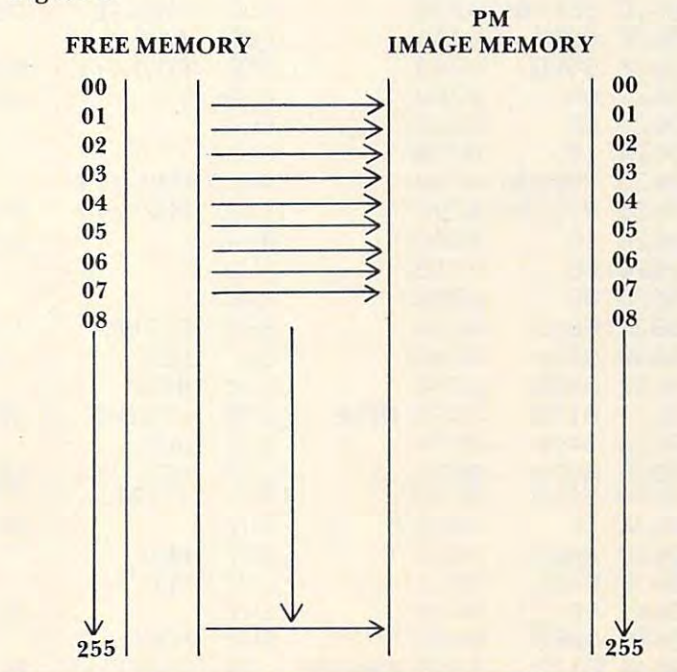


Figure 2.





line resolution that you choose to use. For the purpose of this program, I will discuss single line resolution. But bear in mind that the same program can be used for double line resolution with a few minor adjustments. So we have 768 bytes of free RAM at the top of our PMBase and the allocated RAM area for each player is 256 bytes.

To move the image I reposition the entire 256 bytes, with respect to the Y offset, into the free RAM space. For example, if the Y offset is seven, then byte zero of the player's present memory

location is moved to byte seven of the free memory. This continues until byte 255 of the player is moved into byte six of the free memory and the rotation is complete. Figure 1 illustrates how this is done.

At this point the image is transferred byte for byte back to its original memory location (Figure 2.), and the Player/Missile image has swiftly moved into its new screen position.

With the speed and flexibility provided by this assembly language subroutine, you can write complex Player/Missile programs, all from BASIC.

### Program 1.

#### ASSEMBLY LANGUAGE SUBROUTINE

```

00CB      10 POINT      =    $CB
00CD      20 FREE      =    $CD
00CF      30 OLD       =    $CF
00D0      40 NEW       =    $D0
00D1      50          *=    $600
0600 68      60 START   PLA          GET # OF PARAMETERS IN USR
0601 C903     70        CMP    #3     SHOULD BE 3
0603 D054     80        BNE    ERROR
0605 68      90        PLA          GET PLAYER #, THROW AWAY 1ST BYTE
0606 68     0100       PLA
0607 F04E    0110      BEQ    ERROR1   NO PLAYER 0
0609 C905    0120      CMP    #5     NO PLAYER GREATER THAN 5
060B B04A    0130      BCS    ERROR1
060D A000    0140      LDY    #000
060F 84CB    0150      STY    POINT
0611 84CD    0160      STY    FREE    SET UP FREE POINTER
0613 AC6006  0170      LDY    PMBASE  USING PMBASE+256 FOR FREE MEMORY
0616 C8      0180      INY
0617 84CE    0190      STY    FREE+1
0619 18      0200      CLC          GET START OF MEMORY IMAGE FOR THIS PM#
061A A8      0210      TAY
061B 6D6006  0220      ADC    PMBASE  ADD IN UNUSED MEMORY
061E 6903    0230      ADC    #3
0620 850C    0240      STA    POINT+1 PUT IT IN POINTER
0622 68      0250      PLA          GET THE X OFFSET, THROW AWAY FIRST BYTE
0623 68      0260      PLA
0624 18      0270      CLC
0625 796106  0280      ADC    PMX-1,Y
0628 996106  0290      STA    PMX-1,Y PUT IT IN THE X REGISTERS
062B 68      0300      PLA          GET THE Y OFFSET, THROW AWAY FIRST BYTE
062C 68      0310      PLA
062D A8      0320      TAY
062E F01B    0330      BEQ    RETURN  IF 0 SKIP THIS SECTION
0630 84D0    0340      STY    NEW     USE Y TO ADD OFFSET TO MOVE
0632 A000    0350      LDY    #000
0634 B10B    0360 MOVE   LDA    (POINT),Y GET BYTE OF IMAGE
0636 84CF    0370      STY    OLD
0638 A400    0380      LDY    NEW     GET OFFSET
063A 91CD    0390      STA    (FREE),Y PUT BYTE IN FREE WITH OFFSET
063C C8      0400      INY          ADVANCE OFFSET
063D 84D0    0410      STY    NEW
063F A4CF    0420      LDY    OLD
0641 C8      0430      INY          ADVANCE POINTER
0642 D0F0    0440      BNE    MOVE
0644 B1CD    0450 CHANGE LDA    (FREE),Y MOVE REARRANGED IMAGE BACK

```



# MOONBASE IO

● the battle for the moons of Jupiter ●



Blast your way through the alien mine fields! Defend the moonbases from an attacking alien armada! It won't be easy. To win you will need to mount a heroic assault on the alien mother ship.

**Moonbase Io** combines three exciting arcade adventures in one exciting game. The machine-language program by John

Konopa uses advanced graphics and sound effects. Action is fast and exciting — varying levels of skill required to go from one part of the game to the next.



BEYOND SOFTWARE's arcade-adventures are the next generation in computer games for the ATARI<sup>™</sup> computer. A voice-activated program will help you meet and overcome the challenge — it may be an alien invasion, a fiendish murderer preying on a country village or a treasure trove buried deep in the shark-infested sea.

**Moonbase Io** is available at fine computer dealers. Or, directly from PDI for \$29.95 plus \$2.00 shipping and handling.

Requires 24K ATARI<sup>™</sup> computer with disk and cassette.  
Cassette version available soon.



Program Design, Inc., 11 Idar Court, Greenwich, CT. 06830 203-661-8799

ATARI<sup>™</sup> is a registered trademark of Atari, Inc.

[www.commodore.ca](http://www.commodore.ca)



```

0646 910B 0460 STA (POINT),Y
0648 C8 0470 INY
0649 D0F9 0480 BNE CHANGE
064B A005 0490 RETURN LDY #05 WRITE THE X POSITION REGISTER TO THE HARD
WARE REGISTER
064D B96106 0500 PUT LDA PMX-1,Y
0650 99FFCF 0510 STA 53247,Y
0653 88 0520 DEY
0654 D0F7 0530 BNE PUT
0656 60 0540 RTS GO BACK TO BASIC
0657 A902 0550 ERROR1 LDA #02 ONLY 2 ITEMS LEFT OF STACK IF ENTERED HER
E
0659 A8 0560 ERROR TAY A HOLDS # OF ITEMS ON STACK WHEN ENTERED
HERE
065A 68 0570 PULL PLA PULL TWICE FOR EACH ITEM
065B 68 0580 PLA
065C 88 0590 DEY
065D D0FB 0600 BNE PULL
065F 60 0610 RTS
0660 0620 PMBASE = * STACK IS RESTORED, GO BACK TO BASIC
0661 0630 TEMP = ++1 CONTAINS MSB OF MEMORY FOR PMGRAPHICS
0662 0640 PMX = ++2 5 X POSITION REGISTERS
0660 0650 .END

```

#### Program 2.

```

1 REM LOAD ASSEMBLY ROUTINE INTO MEMORY
4 GRAPHICS 0
5 RESTORE 110
6 B=1536:I=0
7 FOR L=B TO B+100
8 READ A:I=I+A
9 POKE L,A
10 NEXT L
11 IF I<>13309 THEN PRINT "CHECK DATA ST
ATEMENTS FOR ERRORS":STOP
15 SETCOLOR 2,0,0:Y=48
19 REM CALCULATE ADDRESS FOR PLAYER-MISS
ILE GRAPHICS
20 A=PEEK(106)-16:POKE 54279,A:POKE 1632
,A:PMBASE=256*A
30 POKE 559,62:POKE 53277,3
49 REM CLEAR OUT PM MEMORY
50 FOR I=PMBASE+1024 TO PMBASE+2048:POKE
I,0:NEXT I
60 POKE 704,216:POKE 705,85:POKE 706,45:
POKE 707,120
65 RESTORE 80
69 REM SET SHAPES OF PLAYERS
70 FOR I=PMBASE+1024+Y TO PMBASE+1028+Y:
READ A:POKE I,A:NEXT I
71 RESTORE 81
75 FOR I=PMBASE+1280+Y TO PMBASE+1288+Y:
READ A:POKE I,A:NEXT I
76 RESTORE 82
77 FOR I=PMBASE+1536+Y TO PMBASE+1542+Y:
READ A:POKE I,A:NEXT I
78 RESTORE 83
79 FOR I=PMBASE+1792+Y TO PMBASE+1798+Y:
READ A:POKE I,A:NEXT I

```

```

80 DATA 153,189,255,189,153
81 DATA 255,255,195,219,195,255,255
82 DATA 16,8,4,255,4,8,16
83 DATA 24,60,126,255,126,60,24
89 REM CALL ASSEMBLY ROUTINE
90 FOR X=1 TO 8000:C=USR(B,1,2,2):C=USR(
B,2,254,2):C=USR(B,3,1,255):C=USR(B,4,25
5,255):NEXT X
100 GOTO 90
110 DATA 104,201,3,208,84,104,104,240,78
,201,5,176,74,160,0,132,203,132,205,172,
96,6,200,132,206,24,168
111 DATA 109,96,6,105,3,133,204,104,104,
24,121,97,6,153,97,6,104,104,168,240,27,
132,208,160,0,177,203,132,207
112 DATA 164,208,145,205,200,132,208,164
,207,200,208,240,177,205,145,203,200,208
,249,160,5,185,97,6
113 DATA 153,255,207,136,208,247,96,169,
2,168,104,104,136,208,251,96,0,0,0,0,0,0 ©

```

## ACCU - WRITE

THE WORD PROCESSOR FOR  
**ATARI 400/800**  
 WITH CASSETTE DRIVE

ACCU-WRITE IS FOR USE WITH THE ATARI 400 800  
 COMPUTER AND THE EPSOM MX-80 PRINTER. FULL  
 DOCUMENTATION ALLOWS USER MODIFICATIONS  
 TO SUPPORT OTHER PRINTERS AND COMPUTERS.  
 THE SUGGESTED RAM IS 32K.

#### ACCU-WRITE FEATURES INCLUDE:

ON SCREEN CORRECTIONS AUTO-CENTER TITLES  
 INSERT, DELETE LINES LINE COUNTER  
 OUTLINE FORMAT, PARAGRAPH, SUBPARAGRAPH  
 FULL DOCUMENTATION & CASSETTE -- \$49.95  
 VISA AND MASTER CHARGE ACCEPTED  
 CALL (504) 361-8594

DREAM PRINT HOMES INC. SUITE 705  
 1700 STUMPF BLVD. GRETN, LA. 70053



# Learning With Computers

Glenn Kleiman and Mary Humphrey  
Teaching Tools: Microcomputer Services  
P.O. Box 50065  
Palo Alto, CA 94303

## Preschool Computing

Several friends of ours recently used computers for the first time. After we helped them get started, they enjoyed themselves and were eager to do more with the computer. They also had many comments, both positive and negative, about the programs they used. This is all very familiar – most people we have introduced to computers have responded in the same way. The difference is that these particular friends are between two-and-a-half and five years old.

Personal computers can be programmed to present lessons and games which encourage children's learning of such things as color names, numbers, letters, vocabulary, and perceptual skills. While using computers in these ways, children also take their first steps towards computer literacy. They realize that computers are an integral part of the world, and they learn about the keyboard, the cursor, disks and other aspects of computer use. We find children to be far more comfortable playing with computers than are most adults.

## Programs For Preschoolers

A number of available programs are suitable for preschool children. Some of these programs are designed for children to use by themselves (once they are helped to start), others are for two or more children to use together, and others are designed for three way interactions among child, adult and computer. The programs all use graphics and minimize the need for reading. Some also make good use of sound. Many of these programs explicitly teach certain things, others are designed for the child to explore and create. The following programs are our young friends' favorites.

*Hodge Podge* (for Apple computers, from Dynacomp, Inc. 1427 Monroe Ave., Rochester, NY 14618). The instructions accompanying this program describe it as a "surreptitious learning" program. When any key is pressed, a song, animation,

or picture related to the key is presented. Press A for an apple, B for a bear, C for a cat, D for a dog and so on. Some keys result in more elaborate displays. Press F and a farm is shown. "Old MacDonald Had a Farm" is played and, at the appropriate time, an animal appears – a different animal each time. Press U and steps appear, a marker moves up the steps while tones going up the scale are played. Press ? and the alphabet song is played, with each letter appearing in turn. The numbers 1 through 8 each play a musical note and show that note on a staff. Zero turns the number keys into a miniature piano so children play their own tunes. Other keys result in displays illustrating addition facts and the concepts of smaller and bigger. This program is packed with easy to use educational features which can entertain a child for some time.

*Above/Below/Left/Right* (for Apple computers, from Advanced Learning Technology, Inc., 4370 Alpine Road, Suite 201, Portola Valley, CA 94025). This set of programs teaches the concepts given in its title. One program is on *above* and *below*, one on *left* and *right*, and one on all four terms. Each program follows a similar sequence. For example, to use the *Above/Below* program a blue bar (provided with the programs) is placed across the keyboard, dividing it into two sections, one for *above* and one for *below*. The program begins by showing a horizontal blue line across the middle of the screen. In the first part of the program the child can press any key. Pressing a key above the blue keyboard divider results in a colored bar appearing above the blue line on the screen. Pressing a key below the keyboard divider results in a colored bar below the blue line on the screen. Tones play each time a key is pressed. In the second part of the program the child is shown an incomplete colored bar either above or below the blue line. Pressing an appropriate key completes the bar. The next two parts of the program are similar, but two boxes are shown on the screen, one above the other. Pressing keys causes either the upper or the lower box to change color. In the final part of the program a colorful picture is created and the colors change in response to the child's key presses. Keys above the divider cause a change in the top half of the picture, keys below the divider cause a change in the bottom half. This program uses colors, music and pretty pictures to hold the child's attention.

*Bumble Games* (for Apple computers, also from Advanced Learning Technology, see address above) is a set of six programs for beginning number skills. The first program is a Guess My Number game, suitable for older preschoolers. A number line appears with the numbers zero to five. The child guesses a number, and is told whether his guess is less than, greater than, or equal to, the



actual number. The child continues until the correct number is guessed. Then a marker on the number line flashes, music plays, a large colored number appears, and the corresponding number of beeps play. The other games involve two dimensional grids and other things more suitable for older children. Like Above/Below/Left/Right, the Bumble Games have some of the prettiest screen displays we have seen.

*Letters and Numbers* (for PET computers, from Teaching Tools: Microcomputer Services, P.O. Box 50065, Palo Alto, 94303). This program provides practice in matching and fill-in drills with letters and numbers. When matching is chosen, large letters or numbers (created with PET graphics) appear on the top of the screen. The child presses the matching letters or numbers on the keyboard. Correct answers result in smile faces, incorrect answers in an X and another try. When fill-in practice is chosen, a sequence of letters or numbers with one missing appears. The child is to type the missing one. Pressing the question mark key provides hints. The first hint is a display of the alphabet or digits, the second hint changes the area of the answer to reverse field. There are a number of options to be set by an adult, such as whether upper case letters, lower case letters, or numbers are displayed, how many practice sets are given, and how many letters or numbers appear in each set.

*Frog!* (for PET computers, from Cursor #19, The Code Works, Box 550, Goleta, CA 93116). This is a playful program which is enjoyed by people from age two on up. We know a three-year-old who calls all computers "Froggy" since he played this program. Froggy captures bugs and makes terrific sounds each time he gets one. The player controls Froggy by pressing keys on the number pad. The 1 to 9 keys form a 3 by 3 square. The higher up on the square you press, the higher up Froggy jumps. The more to the right you press, the longer Froggy's tongue extends. You do not need to know the numbers, just the location of the key. The bugs keep moving (you can set the speed) and you must catch them quickly enough to prevent Froggy from starving. Very young children easily learn which keys make something happen, and can understand how to get Froggy to jump higher.

*Printsit* (for the PET, from Cursor #24, see address above). This program lets the child create pictures. The child can select any of the graphic or alphanumeric symbols on the PET by simply pressing the appropriate key. The symbol can be changed at any time, and even reverse field characters can be used. The symbol is then plotted on the screen using the number pad to control the movement – the direction of movement corresponds to the position of the key on the number pad. If you have

a printer with PET graphic symbols, the entire picture can be printed. Children enjoy creating a picture and being able to change it easily. We were told by one child that it's much better than trying to draw and erase with a pencil. PET graphics make it possible to create many interesting displays, and children especially enjoy getting a printed copy of their work.

*Music!* (for the PET, from Cursor #20, see address above) turns the PET into a one octave toy piano. Older PETs, which do not have built-in speakers, need a CB2 sound add-on to use this program. The child presses a key to play a note, and have it shown on a staff on the screen. The length of the note is determined by how long the key is held down. The child can create a tune, play it, change it, and save it on tape. This program makes it possible for young children to play with music, create their own tunes, and learn something about musical notation, without first learning to play an instrument.

### Some Principles Of Software Design For Young Children

We have seen many programs designed for young children in addition to those described above. In our opinion and the opinions of children we have observed, the programs described are among the best available. What makes them better than the others? There are many important considerations in designing good educational software. Special care is needed in programs for young children since they cannot compensate for a program's shortcomings as well as older people, and can become confused or distracted easily. Six principles of software design we regard as especially important in programs for young children are discussed below. Although the programs we have described follow these principles for the most part, all the programs could be improved. As all software designers agree, there is no such thing as a perfect program.

*Make the program easy to get started.* For example, the Above/Below/Left/Right program starts with a simple four choice menu in which each choice is described by a colorful picture. This makes it easy for young children to select the option they want. The Letters and Numbers program requires answering a series of questions and so must be set up by an adult.

*Make it easy for the child to understand how his actions cause things to happen.* Children learn by realizing the relationship between what they do and the resulting action of the computer. For example, a child using the Hodge Podge program will not learn to associate A with apple until he realizes that an apple appears each time he presses



# HUNTINGTON COMPUTING

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

### Atari®

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

### Pet®

#### AVALON HILL GAME COMPANY

B-1 Nuclear Bomber (cass.)	\$15.00 now	<b>\$12.66</b>
Midway Campaign (cass.)	\$15.00 now	<b>\$12.66</b>
No Atlantic Convoy Raider (cass.)	\$15.00 now	<b>\$12.66</b>
Nukewar (cass.)	\$15.00 now	<b>\$12.66</b>
Conflict 2500 (cass.)	\$15.00 now	<b>\$12.66</b>
Planet Miners (cass.)	\$15.00 now	<b>\$12.66</b>
Computer Acquire (cass.)	\$20.00 now	<b>\$16.96</b>
Lords of Karma (cass.)	\$20.00 now	<b>\$16.96</b>

#### VIC SOFTWARE

UMI Kiddie Checkers	\$7.95 now	<b>\$6.86</b>
UMI Star Wars	\$16.95 now	<b>\$14.86</b>
UMI AMOK	\$18.95 now	<b>\$16.66</b>
UMI Globbler	\$29.95 now	<b>\$21.86</b>

#### ARTWORX

Teacher's Pet (disk)	\$18.95 now	<b>\$16.66</b>
Teacher's Pet (cass.)	\$14.95 now	<b>\$13.06</b>
Vaults of Zurich (disk)	\$25.95 now	<b>\$19.26</b>

#### EPYX

Introductory 3-Pack (disk)	\$49.95 now	<b>\$39.97</b>
(Rescue, Morloc's, and Datestones)		

Rescue at Rigel (cass.)	\$29.95 now	<b>\$25.47</b>
Temple of Aphai (cass.)	\$39.95 now	<b>\$33.36</b>
Heilfire Warrior (cass.)	\$39.95 now	<b>\$33.36</b>
Starfleet Orion (cass.)	\$24.95 now	<b>\$21.16</b>
Invasion Orion (cass.)	\$24.95 now	<b>\$21.16</b>
Morloc's Tower (cass.)	\$19.95 now	<b>\$16.86</b>
Datestones of Ryn (cass.)	\$19.95 now	<b>\$16.97</b>

#### PERSONAL SOFTWARE

Checker King (cass.)	\$19.95 now	<b>\$16.86</b>
Gammon Gambler (cass.)	\$19.95 now	<b>\$16.86</b>
Bridge Partner (cass.)	\$19.95 now	<b>\$16.86</b>
Time Trek (cass.)	\$19.95 now	<b>\$16.86</b>

#### UNITED SOFTWARE OF AMERICA

KRAM (disk)	\$99.95 now	<b>\$84.97</b>
Super KRAM (disk)	\$175.00 now	<b>\$153.96</b>
Request (disk)	\$225.00 now	<b>\$191.27</b>
Thinker (disk)	\$495.00 now	<b>\$420.77</b>
Space Intruders (cass.)	\$19.95 now	<b>\$16.86</b>
All MICRO-ED		<b>10% Off List</b>

#### 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	<b>\$33.99</b>
World Star	\$375.00 now	<b>\$289.00</b>
Mail Merge	\$125.00 now	<b>\$106.19</b>
Super Sort	\$200.00 now	<b>\$169.99</b>
Wurst of Huntington Computing		<b>\$19.99</b>
Nibble Express	\$12.95 now	<b>\$11.99</b>
Soft Porn Adventure	\$29.95 now	<b>\$25.39</b>
Time Lord	\$29.95 now	<b>\$25.39</b>
French Hangman	\$29.95 now	<b>\$25.39</b>
Alicia Sp. bilingual reader	\$29.95 now	<b>\$25.39</b>
H&H Stock Trader	\$190.00 now	<b>\$161.49</b>
Grow (CIA)	\$35.00 now	<b>\$31.49</b>
All Edu-Ware		<b>10% Off List</b>
VersaCalc	\$100.00 now	<b>\$84.99</b>
Hebrew	\$60.00 now	<b>\$50.99</b>
All Serendipity		<b>10% Off List</b>
All Sirius		<b>10% Off List</b>
Win at the Races	\$49.95 now	<b>\$44.89</b>
Disk Prep	\$25.00 now	<b>\$21.19</b>
PLE Chip	\$60.00 now	<b>\$50.99</b>

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.

### TRS-80®

#### BIG FIVE SOFTWARE

Super Nova (cass.)	\$15.95 now	<b>\$13.58</b>
Galaxy Invasion (cass.)	\$15.95 now	<b>\$13.58</b>
Attack Force (cass.)	\$15.95 now	<b>\$13.58</b>
Cosmic Fighter (cass.)	\$15.95 now	<b>\$13.58</b>
Meteor Mission II (cass.)	\$15.95 now	<b>\$13.58</b>

#### ADVENTURE INTERNATIONAL

Adv. 4-5-6 (disk)	\$39.95 now	<b>\$35.08</b>
Adv. 1-2-3 (disk)	\$39.95 now	<b>\$35.08</b>
Adv. 7-8-9 (disk)	\$39.95 now	<b>\$35.08</b>

#### ARTWORX

Teacher's Pet (disk)	\$18.95 now	<b>\$16.58</b>
Nominoes Jigsaw (disk) color	\$21.95 now	<b>\$19.28</b>
Nominoes Jigsaw (cass.) color	\$17.95 now	<b>\$15.78</b>
Bridge 2.0 (cass.)	\$17.95 now	<b>\$15.78</b>
Hearts (disk)	\$19.95 now	<b>\$17.48</b>

#### BRODERBUND SOFTWARE

Galactic Trilogy (disk)	\$39.95 now	<b>\$35.08</b>
Galactic Empire (cass.)	\$14.95 now	<b>\$12.68</b>
Galactic Trader (cass.)	\$14.95 now	<b>\$12.68</b>
Galactic Revolution (cass.)	\$14.95 now	<b>\$12.68</b>
Tawala's Last Redoubt (cass.)	\$19.95 now	<b>\$16.98</b>

#### DATASOFT

Iago (disk)	\$24.95 now	<b>\$21.18</b>
Football Classics (disk)	\$24.95 now	<b>\$21.18</b>
Arcade-80 (disk)	\$24.95 now	<b>\$21.18</b>
Iago (cass.)	\$19.95 now	<b>\$16.98</b>
Football Classics (cass.)	\$19.95 now	<b>\$16.98</b>
Arcade-80 (cass.)	\$19.95 now	<b>\$16.88</b>
Sigmon (COLOR) (cass.)	\$29.95 now	<b>\$25.38</b>
SECS (COLOR) (cass.)	\$29.95 now	<b>\$25.38</b>

#### ACORN SOFTWARE

Invaders From Space (disk)	\$20.95 now	<b>\$17.78</b>
Duel-N-Droids (disk)	\$20.95 now	<b>\$17.78</b>
Pinball (disk)	\$20.95 now	<b>\$17.78</b>
Pigskin (disk)	\$20.95 now	<b>\$17.78</b>
Quad (disk)	\$20.95 now	<b>\$17.78</b>
Basketball (disk)	\$20.95 now	<b>\$17.78</b>
Gammon Challenger (disk)	\$20.95 now	<b>\$17.78</b>
Everest Explorer (disk)	\$20.95 now	<b>\$17.78</b>
SuperScript (disk)	\$29.95 now	<b>\$25.38</b>
System Savers (cass.)	\$14.95 now	<b>\$12.68</b>
Invaders From Space (cass.)	\$14.95 now	<b>\$12.68</b>
Duel-N-Droids (cass.)	\$14.95 now	<b>\$12.68</b>
Pinball (cass.)	\$14.95 now	<b>\$12.68</b>
Pigskin (cass.)	\$14.95 now	<b>\$12.68</b>
Quad (cass.)	\$14.95 now	<b>\$12.68</b>
Basketball (cass.)	\$14.95 now	<b>\$12.68</b>
Gammon Challenger (cass.)	\$14.95 now	<b>\$12.68</b>
Everest Explorer (cass.)	\$14.95 now	<b>\$12.68</b>
All Automated Simulations		<b>10% Off List</b>
All Avalon Hill		<b>10% Off List</b>
All Hayden		<b>10% Off List</b>
All Microsoft		<b>10% Off List</b>

### 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 \$250.00 List  
**Now \$199.00**

Order by Phone

800-344-5109

Calif. 800-692-4146

Foreign

(209) 992-5411

### 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!

### 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



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.



the A key. Programs for young children require great care so the child is able to discern these relationships. One thing we have observed is that young children often press a key and then keep holding it down. On some computers this registers as many key presses. We have seen children do this using the Froggy program. Froggy then jumps many times. Since the child thinks he has pressed a key just once, he does not realize how he caused the action on the screen.

*Make it easy for the child to enter responses.* All the programs described above require just one or two key presses for each response. Some of them also give the child a chance to erase and reenter a response before anything happens. This is useful as young children often press keys they did not intend to press. Input devices such as joysticks, game paddles, and light pens are often easier for children to use than keyboards, but few available programs use these devices.

*Let the child work at his own pace.* Children are very variable in how quickly they respond. Therefore, programs should be paced by the child's responses, not by internal timers (unless, of course, speed of response is an important part of the lesson or game). Programs that move on too quickly become frustrating. Programs that make the child wait become boring.

*Hold children's attention, but do not distract them from the important information.* Preschool children typically have short attention spans and may not be able to tell which information on the screen is most important. Screen displays have to be carefully designed to make them interesting without being confusing or distracting. Color, sound, and movement are very salient for children. They can be used to draw attention to educational aspects of the lessons or games. However, they can easily be misused and distract or confuse the child. For example, in the Above/Below program the lines and boxes that appear vary in color. This creates a pleasing visual display. However, it can also lead to confusion – several children first thought they were controlling the color of the lines, not the location. Another example of distraction is found in the Hodge Podge program. When music plays, the words DO, RE, ME, FA, SO, LA and TI appear in the corner of the screen as each note plays. This distracted some children from the main part of the display, and confused those who could not read the words. Several programs also leave a flashing cursor in the corner of the screen. Many children find this annoying. A prompt symbol that does not flash would better serve the same purpose.

*Make sure the child can understand the feedback.* In order to learn, the child must understand when his answer is correct and when it is not. We have found

many cases of feedback that children can misinterpret. For example, some programs flash the child's name on the screen when a correct answer is entered. We have observed some children become upset by this – they thought they were wrong and the computer was yelling at them. We have even seen programs for preschoolers which use the words correct and wrong, with no other feedback. For pre-readers, things like smiling and frowning faces are much more appropriate.

Some minimal requirements for software design have not been included in our list of principles. We assume all software designers realize such things as programs should not crash when an unexpected key is pressed and feedback should not be insulting to the child. There is no excuse for programs that do not meet these minimal standards. The quality of educational software has increased greatly in the past few years and will continue to do so. Software designers are now placing more emphasis on making programs user-friendly and pedagogically effective. We expect the most exciting new developments in the next few years to be in software, rather than hardware, and look forward to seeing many innovative and well-designed educational programs. ©



## 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



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



- Low Cost Daisy Wheel Printer

**\$845<sup>00</sup>**

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

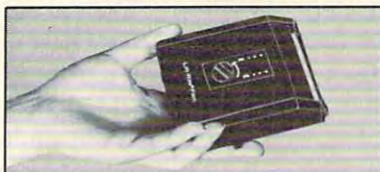
**ACT NOW: Limited Supply, Low, Low Cost**

Smith Corona, one of the largest manufacturers of small printers in the world, gives a whole new perspective to printing with their electronic text printer—TP-1. The TP-1 is a microprocessor controlled, high quality **daisy wheel printer**. It produces perfectly formed, executive quality printouts at the speed of 120 words per minute. Typewriter quality printing at dot matrix prices.

Simple, durable and dependable, TP-1 may be used with word processing systems, microcomputers and most small business systems. Compact and attractively



Additional daisy print wheels . . . \$4.95



Additional ribbons . . . \$2.95

styled, the TP-1 blends well with any setting.

Now, all your letters, documents forms and reports can have the crisp, professional look you demand—for business or personal use—at an affordable price. TP-1, the electronic text printer.

Don't delay. Order your TP-1 TODAY at the low price of **\$845**.

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

**Micro<sup>TM</sup> Printer Marketing**

**Call Micro-Printer Marketing**

CALL  
TOLL FREE

**1-800-523-9859**

MasterCard and Visa Accepted



In PA Call Collect 215 / 433-3366

[www.commodore.ca](http://www.commodore.ca)



## Part I:

# Machine Language: First Steps

Jim Butterfield  
Toronto, Canada

Let's do a simple machine language project from ground zero and try to follow all the steps. We'll pick something very easy, and our coding will be for the CBM/PET. And we'll write it in BASIC first to make our objectives – and the flow of logic – clear.

Programmer F. R. Vescent wants to draw a bar graph of several numbers to the screen. He wants to output the graph as a series of decimal digits so that the viewer can see at a glance the length of a bar (for example, a bar 39 units long will end in a 9 digit). In BASIC, he codes:

```
200 DATA 15,10,30,35,28,28,15,0
210 READ V:IF V=0 GOTO 300
220 J=48:FOR K=1 TO J
230 J=J+1:IF J=58 THEN J=48
240 PRINT CHR$(J);
250 NEXT K
260 PRINT
270 GOTO 210
300 END
```

You may code this and try it in BASIC. It's reasonably fast and a convenient way of representing numbers. But F. R. is a speed demon, and wants to code lines 220 to 260 in machine language. How does he go about it?

A few preliminary decisions: BASIC will pass the value of V (read from the DATA statement) to the program by POKEing it to some convenient place in memory. F. R. chooses hex \$0300, or decimal 768, as the location. No particular reason except that it's not in use.

We'll place the program itself into the first cassette buffer, too. Now to plan out the logic.

F. R. grabs the back of an envelope and starts scribbling. He knows that he has three registers in the 6502 he can use for data: A, X, and Y. He knows that X and Y are especially handy for adding 1, so they seem to be useful for the BASIC variables

J and K. After all, the K loop is stepping by 1, and we have the  $J = J + 1$  calculation on line 230. We'll need to use register A for output.

F. R. looks at the first BASIC command:  $J = 48$ . He writes down `LDX #$30`. Meaning: Load into X the value hexadecimal 30, or decimal 48. X will hold the J-value, you may remember.

Now he looks at the next command: `FOR K = 1 TO J`. He writes, `LDY #$01`, which means, Load Y with value 1. That's where K will start. Now he scrawls himself a note on the next line: "the Y loop comes back here." This part isn't ML coding, it's just a note so that everything can connect up. If he likes abbreviations, he might just note "YLOOP".

There's more to be done to complete the `FOR K` statement, but we'll do it later when the `NEXT K` comes up. On to the next command.

We read  $J = J + 1$ , and F. R. codes `INX`. This means Increment X, and the value in the X register will become one greater. Since X contains the J-value, that's just what we want. Next line: `IF J = 58 ..` calls for a test of the value of X. Let's invert the logic, and read this as [if  $j > 58$  skip the rest of the line]. Same logic, right? And it will make the job easier. Now F. R. codes: `CPX #$3A`, or Compare X to hex 3A, decimal 59. On the next line, he codes `BNE SKIP` as a note to himself ... when he gets to the start of the next line, he'll be able to connect everything up, as the lawyers say. Now for the remainder of the current line:  $J = 48$  becomes `LDX #$30` as before.

Now we've reached point `SKIP` where the code joins up, and F. R. notes this by writing `SKIP` in the left margin. We are ready to `PRINT CHR$(J)`. Now, the value of J is held in the X register; to print it, it must be in the A register. That's easy: our hero codes `TXA`, Transfer X to A, and a copy of the value in X is transferred to A. Once we have it there, we print by calling a subroutine with `JSR $FFD2`. The address `FFD2` hexadecimal contains the start of a subroutine to print the contents of A. When it's done the job, it will return and the program that F. R. is writing will resume where it left off. It's very much like a `GOSUB` in BASIC.

We've arrived at `NEXT K`. We can mentally translate that to  $K = K + 1$ : `IF K <= V .. go back`. Okay,  $K = K + 1$  neatly codes as `INY`, Increment Y. Now we'll need another compare, this time to value V which has been `POKEd` (we hope) to address hex 0300. F. R. writes: `CPY $0300`, Compare Y to address hex 0300, and below it, `BCC YLOOP`. Remember `YLOOP`? That was the note that F. R. wrote to himself quite a bit earlier. `BCC` means Branch Carry Clear: we can read it here as Branch Less Than, since the Branch will take place if Y is less than \$0300. Note that we don't also Branch



Equal as we would like. F. R. scratches his head and makes a note to fix that up later somehow.

All we have left is a bland PRINT statement on line 260. Print what? A Return character, of course, to end the line. F. R. knows that this character is a decimal thirteen or hexadecimal OD, so he codes LDA #\$0D to bring it into the A register and then JSR \$FFD1 to print it as before.

The coding job's done, and F. R. notes down the last instruction: RTS, Return from Subroutine, which will cause the machine language program to return to the BASIC program which called it. He sits back. Then he notices that he's scribbled the whole thing on the back of his subscription renewal to **COMPUTE!** (the magazine gets a lot of programs that way), and decides to make a copy.

This is the program he transcribes onto a stenographer's note pad. The instructions are to the right of the vertical center line; the two notes, YLOOP and SKIP are just to the left. There's plenty of space to the left; he'll be using that later. Here is what his coding looks like:

```
LDX #$30
LDY #$01
YLOOP INX
```

```
CPX #$3A
BNE SKIP
LDX #$30
TXA
JSR $FFD2
INY
CPY $0300
BCC YLOOP
LDA #$0D
JSR $FFD2
RTS
```

You should be able to relate this program to the BASIC program previously given. One important point: where the "#" sign is used – you may call it hash, numbers, or pounds sign – we want the program to use the actual value. Where it is not used, we want the program to use the contents of an address. The "#" sign signals immediate addressing – not really an address at all, but the actual value.

F. R. Vescent's program is not ready to run yet. He has written it in Source (or Assembler) language for his own convenience. The computer can run only Object (or Machine) language. He must translate from Source to Object language by knowing the codes. This translation job is called *assembly*. We'll tackle it next time. ©

## (LABEL), Y (LABEL,X) LABEL + INDX-1

### 6502 Assembler/Editor

- APPLE
- ATARI
- PET
- KIM
- SYM

Before you buy that off-brand Assembler/Text Editor, note that EHS is the only company that provides a line of **compatible** ASM/TED's for the PET/APPLE/ATARI/SYM/KIM and other microcomputers.

When you make the transition from one of these 6502-based microcomputers to another, you no longer have to relearn peculiar Syntax's, pseudo ops, and commands. Not only that, EHS ASM/TED's are the **only resident 6502 Macro Assemblers** available and they have been available for several years. Thus you can be sure **they work**. Our ASM/TED's may cost a little more but do the others provide these **powerful features**: Macros, Conditional Assembly, String Search and Replace, or even up to 31 characters per label? Before you spend your money on that other ASM/TED, write for our free detailed spec sheet.

#### MACRO ASM/TED

- For APPLE/ATARI/PET/SYM/KIM
- Other than our MAE, no other assembler is as powerful.
- Macros/Conditional Assembly.
- Extensive text editing features
- Long Labels
- Designed for Cassette-based systems.

\$49.95

#### MAE ASM/TED

- For APPLE/ATARI/PET
- The most powerful ASM/TED
- Macros/Conditional and Interactive Assembly
- Extensive text editing features
- Long Labels
- Control files
- Designed for Disk-based Systems.

\$169.95



**EASTERN HOUSE SOFTWARE**  
3239 Linda Drive  
Winston-Salem, N. C. 27106  
(Dealer Inquiries Invited)

**PHONE ORDERS**  
USA (919) 924-2889  
(919) 748-8446



.EN .BY .OS .BA .DE .CE



# Telecommunications: Sending Programs Over The Phone

Michael E. Day  
Chief Engineer  
Edge Technology

One use for a modem is to transfer data between your own computer and another one. This mundane aspect of telecommunications can be one of the more interesting and rewarding uses of both the modem and your computer.

The usual method of getting programs into your computer is to either enter them by hand through your keyboard, or to obtain a disk or cassette with the desired program on it. While this is fine if the program desired is readily available, it can make things a bit difficult if it is not. If, for instance, a friend of yours has a BASIC program you would like to have, the usual procedure is for him or her to make a copy of the program on cassette or disk and give it to you. If it is not too large, you might get a printed copy. If you have a cassette and your friend has a disk, the usual response is to not bother. If both of you had a way to transfer the program over the phone, though, you could easily get the program. Another interesting aspect of this is that the program can be transferred to you instantly. With the modem all it takes is a phone call.

In order to make these calls, there must be some agreement as to how you will transfer this information. The actual mechanics can be quite complex. An agreement about how to make the actual transfer is called a *communication format* or *protocol*. Although there are certain basic requirements needed to make the transfer, there is no standard format for the actual details of the transfer.

## Making A Link

The general structural requirements of data transfer are standardized to some extent. This is largely due to the basic requirements of an actual transfer.

The first thing that is done is to establish the communications link. This is done when you call your friend, make the arrangement to do the transfer, and turn on the modems to begin the transfer. Next, the computers must synchronize themselves, and then, finally, make the actual transfer.

The transfer itself is broken into small pieces called *records*. The records generally consist of 128, 256, or 1024 characters (bytes). The actual record size is normally chosen to fit the particular system that the transfer program is running on and, in fact, is one of the reasons for the lack of standardization of these transfer programs.

One popular operating system that is used on some computers is CP/M\*. Because CP/M was originally based on the IBM standard eight inch floppy disk which uses a storage method of 128 bytes per record. So, naturally enough, the record length that is usually chosen for the data transfer through the modem is 128 bytes too. This doesn't mean that the transfer *has* to take place in 128 byte records. But a decision had to be made as to what the record length would be and, since the system stored data in 128 byte groups, this was chosen. It could just as easily be 256 or 1024 bytes.

Actually a 128 byte record is a fairly reasonable size since, at 300 baud, (the number of characters sent per second) takes a little over four seconds to transmit a record. 256 bytes would take over eight seconds, and 1024 bytes would take over 30 seconds. The idea here is to keep the transfer size down so that, if an error does occur, not too much time is wasted retransmitting it. On the other hand, you don't want it broken down into such small pieces that the overhead involved in handling the records significantly retards the transfer time.

Overhead time can be considered the time it takes to acknowledge the receipt of the record. In a simple transfer program this would be a single character. Another part of the overhead that must be considered is the turnaround time of both the computer systems and the phone line. On a local call this generally averages out to about three or four character times (assuming 300 baud). On a long distance call, this can stretch out to eight to twelve character times. (If the call is via satellite, it will be around 40 character times.)

Assuming that the call is local, this means that the overhead would be about five characters. For 128 byte records this would be about 4% overhead. For 256 byte records it would be 2%, and for 1024 byte records it would be 1/2%. This has to be balanced against the expected error rate. The phone line has an average error rate of about one error in every 10,000 bytes of data that is transferred. If the phone line is weak or noisy, it can get much worse. With 1024 byte records, this means that about one out of every ten records will be bad, (a 10% error rate) so the 1/2% transfer rate is lost in the 10% error rate. With 256 byte records, the error rate is down to 2.5%, and with 128 byte records it is 1.25%.

Assuming a 128 byte record format, and ac-



# MICRO



**BUSINESS WORLD INC.**  
Information Line (213) 996-2252  
TOLL FREE MAIL ORDER LINES  
(800) 423-5886 Outside Calif

HEWLETT  
PACKARD



## HP 41CV

OUR PRICE **\$239.00** *Save*  
MSL 325.00 86.00

EXPANSION BOARD  
Q STAR  
16K RAM BOARD



MSL 199.00

OUR PRICE **\$99.00** *Save*  
100.00

## MICRO-SCI

APPLE II+  
COMPATIBLE  
DRIVE



W/ CONTROLLER  
MSL OUR PRICE  
549.00 **429.00**

W/O CONTROLLER  
MSL OUR PRICE  
449.00 **379.00**

*Save* COMPARE TO  
216.00 APPLE DRIVE

APPLE IS A REG. TRADE MARK OF APPLE COMPUTER

## IBM

IBM PERSONAL  
COMPUTER  
INCLUDES:



2 Drives  
Colour Graphic Board  
64K Memory

CALL FOR PRICE

\*Subject to availability

## FRANKLIN 64K

ACE 100

APPLE COMPATIBLE  
INTRODUCTORY OFFER  
CALL FOR PRICE

MSL 1595.00



### Franklin ACE 100 Features

- Apple II compatible
- 64K of RAM memory
- Upper and lower case
- Typewriter-style keyboard
- Twelve key numeric pad
- Alpha lock shift key
- VisiCalc friendly
- 50 watt power supply
- Built-in fan

### COMPARE OUR COMPUTER

FEATURES	APPLE II	FRANKLIN ACE 100
Computer with 48K RAM Memory	\$1530.00	STD
Microsoft 16K RAM Card (16K RAM Memory)	195.00 (option)	STD
ABT 10 Key Pad	125.00 (option)	STD
Vindex Keyboard Enhancer	149.00 (option)	STD
R.H. Electronics Super Fan II (Muffin Fan)	69.00 (option)	STD
TOTAL	\$2068.00	\$1595.00

APPLE IS A REG. TRADE MARK OF APPLE COMPUTER

## LE MONITOR



9" GRN. PHS. MONITOR  
12" GRN. PHS. MONITOR

MSL OUR PRICE *Save*  
189.00 **\$119.00** 70.00

## APPLE II PLUS 48K



\*Subject to availability

CALL FOR PRICE

APPLE IS A REG. TRADE MARK OF APPLE COMPUTER

HEWLETT  
PACKARD

## HP 85

The professional's choice:  
Hewlett-Packard's HP-85



MSL OUR PRICE *Save*  
2750.00 **\$1975.00** 775.00

HEWLETT  
PACKARD

## HP 125



OUR PRICE **\$1975.00** *Save*  
MSL 2750.00 775.00

## DUAL MASTER DRIVE

MSL OUR PRICE *Save*  
2500.00 **1595.00** 905.00

## XEROX 820

WORD PROCESSING SYSTEM  
(INCLUDES SOFTWARE)



MSL OUR PRICE *Save*  
3495.00 **\$2649.00** 846.00

## NEC STUDENT SYSTEM 64K



\*NEC PC 80001  
\*NEC PC 8012  
\*NEC PC 8031  
12" Grn. PHS. Video Monitor

MSL OUR PRICE *Save*  
2839.00 **\$2095.00** 744.00

HEWLETT  
PACKARD

## HP-87



MSL OUR PRICE *Save*  
2495.00 **\$1749.00** 746.00

## Commodore VIC-20



Vic TV Modul ..... \$19.00  
Vic Cassette ..... \$69.00  
Vic Disk Drive ..... \$ Call  
Vic 6 Pack program ..... \$44.00

MSL OUR PRICE *Save*  
299.00 **\$255.00** 44.00

## EPSON



OUR PRICE *Save*  
MSL  
MX 80 645.00 **429.00** 216.00  
MX 80FT 745.00 **535.00** 210.00  
MX 100 995.00 **719.00** 276.00

## Commodore STUDENT SYSTEM



4032 - 32K 40 COL CRT  
4940 - DUAL DISK DRIVE  
MSL OUR PRICE *Save*  
2590.00 **\$1978.00** 612.00

## RADIO SHACK 48K

TRS 80 W/2 DRIVE  
MOD III



MSL OUR PRICE *Save*  
2495.00 **\$1799.00** 696.00

WE RESERVE THE RIGHT TO CORRECT TYPOGRAPHICAL ERRORS. THIS AD SUPERCEDES ALL PREVIOUS ADS.

## 1 Year Extended WARRANTY

**\$99.00**  
INQUIRE



AVINGS  
SERVICE  
ELECTION  
ATISFACTION



**MICRO BUSINESS WORLD**  
MAIL ORDER  
WAREHOUSE  
18720 Oxnard, #108  
Torrance, CA 91356

OUTSIDE CA CALL TOLL FREE 1 (800) 423-5886 IN CA (213) 996-2252

Name (Please print) \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_

Qty Make Model Description Price Total


☐ Certified Check or M.O. ☐ Allow 2 weeks clearance for personal check

☐ Bank Wire Transfer

CREDIT CARD # \_\_\_\_\_

Exp Date \_\_\_\_\_

Telex: 182852  
Answer: MICKO TZNA

\*California residents add 6% sales tax  
\*\*Add 3% Shipping & Handling — Add 3% surcharge for credit cards. Orders cannot be shipped unless accompanied by payment, including shipping, handling and tax where applicable

TOTAL ORDER \$  
TAX IF APPLICABLE  
SHIPPING & HANDLING\*\*  
TOTAL ENCLOSED \$





counting for one error in a transfer of 10,240 bytes, the transfer time would be about six minutes. If the records were 256 bytes long, the transfer time would be about 5.9 minutes, and with 1024 byte records it would be about 6.5 minutes. It would seem that the 256 byte format would be the best choice, but another factor must be taken into account: the error detection method used.

In the method used for the CP/M system, it is very simple, and the more bytes it is required to check, the greater the chance is that it will miss an error. Because of this, the 128 byte format is a better choice even though there is a small increase in the transfer time. If a better error detection method were to be used, it would probably be better to use the 256 byte record format.

### A Transfer Format

The actual structure of the record that is transferred varies from system to system as well. In fact, it is even less standardized than most other parts of the data transfer format.

Since there is no real standard for the record format, I will describe one of the more heavily used formats. This format got its start on CP/M-based systems and originally appeared in a program written by Ward Christensen called, appropriately enough, "MODEM." The first problem when dealing with CP/M is its refusal to acknowledge the existence of a modem. So a transfer program must provide its own link to the modem.

To begin the transfer, the receiving computer sends an ASCII NAK (15H) signal every couple of seconds until the sending computer sends an ASCII ACK (06H). This is the synchronization part of the transfer. The original modem program assumed that the program was predefined at both ends, so, once synchronization was achieved, the data was immediately sent.

The record format that is used consists of a *header*, the data, and finally a checksum character for error detection. The header consists of an ASCII SOH character (01H) followed by the current record number (starting with number one) which is an eight bit value. That is followed by the same number, but inverted. (That is, if record 01H is being sent, then the second number sent will be FEH.) This is then followed by the data itself for the next 128 bytes. Finally, one more character is sent which is the checksum.

The checksum is an eight bit value that is the sum (without carry) of all the data bytes sent. The sending computer then waits for the receiving computer to acknowledge that it received the data. The receiving computer compares its own calculated checksum against the one that the one that the sending computer sent and, if they match, it

sends an ASCII ACK character (06H). If they don't match, it sends an ASCII NAK character (15H), indicating that it didn't receive the data correctly. If the sending computer receives a NAK, it will send the record again. If, after ten tries it is unable to send the record it gives up and aborts the transfer. After all of the records have been sent, a final ASCII EOT (04H) is sent indicating that the transmission is completed.

### Some Problems

There are several problems with the format that is used and some of the later versions attempted to correct for this. Unfortunately, this created a new problem since any change in the basic format meant it was incompatible with the old format. This tended to lead towards a real mess with patches to allow for compatibility to the old programs. Discounting the versions which were simply adaptations for different modems, some of the differences that have occurred are the addition of the program identifier so that the sending computer can tell the receiving computer the program name instead of requiring the operator at the receiving computer to specify it. There was also change from the checksum format to a CRC format. The identifier has been implemented several ways, but the most popular version is also one of the strangest implementations.

After synchronization has been achieved, the currently popular program (MODEM7) sends the filename a character at a time. That is, it sends a character and then waits for an acknowledge (ASCII ACK) from the receiving computer. Then it sends the next character of the filename and repeats this until the entire filename has been sent. After that, it waits for the receiving computer to send the checksum of the filename and then compares the received checksum against its own internally calculated checksum. If they are equal, the sending computer sends an ASCII ACK character. The sending computer goes back and waits for resynchronization. (Waiting for an ASCII NAK character.) At last, it starts receiving data normally after the resynchronization is achieved.

If there was a checksum error, the sending computer sends a bad name character which, for no particular reason, was defined as an ASCII *u* (75H) and goes back to allow resynchronization and retransmission of the name. If, after ten tries the name cannot be sent, the transfer is aborted.

### Improved Error Detection

Another problem that some versions have corrected for is that the checksum method of error detection is not the most accurate means of detecting an error. A CRC (cyclic redundancy check) is a far



CBM Maintenance  
In-House

DOWNEY, CA  
213-923-9361/714-778-5455

ALL UNDER ONE ROOF  
31  
years

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

DES ANNOUNCES

# "VIC-VILLE"® commodore

— ONE STOP VIC 20 CENTER —

\*\* SOFTWARE \*\* HARDWARE \*\* EXPANSIONS \*\* PERIPHERALS \*\* ACCESSORIES \*\*

**GAMES:** From our professional programmers (Robert Winter - Ralph Orton - Dan Haste - Robert Burnett

**Goldbrick** \$12.95  
Many levels of play, sound  
and color.

**A Maze Ing** \$10.95  
Travel through the maze  
game of skill and tense  
action.

**Gobbler** \$9.95  
Sound Easy? You have  
25 seconds to get him and  
the time gets shorter at  
each higher level.

**Hang-U** \$10.95  
Traditional Hangman plays  
against the VIC's 250 word  
dictionary OR another  
person.

**Coggle** \$9.95  
Computerized version of  
Boggle. Great for parties.  
Keeps score for 4 people.

**Baseball Strategy**  
\$10.95  
The excitement of baseball  
as a video strategic game.  
Fun for the entire family.

**Attack on Silo III**  
\$10.95  
You are the commander of  
Silo III. Defend your base.

**Yahtzee** \$10.95  
Solitaire version of this  
famous dice game, good  
graphics & lots of fun.

**Boxing** \$9.95  
Two player boxing for the  
VIC.

**AIR STRIKE** \$9.95  
Fly the new super bomber  
V-20 on a mission.

**Pedestrian Polo** \$12.95  
Drive the car through the  
streets of America

**Simple Inventory Control**  
\$49.95  
LIFO System works with  
5K VIC to 32K VIC.  
Complete documentation.

**ANNOUNCING!!! DES 43K Expansion Board . . . . . \$295.00**

**Send for our pricing lists on all supplies and peripherals.**

**ASK FOR DEALER DISCOUNT !!**

## **DES SUPPLY DIVISION**

### **ATTENTION Software Houses**

- ✓ Best prices on bulk 5¼" floppy discs
- ✓ RIBBONS - All types
- ✓ MAILING LABELS - All sizes
- ✓ PRINTER PAPER - All sizes
- ✓ BASF, Memorex, Dysan, Verbatim, Scotch 3M, "DATA-DISC"

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

**PAYMENT** (add \$3 shipping and handling)

☐ CHECK # \_\_\_\_\_  
☐ VISA  
☐ MASTERCHARGE Exp. Date \_\_\_\_\_  
Acct. # \_\_\_\_\_

Name \_\_\_\_\_  
Address \_\_\_\_\_  
City \_\_\_\_\_  
State \_\_\_\_\_ Zip \_\_\_\_\_

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

**Built on a firm foundation!**

 e.ca

VIC 20 - VIC 1515 PRINTERS - VIC 1540 DISK DRIVES

VIC 20 - VIC 1515 PRINTERS - VIC 1540 DISK DRIVES



better way of detecting errors. The most common CRC that is used is a 16 bit polynomial, defined as  $(X + 1) * (X^{15} + X + 1)$ . By starting with a value of zero in the CRC and passing all the received data through the CRC routine, when the final CRC received passes through the routine, the final result will be 0 (if no errors were encountered).

There is no single best way to transfer programs via a modem, but there are some methods which are better than others.

*\*CP/M is a trademark of Digital Research*

### Computing A Checksum

Checksum generates a sum of the data that is passed through it by adding (without any carry) each byte with the sum of the previous bytes. The checksum is initially set to zero and the final result is sent to be compared against the independently computed checksum at the receiver computer. Since the checksum is an eight bit value, only a single byte of data needs to be sent. It is quick and easy to perform this with a computer. The checksum method can only reliably catch single bit errors. Although it does do reasonably well with multibit errors, the percentages can rapidly drop to the realm of coin toss odds.

To use the checksum program enter it with the data in the accumulator. The result is saved in location CHKSUM for later use. The location CHKSUM should be cleared to zero at the start of sending the data stream.

8080 VERSION		6502 VERSION	
CHKSUM	PUSH PSW	CHKSUM	PHA
	PUSH B		PHP
	MOV C,A		CLC
	LDA CHKSUM		
	ADD C	ADC	CHKSUM
	STA CHKSUM	STA	CHKSUM
	POP B		PLP
	POP PSW		PLA
	RET		RTS

This shorthand version for the 8080 assumes that a register in the CPU does not get destroyed by the calling program, and that it is acceptable for the flags to be destroyed.

8080 VERSION

```
CHKSUM ADD C
      MOV C,A
      RET
```

### The CRC Method

CRC computes CRC-16 crcsum from the polynomial  $(X + 1) * X^{15} + X + 1$

A CRC method of error detection is far superior to the checksum. By using a 16 bit value for the sum instead of an eight bit value, a much improved detection capability is achieved. By using a polynomial of the proper type, the 16 bit value can be used far more effectively as well. The provided polynomial can detect errors of up to 17 bits.

Since the CRCSUM generated is a division remainder, a CRCSUMed data sequence can be verified by running the data through the CRC, and then running the previously obtained CRCSUM through the CRC. The resultant CRCSUM should be zero. When the CRCSUM itself is transmitted, it should not be run through the CRC as this would disrupt the result. Also, the MSB {H} must be run through the CRC first then followed by the LSB {L} when checking the CRCSUM.

To use this routine, enter with the byte to be CRCSUMed in A (accumulator). The CRCSUM is automatically updated upon passing the data through this routine.

8080 VERSION		6502 VERSION	
CRC	PUSH PSW	CRC	PHP
	PUSH B		PHA
	PUSH H		STX XTEMP
	MVI B,8		LDX #08H
	LHLD CRCSUM	CRC1	ASL A
CRC1	RCL		ADC #00H
	MOV C,A		STA CRCTMP
	MOV A,L		LDA CRCSUM
	ADD A		ASL A
	MOV L,A		STA CRCSUM
	MOV A,H		LDA CRCSUM+1
	RAL		ROL A
	MOV H,A		STA CRCSUM+1
	RAL		ROL A
	XRA C		EOR CRCTMP
	RRC		LSR A
	JNC CRC2		BCC CRC2
	MOV A,H		LDA CRCSUM+1
	XRI 80H		EOR #80H
	MOV H,A		STA CRCSUM+1
	MOV A,L		LDA CRCSUM
	XRI 05H		EOR #05H
	MOV L,A		STA CRCSUM
CRC2	MOV A,C	CRC2	LDA CRCTMP
	DCR B		DEX
	JNZ CRC1		BNE CRC1
	SHLD CRCSUM		
	POP H		LDX XTEMP
	POP B		PLA
	POP PSW		PLP
	RET		RTS
	END		END



## 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 lazars 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

# 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 on:

**EXECOM CORP.**

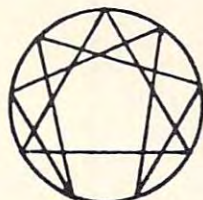
1901 Polaris Ave.  
Racine, WI 53404  
Ph. 414-632-1004

\*Plus installation charge of \$75.00

Available only for Basic 3.0 & Basic 4.0

PET & CBM™

trademark of Commodore Business Machines



## ALTERNATE REALITY SOFTWARE

Presents for the Atari

### THE I CHING

#### THE ANCIENT CHINESE BOOK OF DIVINATION

- The complete text of the world's oldest book on disk
- 40K program
- 73 disk files (155,000 bytes)
- Occupying 1211 disk sectors
- High Resolution Graphics
- Animation
- Music and Sound Effects
- Instructional text material
- Calculates and Displays Hexagrams
- Displays Judgement, Image, Moving Lines for primary & secondary Hexagrams
- \$44.95

order from:

Alternate Reality Software  
2111 W. Arapahoe Drive  
Littleton, Colorado 80120

Dealer inquiries invited  
Atari is a T.M. of Atari, Inc.

## 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; Intelligent Terminal Software: \$79.95

### MX-200 IEEE-488/PET PARITY MODEM/SOFTWARE

Talk to a host computer requiring parity, all features of SX232: \$399.95

### MX-113 THEFT PROTECTION ROM:

Plug in ROM, displays owner's name, etc. when computer turned on: \$49.95

## ECX COMPUTER COMPANY

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

For additional new product information and catalog send self addressed, stamped envelope.



Table 1. Send A 256 Byte Program

Sending Computer	Receiving Computer
— Synchronization Time —	
No Response	NAK
ACK	NAK
— Synchronization Achieved —	
— Send A Record —	
SOH	
Record #1	
Record #1 (Inverted)	
Data	
128 Bytes	
Checksum	
	ACK
SOH	
Record #2	
Record #2 (Inverted)	
Data	
Checksum	
	NAK
SOH	
Record #2	
Record #2 (Inverted)	
DATA	
Checksum	
	ACK
— All Done, Stop Sending —	
EOT	
	ACK
— Transmission Done —	©

## COMPUTE!

The Resource

### 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, ANADIX, 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

### PET/CBM OWNERS

## WALLBANGER™

An original arcade-style game which combines high speed action and strategy. Blast your way through the dodge'm, blast'm, and attack modes. Outsmart the bouncing balls and the walls close in for the next round. Wallbanger™ is written in machine language, has great sound, and has 4 skill levels.

CASS/8K/40 COL SCREEN/OLD-NEW ROMS ..... **\$15.00**  
[CALIF. RES. ADD 6% SALES TAX]

Write for FREE game details:

**ON LINE SOFTWARE**

**P.O. BOX 2044**

**ORCUTT, CA 93455**

**W A R N I N G !**

**WALLBANGER™ causes high panic levels.**

**PET/CBM OWNERS**

## CASDUP

This program allows you to make backup copies of those expensive machine language programs you bought.

Single or multi-file  
copies on any  
ATARI 400/800

Cassette only \$20.00  
(California residents add  
6% tax)

Send check or money  
order to:

**VERVAN Software**  
10072 Balsa Street  
Cucamonga, Calif. 91730



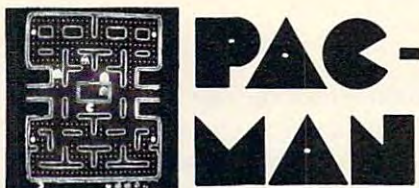
**CALL TOLL FREE  
800 424-2738**



## Save \$\$ on Popular Programs for Atari

LIMITED TIME OFFER

Now, take advantage of our price reduction on these popular programs!



From Atari

The World's most popular arcade game now comes to your home in a ROM cartridge! You'll quickly pay for this one with just the quarters you save. All the little munchers are there to be chased by your Pacman. Use your joystick to run the maze, eating dots and avoiding -- or chasing -- your pursuers.

ROM cartridge...\$44.95 **Save 20%!**

NOW THRU JUNE 15 ONLY: pay just \$35.96

## the PROGRAM STORE



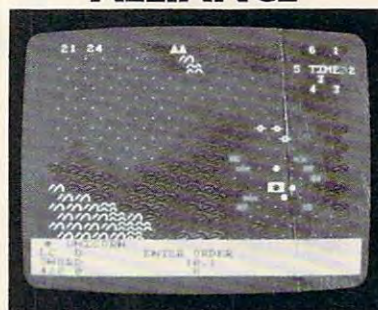
From Avalon Hill

A game of enormous proportions, GALAXY is designed to be played by 1 to as many as 20 players! For each game, the computer sets up a different galactic map of players' planets and neutral planets. Each player's job is to explore the galaxy, find planets with industrial capacity, conquer and colonize them. The problem is: they may not want to be colonized! In this case a battle ensues, complete with sound effects.

16K tape...\$20.00 32K disk...\$25.00

NOW THRU JUNE 15 ONLY: pay just **Save 20%!**  
\$16.00 (T) \$20.00 (D)

## SHATTERED ALLIANCE



By John Lyon from Strategic Simulations

The author of COMPUTER BISMARCK, NAPOLEONICS and TORPEDO FIRE leaves Earth and its wars behind as he takes you to the fantasy planet Osgorth. With excerpts from the "Chronicles of Osgorth," you can conjure up an almost endless number of scenarios. These battle simulations take place on a colorful Hi-Res map, along with rosters that show what resources and spells you have available.

48K disk...\$39.95 **Save 20%!**

NOW THRU JUNE 15 ONLY: pay just 31.96



## DARTS

From Creative Computing

The traditional pubroom game of darts, depicted in strikingly beautiful graphics and sound. Use your joystick to position the thrower's hand -- 10 skill levels allow for all ages of players. Once thrown, the dart either finds its mark or bounces off the wire separators (with a realistic "clink" sound). DARTS allows from one to four players. The disk version includes the ball maze game, TILT, for one or two players.

DARTS, 16K tape...\$14.95  
DARTS/TILT, 16K disk...\$24.95



## LOST COLONY

By David Feitelberg from Acorn

It's the world's first deep space colony and you are the economic manager. A remarkable simulation, LOST COLONY lets you communicate with the computer using full sentences or short commands -- just like an adventure. It arms you with maps and charts as tools for resource management. You assign human and robotic labor, explore new land, and set production quotas. A challenging game, it might give you insight into real life management as well.

16K protected tape or  
32K protected disk...\$19.95 each.

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

## Ali Baba and the forty thieves

By Stuart Smith from Quality

Music and Hi-Res graphics abound in this fantasy/adventure for one or more players. Guide up to 17 "friendlies" through the many rooms of Ali Baba's mountain stronghold in search of the princess. You may find treasure and magic; you're sure to find danger!

32K disk...\$32.95



## ACTION QUEST

From JV Software

You are the ghost of the mansion in this arcade adventure. Using your joystick, you move through the rooms in quest of prizes. You must elude monsters and solve puzzles along the way. Dexterity and reasoning skills are required to be successful in this fast-moving game.

16K tape...\$29.95 16K disk...\$29.95

## PROTECTOR

From Synapse Software

The planet's inhabitants are endangered by a malevolent alien that beams them to his ship and transports them to an active volcano. You must pick them up one-by-one with your rescue/attack ship and transport them to another city while dodging lasers and rough terrain. After you save as many as possible, the volcano erupts. You must then move each person to a volcano-proof vault in the mountains! Great graphics and sound in this arcade game.

16K tape...\$24.95 32K disk...\$27.50

## ANDROMEDA

By Jaime Cummins from Gebelli

You are Andromeda, and you have entered the body of a multicellular organism. Flitting about the vascular system and fat layers, you must eat cells to stay alive and grow in size and strength. If you can achieve an enlarged state you can also eat the various antibodies, but otherwise -- watch out! Select from 7 levels of antibody protection as you play this science fiction arcade game.

24K disk...\$34.95

## ABUSE!

From Don't Ask

Computer manufacturers take great pride in developing "friendly" computers. ABUSE takes equal pride in nullifying their efforts! The Don Rickles of computer games, it's a battle of wits as you try to outrank this program. It hurls insults that will curl your toes then waits for your reply. When the air clears, who will emerge victorious?

40K disk...\$19.95

## DE RE ATARI



From APX

Translated from Latin, the title of this book is "All About Atari" and it means what it says! Used in combination with Atari's Technical Reference Manual, advanced programmers will be able to learn to exploit the many hardware and operating system features that make the Atari 400/800 so tremendously versatile. Includes a useful discussion of the new GTIA chip. Once you know Atari BASIC and assembler, this book is a must.

Looseleaf (binder not supplied)...\$19.95

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



**TO ORDER CALL TOLL FREE 800 424-2738**

For information  
Call (202) 363-9797

MAIL ORDERS: Send check or M.O. for total purchase price, plus \$1.00 postage & handling. D.C. residents, add 6% tax. Charge card customers: include all embossed information on card.

**THE PROGRAM STORE**  
4200 Wisconsin Avenue NW, Dept. J205 Box 9609  
Washington, D.C. 20016

©1982, The Program Store, Inc.

[www.commodore.ca](http://www.commodore.ca)



*FORTH is a programming language which falls in between BASIC and machine language – in speed, and difficulty. **COMPUTE!** will be covering FORTH on a regular basis and, to start things off, we have Jim Butterfield's introduction to this increasingly popular language.*

# And So FORTH

Jim Butterfield  
Toronto, Canada

Suppose you were offered a collection of pre-tested machine language subroutines which could do many common programming jobs. Nice? Of course. And suppose it were pointed out to you that if there was anything missing from the collection, you could write your own, mostly by calling sequences of the pre-written subroutines. In fact, you could build your own library.

With such a package, you'd have the speed of machine language and the ease of programming that comes with calling up prewritten code. The best of all possible worlds: and that's more or less what you get with FORTH.

FORTH isn't exactly a language. It's a set of useful routines organized in such a way that you can expand the collection with your own items, building on what's previously been written. You don't exactly program; you build larger and larger modules out of the smaller pieces, and what results is in machine language (more or less). The code doesn't need to be interpreted or compiled; each program module is ready to run as soon as you have defined it. And many FORTH writers do just that: they check out each module as it is written.

## How It Works

FORTH was originally written for the PDP-11 computer. This computer has an unusual addressing mode which allows subroutines to be called indirectly. All you need is a list of the subroutine addresses – no instructions or op codes – and you can arrange to execute each subroutine in the list in turn. This type of organization is called *threaded code*. It's economical of memory: what could be briefer than just the address of the subroutine?

In most microprocessors, threaded code is achieved with an "inner interpreter" that picks out each address from the list and then sets up the subroutine call. There's a small penalty in running time for doing this extra chore, but it's not great.

FORTH allows each routine to have a name. Type in the name, and the routine runs. Type in several names, separated by spaces, and each routine runs one after the other. If I type "1 2 + ." the computer will perform the routine called "1",

placing the value one onto the stack; then routine "2", placing the value 2 above the 1 on the stack; then "+" which takes the two values from the stack, adds them, and returns the result to the stack; and finally "." which takes the value from the stack and prints it. Result: the total, 3, is printed.

You may define your own routine. The colon is used to start a definition line: everything following it (up to the ending semicolon) is compiled into the appropriate sequence of subroutine calls. So we may type ": 1 + 2 1 2 + . ;" and define a new routine called "1 + 2" which will add 1 to 2 and print the result. This new item becomes a permanent part of FORTH at least until we power down; and can become truly permanent if we save the new FORTH to storage.

This is one of the problems in trying to pin down FORTH as a language. Which is the true FORTH – the one you brought home from the store, or the one you are now using which includes handy routines that you have subsequently devised? It becomes very hard to say "FORTH can't do this..."; if you do, scores of outraged users will reply, "Yes it can; I added it," or, "I bought a version with extensions that do that..."

There's little point in defining a routine which calculates a fixed value of three; but we can call on the whole FORTH vocabulary to build ever larger commands. The existing vocabulary provides input, output, calculate, testing, and loops. With these, you can construct almost any logical combination. If there's anything that's not there, you can usually put a built-in assembler to work and do the job in machine language.

## The FORTH Estate

There's a standard, more or less, for FORTH. It's defined by FIG-FORTH. But what you are likely to obtain from a supplier – at prices ranging from free to several hundred dollars – may have a few pieces missing (like the assembler) and is likely to have a number of pet extensions and support features, such as editors or file handlers.

One aspect of FORTH can give difficulty to beginners. The language is based on the use of a stack (two of them, in fact), and data is manipulated



 [www.commodore.ca](http://www.commodore.ca)



by pulling and pushing it to and from the data stack. When a command is given to manipulate data, that data had better be on the stack already. To add 1 and 2, we must say `1 2 +` so as to place the two numbers on the stack first. This type of notation is called Reverse Polish (or Postfix) notation and seems clumsy at first; but a little practice will make it seem quite natural.

Values in the data stack are usually limited to 16 bit signed integers. That means that numbers can be integers in the range of -32768 to +32767. You can't fit the value of pi into an integer, nor can you represent your annual salary in pennies unless you're unusually poor. And what about arrays? You might have trouble fitting marks for 300 students on the stack. And, if you made it somehow, you'd have a devil of a time getting the particular one you wanted. And we haven't even mentioned how to deal with strings...

Because of such limitations, FORTH quickly leaves its simple form and starts to demand considerable skills of the programmer. He must be able to allocate memory space and set up sets of indirect pointers that will steer him to the particular unit of data he needs. He will need to be able to handle floating point by building special commands; in many cases this feature is at least partly provided by the vendor.

The beginner is faced with a huge vocabulary of commands, most of which he will need to learn. Not everyone will have the patience to slug through this in order to develop competence in FORTH. When he finds he needs to handle indirect pointers and tables, he'll need to have an aptitude for this kind of thing. FORTH demonstrations can be misleading; the language seems to be so easy when a few simple things are shown.

For those who take the time and trouble to develop FORTH competence, the payoff can be high: fast-running code that can be written quickly. But the beginner must realize that it's not all easy sailing; FORTH won't help you along in the same way that BASIC does.

Advocates of structured programming tend to be suspicious of FORTH. Since FORTH encourages to build upward from the detailed code to the total job, it is considered a "bottom up" type of language. Many computer scientists would prefer to see you go the other way: from the top – the big picture – down into increasing levels of details, or "top down" programming.

It's a language that excites many users. For others, it may be tough sledding and too far off the mainstream of small computer activities. Those who are hooked on FORTH become fanatics: they insist that a job is well done only if it's FORTH right. ©

*Part I of this three-part machine language monitor for the OSI Superboard appeared in March, 1982, issue #22.*

## Part II:

# A Superboard II Monitor

Frank Cohen  
Pacific Palisades, CA

In the March issue of **COMPUTE!** we presented the first part of a fairly complex program to add a sophisticated "monitor" program to the Superboard II. A monitor does nothing more than to peek into the machine's memory and enter, display, move, or store data in the form of hexadecimal bytes.

Stored in the ROM memory on the Superboard is OSI's monitor program. When originally designing the Superboard, a microcomputer called the KIM-1 was selling well in the microcomputer market. The OSI monitor largely resembles the monitor for the KIM-1. The KIM-1 had a six digit display and a hexadecimal (base 16) keypad plus some other keys which had specific functions devoted to each. With the six digit display, there was room to display a two byte address and the contents of that memory location.

There were two modes of operation: the address mode, and data mode. In the address mode, a key pressed was rotated into the current address being displayed. By rotating the key in, the existing address digits are all shifted left one position (the left-most digit was lost) and the new key pressed is put into the right-most digit. The same kind of scheme is used for entering data in the data mode. However, instead of changing the address digits, the contents of that location are changed.

Changing from the data to the address mode (or vice versa) is accomplished by pressing the AD key, or the DA key. The Superboard II uses the period ( . ) key instead to enter the address mode and the comma ( , ) key for the data mode. This system works well for the KIM-1 considering that it cost about \$175 and did not have a video display or an advanced keyboard as the Superboard does.

Of course, the monitor program for the Superboard only occupies a small fraction of the space that Super-Monitor uses. However, if you start using your Superboard more and more, you normally will learn how to program in machine language. Possibly blocking your move into the



wonderful world of machine language is the resident monitor program.

Last month we outlined what the capabilities of the Superboard II's new monitor program should include. Top on the list was the ability to look into memory, a group of locations at a time. Second, we wanted to be able to modify the Superboard's memory and, at the same time, see what we just modified. Third, we want to fill a block of memory with some value. Next, we want to be able to move a whole block of memory from one location to another. Finally, we'll need an intelligent cassette interface routine for storing and retrieving blocks of memory.

Since Super-Monitor is over 500 bytes long, it has been split into sections. Last month's issue presented the listing for a program called HEXDUMP. HEXDUMP was listed first since most of the other routines in Super-Monitor use its subroutines. When looking at the listings of the individual programs, you will find that they are each mini-programs. The start of each listing also tells what other programs (subroutines) are needed to make it work. The logic behind the listing's structure lies in the fact that loading Super-Monitor in its entirety takes about five minutes with the Superboard's slow cassette interface. By loading just the routines that you want, Super-Monitor can be customized.

HEXDUMP fills the screen of the Superboard with data from memory, eight bytes at a time. HEXDUMP, like most of the routines in Super-Monitor, uses a program called Super-Cursor V1.3 (**COMPUTE!** December, 1981, #19, pp. 124-128) to handle its video output. To use Super-Cursor V1.3, a program puts the ASCII character in the CPU's accumulator and executes a jump-to-subroutine (JSR) to the start address of Super-Cursor, 1E40 (Hex). Super-Cursor also is used to clear the screen, address 1EC2 (Hex), and to home the cursor, address 1E80 (Hex). If you don't want to use Super-Cursor, you will have to write your own video output routine. If you want Super-Cursor and Super Monitor you can send a blank cassette and \$3 to the address below and I will copy it for you.

The main subroutines from HEXDUMP that the other routines use are called INADR and PLINE. INADR, starting at address 1D93 (Hex), inputs a two byte address from the keyboard and echoes it to the video screen. The resulting address is stored in address 00E7, called ADR, and 00E8. PLINE is used to print a row of eight bytes of data on the screen. The beginning address is located in ADR, 00E7 and 00E8.

## INDATA

The first program in this issue is called INDATA.

This program is approximately 199 bytes long and allows the user to look into, and modify, any group of memory locations. Entering machine language programs is simple using INDATA. In fact, after writing HEXDUMP, and Super-Cursor V1.3, I used INDATA to enter the other routines. It is fast and efficient.

INDATA shows the programmer a line of eight bytes of data at a time. Preceding the data is the address of the left-most byte of data. A greater-than sign (>) is placed next to the currently "open" memory location. Any hexadecimal key you hit will be rotated into that byte. When you have finished changing the contents of the current memory location, you can move the greater-than sign to the next location (one space right) by pressing the SPACE bar. Or, you can go back to the last location (one space left) by pressing the RUB-OUT key. If you think that you made a mistake just look up at the screen and compare.

If you are at the right-most byte on a row, the next time you hit the space bar the next line of eight bytes will be displayed. The opposite is true for typing a Rub-Out when you are at the left-most byte. When you are finished entering data, pressing the RETURN key will exit the program. In the listing, when you press the RETURN key, the program will go back into OSI's ROM monitor program.

Program 1 is a complete assembly listing of INDATA. As it is listed, it fits right under HEXDUMP on an 8K Superboard II. I do not suggest trying to move INDATA to another part of memory as it uses many absolute addresses which would have to be modified. However, if you don't have an assembler, it is possible to move it. (This is your encouragement to get a more complex system.) If your Superboard has only the original 4K bytes of RAM, I suggest you add some 2114's.

## BMOVE

BMOVE is short for Block Move Routine. As the name implies, this routine is set up to move any size block of memory from one location to another. This is especially handy if you have entered a long program and found that you accidentally started at the wrong location. Another application is looking into the ROM's on your Superboard. By telling BMOVE to move the beginning of the BASIC-ROM, located at A000, to the memory mapped video area you can see the internal organs of BASIC.

To use BMOVE, you enter the program at location 1BC6 (Hex). The program first asks you for the starting location of the block to be moved by printing "S=" on the screen. Then it asks you for the ending address by printing "E=" on the screen. (No, it is not asking for Einstein's Theory of Relativity.) Finally, BMOVE prompts you to enter



the beginning destination address by printing "D=."

BMOVE is very fast. You will find that it can move a block 8K long in about a second. The majority of BMOVE's program listing is devoted to inputting the three addresses. After it has those addresses loaded, BMOVE calculates the last address of the destination. It then proceeds to move the block, byte by byte, from the top down. For every byte it moves, it will decrement the ending address and check to see if it is equal to the starting address. When the two are equal, it will return to OSI's ROM monitor. Again, later, we will modify the program to return to Super-Monitor's main menu routine.

In the third and final installment, next month, the listings will be described and listed. So far we have enough to call this an advanced monitor routine. The three programs, HEXDUMP, INDATA, and BMOVE, allows you to look at, modify, and move, data in very simple steps.

*These routines make extensive calls to HEXDUMP and SUPER-CURSOR V1.3. It also changes SUPER-CURSOR "system variables," such as cursor position. If you want to use INDATA and BMOVE without HEXDUMP and SUPER-CURSOR, you will need to refer to the listings of SUPER-CURSOR (COMPUTE! February, 1982, #21) and HEXDUMP (COMPUTE! December, 1982, #14). Zero page usage:\$E7-\$ED*

#### Program 1: INDATA

```
1C56 20 80 1E A9 41 20 40 1E
1C5E A9 3D 20 40 1E 20 96 1D
1C66 A9 00 85 EA A9 3E 8D 61
1C6E 1F 20 80 1E 20 00 1E AE
1C76 CC D0 20 80 1E 86 E2 20
1C7E FB 1E A6 EA 20 FB 1E 20
1C86 FB 1E 20 FB 1E E0 00 F0
1C8E 04 CA 4C 82 1C A5 E7 38
1C96 E9 08 85 E7 B0 02 C6 E8
1C9E A4 EA B1 E7 85 E9 20 BA
1CA6 FF C9 0D D0 08 A9 A0 8D
1CAE 61 1F 4C 43 FE C9 20 D0
1CB6 03 4C D8 1C C9 7F D0 03
1CBE 4C F8 1C 20 F3 1D 8D CF
1CC6 1C A5 E9 0A 0A 0A 0A 18
1CCE 69 00 85 E9 20 15 1D 4C
1CD6 6F 1C 20 15 1D A5 EA C9
1CDE 07 D0 12 A9 00 85 EA A5
1CE6 E7 18 69 07 85 E7 90 02
1CEE E6 E8 4C 6F 1C E6 EA 4C
1CF6 6F 1C 20 15 1D 98 D0 12
1CFE A9 07 85 EA A5 E7 38 E9
1D06 08 85 E7 B0 02 C6 E8 4C
```

```
1D0E 6F 1C C6 EA 4C 6F 1C A4
1D16 EA A5 E9 91 E7 60 AA AA
```

#### Common routines:

1C56	INDATA	Entry point for INDATA program
1C66	BLOOP	Main loop start for INDATA
1C6F	BPCS	Print a line and fix SUPER-CURSOR bug
1C80	SKIP	Positions cursor to current open cell
1C93	CKSP	Fix HEXDUMP bug by adding \$08 to ADR
1C9E	OPCELL	Load BYTE with current open cell
1CA4	KEY	Decodes key pressed and jumps to routine
1CC1	ROTIN	Rotates key pressed value into current cell
1CD8	GNCCELL	Open next cell
1D15	CLCELL	Close last cell

#### Program 2: BMOVE

```
1BC6 20 80 1E A9 53 20 40 1E
1BCE A9 3D 20 40 1E 20 96 1D
1BD6 A5 E7 85 EB A5 E8 85 EC
1BDE 20 95 1E 20 AB 1E A9 45
1BE6 20 40 1E A9 3D 20 40 1E
1BEE 20 96 1D A5 E7 85 E9 A5
1BF6 E8 85 EA 20 95 1E 20 AB
1BFE 1E A9 44 20 40 1E A9 3D
1C06 20 40 1E 20 96 1D A5 E9
1C0E 38 E5 EB 85 ED A5 EA E5
1C16 EC 48 A5 ED 18 65 E7 85
1C1E E7 68 65 E8 85 E8 A0 00
1C26 B1 E9 91 E7 A5 EB C5 E9
1C2E D0 09 A5 EC C5 EA D0 03
1C36 4C 43 FE A5 E9 38 E9 01
1C3E 85 E9 B0 02 C6 EA A5 E7
1C46 38 E9 01 85 E7 B0 02 C6
1C4E E8 4C 24 1C EA EA EA
```

#### Common routines:

1BC6	BMOVE	Inputs starting location of block to be moved
1BDE	INELOC	Inputs ending location of block to be moved
1BF9	INDADR	Inputs destination address of block to be moved
1B0C	CALC	Calculates ending address of destination block
1C24	MOVIT	Moves a byte from EBAD to DBAD
1C2A	CKFIN	Checks to see if we're finished
1C39	NFIN	Decrements two byte registers EBAD and DBAD

©

# COMPUTE! The Resource.



# OSI

# TRS-80

# COLOR-80

# OSI

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

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

## THE AARDVARK JOURNAL

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

- 1) A tutorial on Machine Code for BASIC programmers.
- 2) Complete listings of two word processors for BASIC IN ROM machines.
- 3) Moving the Directory off track 12.
- 4) Listings for 20 game programs for the OSI.
- 5) How to write high speed BASIC - and lots more -

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

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

## ADVENTURES!!!

For OSI, TRS-80, and COLOR-80. These Adventures are written in BASIC, are full featured, fast action, full plotted adventures that take 30-50 hours to play. (Adventures are interactive fantasies. It's like reading a book except that you are the main character as you give the computer commands like "Look in the Coffin" and "Light the torch".)

Adventures require 8K on an OSI and 16K on COLOR-80 and TRS-80. They sell for \$14.95 each.

## ESCAPE FROM MARS (by Rodger Olsen)

This ADVENTURE takes place on the RED PLANT. You'll have to explore a Martian city and deal with possibly hostile aliens to survive this one. A good first adventure.

## PYRAMID (by Rodger Olsen)

This is our most challenging ADVENTURE. It is a treasure hunt in a pyramid full of problems. Exciting and tough!

## TREK ADVENTURE (by Bob Retelle)

This one takes place aboard a familiar starship. The crew has left for good reasons - but they forgot to take you, and now you are in deep trouble.

## DEATH SHIP (by Rodger Olsen)

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

## VAMPIRE CASTLE (by Mike Bassman)

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

## NEW-NEW-NEW TINY COMPILER

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

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

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

TINY COMPILER - \$19.95 on tape or disk OSI

## SUPERDISK II

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

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

It also has a full screen editor (in machine code on C2P/C4)) that makes corrections a snap. We'll also toss in renumbering and program search programs - and sell the whole thing for - SUPERDISK II \$29.95 (5 1/4") OSI

## BARE BOARDS FOR OSI C1P

**MEMORY BOARDS!!!** - for the C1P - and they contain parallel ports!

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

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

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

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



*Please specify system on all orders*

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

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

## WORD PROCESSING THE EASY WAY - WITH MAXI-PROS

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

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

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

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

Best of all, it is in BASIC (OS65D 51/4" or 8" disk) so that it can be easily adapted to any printer or printing job and so that it can be sold for a measly price.

MAXI-PROS - \$39.95. Specify 5 1/4 or 8" disk.

## SUPPORT ROMS FOR BASIC IN ROM MACHINES

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

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

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

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

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

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

**QUEST - A NEW IDEA IN ADVENTURE GAMES!** Different from all the others, Quest is played on a computer generated map of Alesia. Your job is to gather men and supplies by combat, bargaining, exploration of ruins and temples and outright banditry. When your force is strong enough, you attack the Citadel of Moorlock in a life or death battle to the finish. Playable in 2 to 5 hours, this one is different every time.

16K COLOR-80 OR TRS-80 ONLY. \$14.95



# OSI



# COLOR-80

[www.commodore.ca](http://www.commodore.ca)



# Getting Your Atari Disk Drive Up To Speed

Bob Christiansen  
Vice-President, Quality Software  
Reseda, CA

If you have an Atari 810 disk drive that has always worked reliably, then count yourself fortunate. At Quality Software we have spent about half our original investment repairing our Atari drives. Other publishers of software for the Atari Personal Computers have told us of similar experiences. It appears that the 810, at least the original version, was not built to work eight hours a day. In Atari's defense, substantial improvements have been made to the 810 since it first appeared and the reliability of newer models should be better.

One of the most frequent problems with the 810 is that it can get out of speed adjustment. The 810 is supposed to spin diskettes at 288 revolutions per minute (RPM). The hardware, which was actually designed to operate at 300 RPM, has a potentiometer that allows the RPM to be adjusted over a considerable range. Speed adjustment capability is important because many factors can vary the actual speed of the drive. The speed potentiometer is not accessible without removing the drive cover, but anyone who can handle a screwdriver can, with proper care, adjust the speed of their own 810 drive. We will explain later how to do this.

## The Symptoms Of An Improperly Tuned Drive

The symptoms exhibited by a drive that is out of adjustment are usually that it starts getting format errors (drive is too fast) or that someone else cannot read a diskette that your drive wrote (drive is too slow). Other reading and writing errors may also occur, but these two symptoms are the most common.

A drive that spins too fast is in danger of improper formatting. It may not write all 18 sectors before completing one revolution. The last sector is written on top of the first, destroying the first. A fast drive will also have trouble writing a sector onto a diskette that was formatted on a slower drive, because it will overwrite the physical space provided on the diskette.

A drive that spins too slowly packs the data so close together that the diskette becomes hard to

read, especially by another drive turning at a faster speed. Slower drives can usually read diskettes formatted and written at faster speeds, but the reverse is not true. Thus, if your friend cannot read a disk that you wrote, the most likely event is that his drive is normal and your drive is too slow. If your drive is too slow, you may never know it until you make a diskette and send it to a friend.

The fact that outside tracks have a bigger circumference than inside tracks means that data is

---

**...the 810 ... was not built to work eight hours a day.**

---

packed closer together on the inside tracks (the higher sector numbers). Therefore the higher sectors' numbers are usually the first to fail on a slow drive. These facts about slow and fast drives are generally true about all soft sectored disk drives, not just the Atari 810.

## A Program To Test RPM

Program 1 is a listing of a BASIC program that you can use to test the speed of your 810 disk drive. You don't have to know how the program works to perform the test, but explanations of both the BASIC program and the machine language program it creates are given later in this article.

Type in the BASIC program, being careful to double check the numbers in the data statements. Save the program by giving it a name like "D:RPMTEST". Then, with any formatted diskette in the disk drive, RUN the program.

The program reads sector one once each time the diskette spins around. It allows the drive to get up to speed, then reads sector one 100 times. This takes about 20 seconds. The elapsed time it takes the drive to turn 100 revolutions is clocked and the RPM is computed and printed out.

At Quality Software, we judge the outcome of the RPM test as follows:

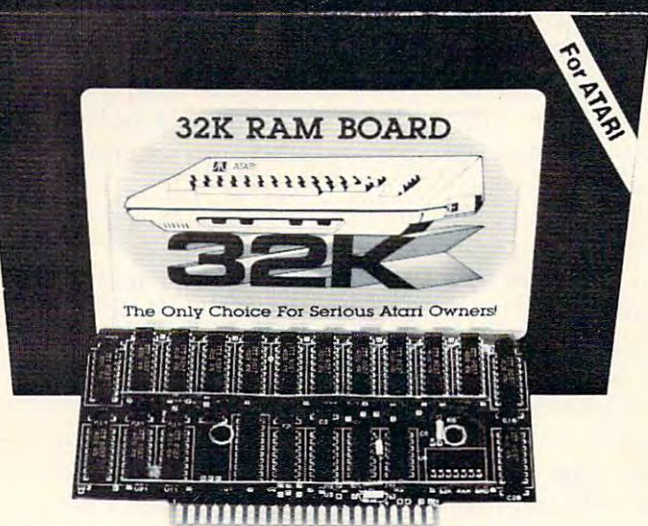
RPM	CONCLUSION
under 285	too slow
285-290	OK, don't adjust
over 290	too fast

The RPMTEST program should be accurate within plus or minus one RPM. If you run the program several times and get results that vary by more than one RPM, it may be due to one or both of the following problems which affect drive



# THE MOSAIC 32K RAM FOR ATARI

# THE BEST:



- ✓ QUALITY
- ✓ RELIABILITY
- ✓ FLEXIBILITY
- ✓ COMPATIBILITY
- ✓ SCREEN CLARITY

## 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	■	
Four 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	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 8K 32K	40K RAM	<b>Danger!</b> This Configuration Can Damage Computer
Empty 32K 8K	40K RAM	<b>Danger!</b> This Configuration Can Damage Computer	8K 32K 8K	48K RAM 40K With BASIC Cartridge	<b>Danger!</b> This Configuration Can Damage Computer
Empty 16K 32K	48K RAM 40K With BASIC Cartridge	<b>Danger!</b> This Configuration Can Damage Computer	Empty Companion 32K	32K RAM	<b>Danger!</b> This Configuration Can Damage Computer

Now from your nearest Mosaic dealer

# \$179.95

Direct from Mosaic \$189.95

**MOSAIC**  
ELECTRONICS

1-800-547-2807

P.O. Box 748, Oregon City, Oregon 97045 503/659-7574

[www.commodore.ca](http://www.commodore.ca)

\*Trade Mark of Atari, Inc.



speed:

1. Diskettes that don't turn freely inside their disk jackets significantly slow down the drive. Be sure you make the test with the "loosest" diskette you can find.
2. Drives seem to turn fastest when they are first turned on and slow down slightly as they warm up. We don't know why this is true.

### Adjusting Drive Speed

Before you attempt to adjust your drive, we must caution that the operation described here may void any warranty you have for your Atari drive. Even if the drive is out of warranty, Atari does not recommend that users attempt to adjust the speed of their drives. A special speed test diskette and an oscilloscope are necessary, they maintain, to properly set the drive speed.

Neither the author, Quality Software, nor **COMPUTE!** magazine can assume any responsibility for damage caused to your drive while attempting to make a speed adjustment. We do know that hundreds of Atari owners are already adjusting the speed of their drives with no known negative results. Atari does provide an 810 Service Manual for \$30 to anyone who insists on doing their own repairs. Write to Atari Personal Computers, 1395 Borregas Avenue, Sunnyvale, CA (attn: Lupe Soto).

If you decide you want to make a speed adjustment, carefully follow the following steps:

1. Be sure you have a clean working environment so that dust, hair, etc. will not get into the disk drive. Turn off the power to the disk drive.
2. You will need a pen knife, a small to medium sized phillips head screwdriver, and a small slot screwdriver.
3. Using the pen knife or similar instrument or strong fingernails, lift off the four plastic stick-on screw hole covers on the top of the drive.
4. Using the phillips screwdriver, remove the four screws that secure the drive cover to the base of the drive.
5. Carefully lift the cover off the drive and set it aside.
6. With the drive facing you, locate the drive speed potentiometer, which is a nylon wheel with a slot in it, in the back of the drive to the left side. We have found two different colors of potentiometers, white and blue.
7. Using the slot screwdriver, turn the tuner in a clockwise direction to slow it down, counter-clockwise to speed it up. You will have to experiment until you have arrived at the right speed.

8. Replace the drive cover before testing the drive, but don't screw it back on until you have the drive speed properly adjusted.

### The BASIC Program

Looking again at Program 1, we will explain how the BASIC program works. Lines 100 and 110 read in data statements 1000 to 1080, POKEing 73 values into the reserved RAM area starting at memory location 1536 (\$600). These 73 bytes comprise a machine language routine that determines the amount of time it takes the drive to rotate 100 times.

Lines 120 and 130 clear the video screen and ask the user to input the drive number. Line 140 pokes the indicated drive number into memory location 1610 (\$64A).

Line 150 calls the machine language routine. After returning to the BASIC program, the time it took the drive to rotate 100 times is in memory locations 1611 and 1612 (\$64B, \$64C). These values are read into A and B by line 160.

The value A is a number ranging from 0 to 255 that is incremented 60 times a second and the value B is a number that is incremented once for every time the value of A reaches 256. Thus line 170 computes the elapsed time in minutes by computing it in sixtieths of a second and dividing by 3600.

Line 180 computes RPM to the nearest unit by dividing 100 (the number of revolutions) by MINUTES. The 0.5 permits proper roundoff. Line 190 displays the result on the video screen.

### The Machine Language Program

Program 2 is a listing of the source code for the machine language program and is provided for those familiar with machine language.

The first instruction (line 0010) is required to clean up the stack for reentering BASIC. Lines 0011 to 0014 set the sector number to 1. Lines 0015 to 0017 set up a buffer area at \$500 into which the data from sector 1 will be read.

Lines 0018 and 0019 set the drive number to the value specified by the user. Lines 0020 and 0021 specify a read operation. Lines 0022 and 0023 will cause sector 1 to be read five times to ensure that the drive is up to full speed. Lines 0024 and 0026 perform these five reads by calling the resident disk handler (\$E453).

Lines 0027 and 0028 request 100 more reads. Lines 0029 and 0031 zero out two of the Atari's timer bytes (\$13 and \$14). Lines 0032 to 0034 perform 100 reads. Then lines 0035 to 0038 save the elapsed time in ALOC AND BLOC (\$64B, \$64C). Line 0039 returns control to the BASIC program.



# 6.6 SECONDS



If you spend more than 6.6 seconds  
of your valuable time reading this ad.

YOU NEED

## ***SpeedRead+***

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

### AVAILABLE NOW!

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

### SEE YOUR DEALER TODAY!

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

ATARI, APPLE II, TRS-80, and TINY C are trademarks of Atari, Inc., Apple Computer, Inc., Radio Shack, and Tiny C Associates, respectively. **SpeedRead+** is a trademark of Eagle Software and Optimized Systems Software.

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



Lines 0040 to 0043 allot space for the variables used by the program.

#### Program 1.

```
10 REM *****
20 REM *
30 REM * DISK DRIVE SPEED *
    TEST
40 REM *
50 REM *****
100 FOR I=0 TO 72:READ A
110 POKE 1536+I,A:NEXT I
120 PRINT CHR$(125);
130 PRINT "DRIVE NUMBER";:
    INPUT DRIVE
140 POKE 1610,DRIVE
150 X=USR(1536)
160 A=PEEK(1611):B=PEEK(1612)
170 MINUTES=(256*B+A)/3600
180 RPM=INT(100/MINUTES+0.5)
190 PRINT :PRINT :PRINT "RPM'S = ";RPM
200 END
1000 DATA 104,169,1,141,10,3,169,0
1010 DATA 141,11,3,141,4,3,169,5
1020 DATA 141,5,3,173,74,6,141,1
1030 DATA 3,169,82,141,2,3,169,5
1040 DATA 141,73,6,32,83,2,28,206,73
1050 DATA 6,208,248,169,10,141,73,6
1060 DATA 169,0,133,19,133,20,32,83
1070 DATA 228,206,73,6,208,248,165,20
1080 DATA 164,19,141,75,6,140,76,6,96
```

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

DON'T ASK Presents  
the new, fast-moving  
dictionary game for  
all ages:

# WORDRACE

FOR THE ATARI 400/800  
AND THE APPLE II/II+

- for 1-4 players
- requires SPEED and STRATEGY
- 2000 words and definitions
- competitive, exciting, fun

Beat the clock! Outsmart your friends!  
And while you're at it, build an impressive vocabulary.

#### 3 LEVELS OF PLAY ON ONE DISK:

Beginner — for ages 9-14  
Regular — for older teens and adults  
Challenge — for anyone who dares!

WORDRACE boosts your verbal I.Q. for college entrance exams and civil service tests.  
Great for giving school children a head start.

The challenge and pleasure of WORDRACE are irresistible.  
**At last, an educational game that's really fun to play!**

ATARI 400/800 32K DISK/BASIC \$19.95  
APPLE II/II+ 48K DISK/APPLESOFT \$19.95

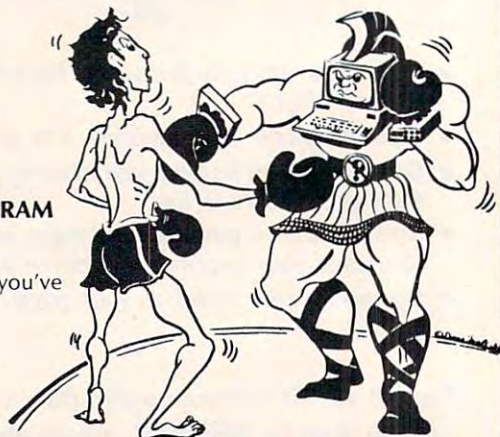
#### ALSO FROM DON'T ASK:

## ABUSE

#### THE ORIGINAL INSULT PROGRAM

- Millions of different insults.
- Guaranteed to call you something you've never been called before.

ATARI 400/800 40K DISK/BASIC \$19.95  
APPLE II/II+ 48K DISK/APPLESOFT \$19.95



At your computer store  
or send \$19.95 +\$2.00 shipping for each program to **DON'T ASK**.  
California residents add 6% tax.

ATARI is a trademark of ATARI INC.  
APPLE is a trademark of APPLE COMPUTER INC.  
Dealer inquiries welcome

**DON'T ASK**  
**COMPUTER SOFTWARE**

2265 Westwood Blvd. B-150  
Los Angeles, CA 90064  
(213) 397-8811



## Program 2.

```

0001      PON
0002 *****
0003 *
0004 * TIME 100 DISK
0005 * REVOLUTIONS
0006 *
0007 *****
0008      ORG      $600
0009      OBJ      $600
0600 68      0010      PLA
0601 A9 01    0011      LDA      #1
0603 8D 0A 03 0012      STA      $30A
0606 A9 00    0013      LDA      #0
0608 8D 0B 03 0014      STA      $30B
060B 8D 04 03 0015      STA      $304
060E A9 05    0016      LDA      #5
0610 8D 05 03 0017      STA      $305
0613 AD 4A 06 0018      LDA      DRIVE
0616 8D 01 03 0019      STA      $301
0619 A9 52    0020      LDA      #'R
061B 8D 02 03 0021      STA      $302
061E A9 05    0022      LDA      #5

```

```

0620 8D 49 06 0023      STA      NREAD
0623 20 53 E4 0024      LOOP      JSR      $E453
0626 CE 49 06 0025      DEC      NREAD
0629 D0 F8      0026      BNE      LOOP
062B A9 64      0027      LDA      #100
062D 8D 49 06 0028      STA      NREAD
0630 A9 00      0029      LDA      #0
0632 85 13      0030      STA      $13
0634 85 14      0031      STA      $14
0636 20 53 E4 0032      LOOPA     JSR      $E453
0639 CE 49 06 0033      DEC      NREAD
063C D0 F8      0034      BNE      LOOPA
063E A5 14      0035      LDA      $14
0640 A4 13      0036      LDY      $13
0642 8D 4B 06 0037      STA      ALOC
0645 8C 4C 06 0038      STY      BLOC
0648 60      0039      RTS
0040 NREAD DS      1
0041 DRIVE DS      1
0042 ALOC DS      1
0043 BLOC DS      1

```

## SYMBOL TABLE

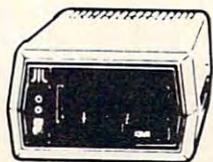
```

LOOP 0623, LOOPA 0636 NREAD 0649
DRIVE 064A ALOC 064B BLOC 064C

```

©

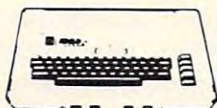
## Look Out! MX 80 PRO/WRITER IS HERE



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
- Atari to Pro/Writer Word Processor Free with Printer
- Atari
- Zenith
- Printers
- Atari to Pro/Writer Software
- Hi Res. Key & Dot Graphics
- 100 CPS
- Proportional Spacing
- Math Symbols
- And More For Less
- Paper
- Elephant Disks
- Software

**Poquette's**

14538 Lynshar Road, Grass Valley, CA 95945  
Dealer inquiries invited (916) 272-6808



## UNLOCK YOUR ATARI

"ATARI HAS THE CAPABILITY—  
ACE GIVES YOU THE KEY"  
**RIGHT IN YOUR POCKET!!**



16 comprehensive pages

- error codes
- basic commands with abbreviations
- peek and poke locations
- internal codes
- machine language aides
- much, much more!!

**Get Yours Now only \$9.95**

(dealer ad space available)

**ORDER NOW-CASH, CHARGE OR C.O.D.**

ADVANCED COMPUTING ENTERPRISES  
5516 ROSEHILL  
SHAWNEE, KS. 66216  
(913)262-2875 • (913)631-4180



ATARI IS A REGISTERED TRADEMARK OF ATARI INC.



*It is not immediately obvious, but if you know machine language and want to change some aspect of your ROM, you could move the code down into RAM and adjust the JSR's, JMP's, etc. to reflect the new locations. Then, you can modify the previously ROM-frozen program all you want. Here, the Apple II Plus Monitor is augmented by adding Step and Trace functions after moving it into RAM.*

# Softmon: Restoring Trace And Step To The Apple II Plus Monitor

R. Hiatt  
Brock University  
St. Catharines, Canada

A monitor trace is a useful tool for debugging machine code programs, in the same way that TRACE works for BASIC. In addition, it is invaluable for deciphering commercial binary routines which tend to be convoluted and to rewrite parts of themselves so that a disassembled listing taken at one point of a run will differ from that taken at another point.

The old Apple II *had* monitor trace, and the Apple II Handbook gives a disassembled listing of Steve Wozniak's monitor, resident at \$F800-\$FFFF, (as is the monitor of the current II Plus). Comparison of the new with old monitor shows relatively few differences, only 200 or so bytes of code. In particular, the subroutine STEP (\$FA43-FAD6, \$FAFD-FBID) has been changed so that keyboard entry of 'AdrT' or 'AdrS' <CR> simply does an RTS.

The II-Plus monitor, being in ROM, can't be changed, of course, (and perhaps that's just as well). But it can be copied. In fact, it will obligingly copy itself. The procedure is as follows:

Key in:

1. CALL-151 <CR> (puts you into monitor)
2. 4800<F800.FFFFFM <CR> (moves, i.e., copies the code from the second address through the third address to a location starting at the first

address.)

The move is so fast that it's hard to believe anything has happened, but you really do have a copy of the monitor in RAM, residing at \$4800-4FFF, and *this* can be changed in any way you want. Moreover, it already works. Try it; leave monitor with RESET and execute a CALL 20329. The familiar star appears on the screen and you're in *soft monitor*.

The soft monitor appears to do everything that the resident monitor does. In fact, the resident monitor is still doing all the work. The only part of the soft monitor being used is the directory (now at \$4F65-4FFF) which still calls subroutines in the \$F800-FFFF area. Some changes have to be made. (If you're making changes on your own, it's expedient to make one change right away, and that's the prompt character. For example replace the \$AA at \$4F6A with \$A3. This will produce a "#" instead of a "\*". Without this, it's awfully easy to get back into resident monitor without knowing it.)

The changes I was interested in making are done by the BASIC program, Makesoftmon (Program 1). In fact, the program does the whole job, including the original monitor copy, since that subroutine can be called from BASIC after POKEing the appropriate addresses (lines 110-130). A simple loop (lines 160-190) detects the JSR's and JMP's that transfer control within monitor, and changes the high byte of the address from \$Fx to \$4x. A few discrete POKES are necessary (lines 210-220) to change the high bytes of some indirect addresses and to restore the character table to old monitor form. Finally, the STEP subroutine is read from DATA statements and POKEd.

That completes Softmon. For the user who doesn't want the soft monitor at \$4800-, but somewhere else, Makesoftmon is easy to change. It's just a matter of replacing all \$4x high bytes with the high byte address desired.

Obtaining the STEP and TRACE functions via Softmon carries a fairly high overhead. Ideally, Softmon should call the resident monitor for those routines it handles and carry code only for those things it won't. But that's another project. At the moment, I'm more interested in unravelling some other mysteries using the newly acquired functions.

To use monitor TRACE and STEP with Softmon in memory (\$4800-\$4FFF), either CALL 20329 or CALL-151 and then execute 4F69G.

To execute a *trace*, key in the desired start address (in hex, of course), followed by "T," and then carriage return (CR). For STEP, key in start address followed by "S," CR. To go on to the next instruction, simply key "S," CR. Both trace and step display the disassembled instruction plus the



contents of the A, X, Y, S and P registers, and most importantly, *perform* the instruction, so that branches are followed properly and new code is created.

A certain amount of discretion is involved with these. Trace scrolls the information up the CRT about twice as fast as a BASIC LIST. To see what it's doing in detail really requires a printer, and even then it's very easy to go through a ream of paper needlessly if trace encounters loops of any length. I'd be inclined to say the best use of trace would be in debugging your own programs on CRT only, to the extent of insuring that there are no infinite loops, and that the final RTS is correct.

*Step* has much more potential, the pace is user-controlled. If trapped in a long loop, one can simply step out by keying a step to the alternate branch. Better yet, rather than stepping out explicitly, monitor in the information that will cause the desired branch and then step the location again. This will ensure that subsequent step-traces will be getting correct code.

A cautionary note: since trace and step do perform the instructions, the same caution has to be observed as with any POKE or user-written machine code routine. Switches will be set or reset, *no* permanent damage can be done, but the system can be crashed or DOS can be disabled.

```

10 REM MAKE SOFTMON
20 D$ = CHR$(13) + CHR$(4)
30 DEF FN LB(D) = D - 256 * INT(D / 256)
40 DEF FN HB(D) = INT(D / 256)
50 DEF FN DC(I) = PEEK(I) + 256 * PEEK(I + 1)
60 REM PAGE ADDRESSES IN $4800-$4FFF BY HEX DIGIT
70 SB = 18432: S9 = 18688: SA = 18944: SB = 19200: SC = 19456: SD = 19712: SE =
  19968: SF = 20224: SG = 20480
80 REM A1,A2,A4 HIGH & LOW BYTES
90 L1 = 60: H1 = 61: L2 = 62: H2 = 63: L4 = 66: H4 = 67
100 MS = 63488: ME = 65535
110 PRINT "COPYING MONITOR INTO $4800-$4FFF"
120 POKE L1, FN LB(MS): POKE H1, FN HB(MS): POKE L2, FN LB(ME): POKE H2,
  FN HB(ME): POKE L4, FN LB(SB): POKE H4, FN HB(SB)
130 CALL 65068
140 PRINT "CHANGING ADDRESSES FOR JSR'S AND JMP'S"
150 PRINT "CHANGED ADDRESSES ARE LISTED"
160 FOR I = SB TO SG - 1: P = PEEK(I): IF P < > 32 AND P < > 76 THEN
  190
170 P = PEEK(I + 2): IF P < 248 THEN 190
180 POKE I + 2, P - 176: PRINT I
190 NEXT
200 PRINT "MAKING A FEW OTHER ADDRESS CHANGES"
210 POKE SA + 235, 74: POKE SE + 168, 77: POKE SF + 127, 79: POKE SF + 191,
  78: POKE SF + 195, 79: POKE SF + 207, 237
220 POKE SF + 210, 236: POKE SF + 233, 195: POKE SF + 253, 79: POKE SF + 25
  5, 74: POKE SF + 106, 163
230 PRINT "READING NEW DATA AND POKING"
240 FOR I = SA + 64 TO SA + 214: READ P: POKE I, P: NEXT
250 FOR I = SA + 253 TO SB + 29: READ P: POKE I, P: NEXT
260 FOR I = SE + 194 TO SE + 201: READ P: POKE I, P: NEXT
270 PRINT "NEWMON IS NOW READY": PRINT: PRINT "TO ENTER IT, CALL 20329"
  : PRINT "OR"
280 PRINT "ENTER MONITOR WITH CALL -151": PRINT "AND THEN KEY '4F696'"
290 INPUT "SAVE SOFTMON ? "; Q$: IF Q$ > = "Y" THEN PRINT "SAVING SOFTM
  ON, A$4800, L$800": PRINT D$ "BSAVE SOFTMON, A$4800, L$800"
300 END
310 DATA 255, 255, 255, 32, 208, 72, 104, 133, 44, 104, 133, 45, 162, 8, 189, 16

```



320 DATA 75,149,60,202,208,248,161,58,  
240,66,164,47,201,32,240,89

330 DATA 201,96,240,69,201,76,240,92,  
201,108,240,89,201,64,240,53

340 DATA 41,31,73,20,201,4,240,2,177,  
58,153,60,0,136,16,248

350 DATA 32,63,79,76,60,0,133,69,104,  
72,10,10,10,48,3,108

360 DATA 254,3,40,32,76,79,104,133,58,  
104,133,59,32,130,72,32

370 DATA 218,74,76,101,79,24,104,133,  
72,104,133,58,104,133,59,165

380 DATA 47,32,86,73,132,59,24,144,20,  
24,32,84,73,170,152,72

390 DATA 138,72,160,2,24,177,58,170,  
136,177,58,134,59,133,58,176

400 DATA 243,165,45,72,165,44,72

410 DATA 24,160,1,177,58,32,86,73,133,  
58,152,56,176,162,32,74

420 DATA 79,56,176,158,234,234,76,11,  
75,76,253,74,193,216,217,208

430 DATA 211,198,52,32,117,78,76,67,74

©

# TERMINALL

## HAM RADIO INTERFACE



&  
**TRS-80**



**TERMINALL** now available for **APPLE II**! **TERMINALL T2** is a hardware and software system which converts your Apple II computer into a state of the art communications terminal. Simply insert the interface card into any available card slot (1-7) in your APPLE, then plug the attached cables into your radio.

**Simplicity.** **TERMINALL** was designed from the outset to be easy to connect to your radio and easy to use. Plug into your receiver headphone jack and copy Morse Code or radioteletype (RTTY). Plug into your CW key jack and send Baudot or ASCII RTTY using audio tones (AFSK). That's all there is to hooking it up.

Complete software on DOS 3.2 diskette (Muffin to 3.3), assembled and tested hardware, and extensive instruction manual. Requires an APPLE II or APPLE II Plus, 48K and disk.

The software is loaded into your computer from disk. Enter your call sign and the time and you will start receiving immediately. No settings or adjustments are necessary to receive Morse Code, it's fully automatic - and it works! You may type your message while receiving or transmitting. You will be on the air, receiving and transmitting in any mode, in minutes. As we said, **TERMINALL** is simple.

■ **TERMINALL** has the RTTY terminal unit - demod and AFSK - built in. This results in a lower total cost.

■ **Fantastic Morse reception.** Six stage active filter demodulator copies the weak ones. Auto adaptive Morse algorithm copies the sloppy ones. Received code speed displayed on status line.

■ **Outstanding documentation.** Professionally written, 90 page user manual contains step-by-step instructions.

■ **Built in, separate, multi-stage active filter RTTY and CW demodulators.** No phase lock loops. RTTY demodulator has 170 and either 425 or 850 Hz shift - keyboard selectable - and uses either the panel meter or scope outputs for easy tuning. Copy the weak ones. Copy the noisy ones. Copy the fading ones.

■ **Built in crystal controlled AFSK.** Rock stable for even the most demanding VHF or HF applications. A must on many VHF RTTY repeaters.

■ **Built in 110 or 220 volt AC power supply.**

■ **Built in parallel printer driver software.** Simply attach a parallel ASCII printer (e.g. the EPSON MX-80) to your printer port to obtain hard copy in all modes.

■ **Multi level displays** - allows examining and editing of historical text. Word wrapping, word mode editing, diddle, ignore carriage returns, user programmable end of line sequence, adjustable carriage width, multi user - defined WRU, transmit delay (fixed, none or auto adaptive), break mode and more!

■ **The all-in-one TERMINALL** design makes it great for use on HF or VHF Ham, Commercial, SWL or MARS! SWL's: **TERMINALL** may be jumpered for either 425 or 850 Hz reception to copy news and weather services.



**TO ORDER**  
(209) 667-2888  
(209) 634-8888

**MACROTRONICS, inc.**

C/O DEPT CP  
1125 N. Golden State Blvd.  
Turlock, California 95380

# EDITRIX + GRAPHTRIX

**EDITRIX + GRAPHTRIX = THE MOST POWERFUL WORD PROCESSOR  
THIS SIDE OF A NEWSPAPER COMPOSITION ROOM**

## EDITRIX™ TEXT EDITOR

### EASY TO USE

- **HELP!** Key
- Friendly, **COMPLETE** instructions that you or your secretary can understand.
- Easy to remember 1 or 2 keystroke commands.
- See your document formatted on the screen **AS YOU EDIT IT.**

### POWERFUL

- 250 Column Horizontal Scrolling.
- Automatic Graphic Insertion and Formatting.
- Automatic Footnote Insertion.
- Underline - Superscript - Subscript - Search - Replace - Block Move.
- Full Printsize, Emphasis, Justify, Margin and Cursor Control.

### FLEXIBLE

- Capital letters with ESC or Shift Key modification.
- To be supported by Data Transforms new headline generator coming soon.
- Printout through **GRAPHTRIX** to 11 different Printers **WITHOUT CHANGING YOUR TEXT FILE!**

**REQUIRES:** Apple II with 48K. Applesoft in ROM, DOS 3.3 and the **GRAPHTRIX** Matrix Graphics System

## GRAPHTRIX™

### TEXT PRINTER AND GRAPHICS SCREEN DUMP

### EASY TO USE

- Complete **READABLE** documentation.
- Fully Menu Driven.
- Self-running Introduction and Demonstration.

### POWERFUL

- Graphic Magnification, Normal/Inverse, Page Centering, Hi and Low Crop Marks, Title String.
- Automatic Formatting of Graphics in your Document.
- Print Size, Emphasis, Underline, Superscript, Footnotes, Chapters, controlled from your text file.

### FLEXIBLE

- Prints **ANY HI-RES** Graphic your Apple II can create.
- Formats Text files from Applewriter OR EDITRIX.
- Use as a Menu Driven Screen Dump OR from in YOUR OWN Applesoft Program.
- Compatible with 11 different Matrix Line Printers AND 7 different Parallel Interface Cards.

**REQUIRES:** Apple II with 48K. Applesoft in ROM, DOS 3.3 and one of the following line printers: EPSON MX-70/MX-80/MX-100, ANADEX 9500/9501, IDS 440G/445G/460G/560G, CENTRONICS 739, MPI 88G, SILENTYPE.

FROM DATA TRANSFORMS, INC., THE GRAPHICS LEADER

EDITRIX and GRAPHTRIX are the trademarks of Data Transforms Inc., a division of Solarstatics Inc. Apple II and Applewriter are trademarks of Apple Computer Inc. (c) Copyright 1981 Data Transforms, Inc. 616 Washington, Suite 106, Denver, CO 80203 (303) 932-1501 All Rights Reserved



*Did you ever type NEW and regret it? In theory, your program in memory is wiped out by the word NEW. On the Atari, memory is, in fact, washed clean with zeros. But the Apple and PET/CBM computers just reset some pointers (the program is still in there). Here's how you un-new.*

# Recovering From NEW On Apple And CBM

Jimmy Stephens  
Athens, GA

If you are like me, at least once in your life you have had that sickening feeling that comes with the realization that you have just typed NEW, or maybe turned your computer off, without saving your program. The next thing you probably did was to look for the nearest window to jump out. Unfortunately, if you have just turned off the computer, I can't give you any alternative to jumping out the window. (Hopefully it's a *text window*.) What I can do is to show you how to combat an accidental NEW.

When the NEW command is processed by Apple and CBM computers, the program in memory is not erased. All that happens is that several pointers are reset so that, when the next program is entered, it will write over the old one. If you have entered NEW and have not written any lines to a new program or loaded another program, then your old program is still in memory and can be recovered with a little work.

On both Apple and CBM, NEW destroys only two bytes of the actual program. Fortunately, these two bytes deal with the length of the first line of the program and can be recovered. The first POKE to be made is POKE 2050,8 on the Apple or POKE 1026,4 on CBM. Now, try LISTing the program. Surprise! You should see the first line of the program on the screen.

Your next POKE will be determined by the number of bytes in this first line. Count the number of characters in the line, remembering that reserved BASIC commands are tokenized and take up only one byte. Therefore, count all BASIC commands as one character. Add six to your total and POKE this value into location 2049 on the Apple or location 1025 on CBM.

There is a good chance, because of spaces in the line and such, that this POKE will not be correct the first time. LIST the program again. If it LISTs O.K. then the POKE was right and you can skip to

the next paragraph. Otherwise, you will need to try a new value and redo the POKE. Keep trying values around your original total until the program LISTs correctly.

At this point, although it looks alright, *do not run the program*. If you do, the variables will overwrite the program and all your work will have been for nothing. You will have to make POKEs in a pair of locations to reset the pointer for the beginning of the variables. For CBM computers, these locations depend on which ROM revision your system has. For Original ROMs, these locations are 124 and 125, and, for both Upgrade and 4.0, ROMs the locations are 42 and 43. The Apple has a pointer to the start of variable space as well as a pointer to the end of the program. This means that if you have an Apple, you will have to make all remaining POKEs in both of these locations. The first pointer is at 105 and 106 and the second one resides at 175 and 176.

The next POKEs are designed to give your program some room to breathe and are determined by the amount of RAM you have. These POKEs are only temporary and will be changed later. Use the formula below to determine the value of the first POKE.

$((\text{Number of Kilobytes of RAM}) * 4) - 2$

This POKE is made in the high byte of the variable pointer(s). Next, POKE 0 into the low byte of the pointer(s). For example: On a 16K Apple, the operation would look like this

POKE 105,0 : POKE 106,62 : POKE 175,0 :  
POKE 176,62

Your program will now run, but, if it uses lots of variables, you will soon get an OUT OF MEMORY message. To fix this, add the following lines to your program:

```
0 I=XXXX
1 I=I+1:IFPEEK(I)<>81THEN1
2 IFPEEK(I+1)<>81THEN1
3 I=I+6:A=INT(I/256):B=I-256*A
4 POKE YY,B:POKE ZZ,A:END
63999 QQ
```

XXXX = 1024 for CBM or 2048 for Apple.

YY = low byte of variable pointer(s).

ZZ = high byte of variable pointer(s). (Two more POKEs for the second pointer will be necessary on the Apple.)

When the program is run, it will change the POKEs to the correct value. Obviously, if your program's first line number is less than five or if it has a line numbered 63999, you will have to make some adjustments. Also, you will need to make sure that "QQ" does not appear anywhere else in the program. If it does, you will have to use another two-letter sequence in line 63999 and you will need to adjust the ASCII codes in lines one and two accordingly.

©



*There are times when you will write a program that is so large that it will need every last memory cell in your computer. This article shows several ways to reduce the size of your programs. In general, these techniques apply to any computer using BASIC, not just to the VIC-20.*

# Putting The Squeeze On Your VIC20: Getting The Most Out Of 5000 Bytes

Stanley M. Berlin  
Dallas, TX

Five-thousand of almost anything seems like a lot; a Christmas bonus of \$5000 would make anyone happy; 5000 jelly beans would be more than even our President could eat; 5000 days is over fourteen years. However, there are other circumstances when a quantity of five-thousand really is not so much. Five-thousand raindrops would probably go unnoticed. The time when five-thousand is a really small quantity is when you are writing a program and have only five-thousand bytes (memory locations) in which to do the job!

I remember working twenty years ago with a 4000 character IBM 1401 Computer and feeling confident that if I had only another twenty-five memory locations I would be able to complete the program. Times have changed during the last twenty years and there are many programmers who now work with *virtual memory systems*. [Where the computer can use disk memory as if it were RAM — Ed.] where there is no such problem as being constrained by the amount of memory. With technology moving as fast as it is, persons working on small micro-computers probably will not have to wait twenty years for a virtual memory-like system. When that time arrives, people will not have to write articles like this one. However, today, if you are writing programs for the Commodore VIC-20 you will have to live with the constraint of having only 5,000 bytes worth of memory in which to do your work.

Anyone who has done any serious programming for the VIC20 knows that it does not take many BASIC statements before you get that dreaded "OUT OF MEMORY" message. Of course, the VIC20 is nice enough to let you know when you turn it on that by the time it gets through allocating

506 bytes for the video mapping, and another 506 bytes for color mapping on the screen, and reserves memory space for such other things such as tape cassette buffers, you have only 3583 bytes of memory in which to store your program.

So, there you are busily entering your new BASIC program and VIC sends you the "OUT OF MEMORY" message. What are your options? You can resign yourself to the fact that, no matter what you do, the program will never fit into memory and abandon your project. Surely, no programmer worth his or her salt would exercise this option! Another option is to run out and purchase a memory expansion unit. This is not too bad a solution except, at the time this is being written, there is no such item available. Even if there were, it is surely a costly solution. The last option is to roll up your sleeves and dig into your program with a finely honed scalpel to perform surgery on it. That is certainly the most challenging option, and it is the purpose of this article to pass on a few points to help you in your efforts.

The items discussed will be from a BASIC programmer's point of view. Technical system information will be avoided except when it is necessary for a clear understanding of the issue. Although these suggestions are aimed at the VIC20 owner and the VIC20 is the computer used to validate the data, much of the information is pertinent to other computers using Microsoft BASIC.

One last point prior to moving to the meat of this article; many of the suggestions presented here are a tradeoff between good program documentation and the amount of memory used. Remark (REM) statements liberally scattered throughout a program provide the roadmap when you are



# VIC-20 SOFTWARE

from  
MIDWEST MICRO

## GRAFIX MENAGERIE .... \$11.95

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

## GRAFIX DESIGNER .... \$14.95

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



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

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

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

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

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

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

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

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

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

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

MIDWEST MICRO Associates

PO Box 6148

K.C. Mo. 64110

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

## 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

## FOR THE VIC-20®

### • TYPING TUTOR - \$9.95

If you've ever wanted to learn touch typing, this is for you! Makes learning the keyboard easier. 4 programs on one tape teach the keys in the correct progression. Automatically advances to new keys as your skills develop. Highly praised by customers. "Fantastic".

### • WORD INVADERS - \$8.95

Sharpen up your touch typing skills by blasting the invading words out of the sky.

### • FLASHCARD MAKER & FLASHCARD QUIZ - \$9.95

2 programs on one tape allow you to prepare your own study material and make it easier to learn. Quiz program has options for study, full test and easy learning mode. Keeps score and allows re-test of missed questions or entire set. Used by school systems. Includes sample data tape with 50 states and their capitals.

### • HANGMAN - \$8.95

Rewards the correct answer with music and a dance as well as providing the traditional hanging for the wrong answer. Includes full instructions to change vocabulary and use as a learning tool.

### • VIC-DIS (DISASSEMBLER) - \$9.95

For the beginning or expert machine language programmer. Provides best screen display and, if you have the VIC-1515 printer, great printout. Allows disassembly of M/L programs of up to 1K bytes loaded from tape and ROM routines in the unexpanded VIC. HEX/DEC and DEC/HEX conversions.

### • SHARK JAWS - \$8.95

Swimmers try to cross the channel, round the buoys and reach the boat as sharks attack. Harpoon the sharks and beat the high score.

Available at dealers or by mail

Shipping & handling \$1.00 per order  
California residents add 6% sales tax

ACADEMY SOFTWARE  
P.O. BOX 9403  
SAN RAFAEL, CA 94912



trying to debug a program.

At this time, most VIC20 owners probably do not have the benefit of a printer, so there is no printed copy of the program on which to make comments regarding the various routines used in a program. If you are going to do any serious programming, a printer makes the task much easier because you can follow the logic and flow of a program from beginning to end without having to enter multiple LIST statements. A disk drive provides a lot of speed and flexibility when you are using a program, but a printer is worth its weight in gold when you are trying to debug a program. If it is necessary to remove REMark statements from a program in order to conserve memory space, it is worthwhile keeping some handwritten notes concerning the program. At a minimum, you should write down the BASIC line numbers of subroutines and major sections of the program so that you will at least have an idea of what area of the program to LIST when you want to look at or change an area of code.

### REMs And Blanks

That brings us back to being confronted with the "OUT OF MEMORY" message and the first technique for buying back a few bytes of storage. REMark statements, as important as they are, require memory. They do not provide any function in your program. A REMark statement on a line by itself will require a minimum of six bytes, even if there is no text associated with the REMark. If the REMark contains textual information (as it usually does), add the length of the text to that six bytes.

The quickest way to obtain memory is simply to remove the REMark statements. Remember to write down a notation about the REMark so that the information is at least externally preserved for documentation purposes. One word of caution: if your program contains GOTO or GOSUB statements whose object is a line number containing a REMark that you removed, you will receive an "UNDEFINED STATEMENT" error message. Should you get an "UNDEFINED STATEMENT" error message you will have to figure out from where the REMark was removed and change the GOTO line number to be the line following the original REMark; this can be a long and tedious task if your program contains many statements that GOTO a REMark statement. Try to avoid this situation when you are originally writing your program by not coding GOTO or GOSUB statements which land on a line number containing a REMark.

Another way to buy back a few bytes of memory at the expense of good "internal" documentation is to remove all unnecessary blanks from the program.

This makes the program a bit harder to read, but every blank removed is a byte of usable memory. That does not seem to be much, but if you add up the number of unnecessary blank spaces in your program, you will be surprised; besides, no one ever said that this was going to be easy!

The VIC20 makes it easy to remove the blanks with the use of the INST/DEL key, but remember not to remove any blanks from within quotes. Data between quote marks are called *strings* and, if you are displaying them on a television or printer, you undoubtedly need those blanks. For example, if you are displaying the message "PRESS X TO EXIT" you would not want that information displayed as "PRESSXTOEXIT".

The following routine was entered on the VIC20:

```
NEW
100 PRINT "clr"
200 A=2:B=3:C=4:D=5
300 IF A = B AND C = D THEN D = D + 1: GOTO
    400
400 PRINT "FREE=";FRE(X)
```

The results of running this program showed that there were 3465 bytes available; the program occupied 118 bytes ( $3583 - 3465 = 118$ ). By simply removing the fifteen blanks in statement 300 it became less readable:

```
300 IFA=BANDC=DTHEN D=D+1:GOTO400
```

but the results of that run showed 3480 bytes of free space; a 100% return for each blank removed. Finally, you might have observed that I did not remove the blank that separates the line number from the statement. It does not really occupy any memory and is there only for readability; in fact, if you remove that blank, you will find that BASIC will reinsert it when you LIST the statement.

### Multiple Statements And Short Variable Names

Closely allied with removing blanks and removing REMark statements is putting multiple statements on a line. Your VIC20 can only display 22 characters on a line, but BASIC will actually accept up to 80 characters. It is possible to have your BASIC statements occupy about three and one half display lines. It is also possible to combine statements on one BASIC line by using the colon separator character. Every line number in your program contains an overhead of five bytes (for technicians: that five bytes consists of two bytes for the line number, two bytes for an internal pointer, and one byte for a delimiter at the end of each statement). You can save four of these five bytes for every statement combined on a line (the colon separator will use one of the five bytes eliminated). For example, instead of coding:



```
100 A = A + 1
200 IF A > 25 THEN Z% = 0
```

you can save four out of the five byte overhead of the second line by coding:

```
100 A = A + 1:IF A > 25 THEN Z% = 0
```

However, no suggestions are free of charge and there is also something to watch out for in this instance. You may freely combine statements for up to eighty characters, but it is possible that one of the statements you are combining might be the object of a GOTO or GOSUB statement, in which case you will receive the "UNDEFINED STATEMENT" error message. In the example immediately above, if there were another statement in the program which was "GOTO 200", line number 200 would not be in the program after combining the two lines and you would get the error message. If you had a statement that was "GOTO 100", that would not cause any problems.

Another item to watch for when combining statements is not to combine a line with a preceding line that contains an IF statement. The statement shown above is alright. However, if the two lines were reversed:

```
100 IF A > 25 THEN Z% = 0
200 A = A + 1
```

you would not be able to combine the lines as:

```
100 IF A > 25 THEN Z% = 0:A = A + 1
```

without altering the meaning of the statement. In this instance, the addition statement would only be executed if A were greater than twenty-five which is not the intent of the original.

The last item that returns a little usable memory at the expense of readability and documentation is the use of short variable names. Although variable names may be up to 255 characters, BASIC uses only the first two characters (plus the \$ and % suffixes for string and integer variables respectively). Each character in the variable name occupies a byte wherever it is used; therefore you should limit variable names to two characters and one character would be even more thrifty from a memory-use point of view. Limiting the names to two characters could have a side benefit inasmuch it may eliminate a potential source of programming error. If you had two variables, one named "TAPE" and the other named "TASTE," BASIC would only recognize "TA" as the name and would, in effect, be dealing with a single variable.

Avoiding a technical discussion as to why it is so, it is usually more economical to use constants instead of variables whenever possible. A constant consists of data stored in the BASIC statement itself. Constants may require less memory than variables, especially in cases where the constant is a

relatively short string. As the length of the string increases, the amount of savings diminishes because the repetition of the constant also occupies memory.

The following lines each contain a constant:

```
100 A = A + 1
    ("1" is the constant)
100 S$ = D$ + "SUFFIX"
    ("SUFFIX" is the constant)
100 PRINT "TOTAL=";X
    ("TOTAL=" is the constant)
```

An illustration of the savings that can be gained is:

```
100 T$ = "THIS IS A TEST"
200 PRINT T$
300 PRINT T$
```

occupies 72 bytes of memory, whereas

```
100 PRINT "THIS IS A TEST"
200 PRINT "THIS IS A TEST"
```

only occupies 69 bytes of memory. That is not much of a savings because the string "THIS IS A TEST" is relatively long; if it were shorter, the savings would be more dramatic. The reason for this is that, when data is assigned to a variable, it requires two areas of memory, but a constant requires only one (in the instruction itself).

BASIC is sometimes very shrewd as far as memory management is concerned. BASIC is smart enough to know when you have used a string and will reuse it rather than recreating it again in memory. Thus, if you have coded the statement: PRINT "THIS IS A TEST" and, elsewhere in your program you coded A\$ = "THIS IS A TEST", although memory would be required to contain pointers for the variable "A\$", the actual text string "THIS IS A TEST" would not be recreated in memory (except in the instruction itself). The original text string would be pointed to by the variable. This is starting to border on the kind of technical information that this article has tried to avoid, but is interesting enough to pass on.

### Don't Avoid Integer Variables

It seems that most people writing BASIC programs never bother with integer variables and, yet, that is where a significant savings in memory can be attained. This is particularly true if the program contains arrays. Consider that, for each element in an array, the number of bytes occupied is as follows:

```
A string array = 3 bytes plus the length of the
                  string per element
A floating point array = 5 bytes per element.
An integer array = 2 bytes per element.
```

The contents of many arrays do not require the use of decimal points, but it is easier to code "DIM A(15)" rather than "DIM A%(15)." By using the integer form, you would save 45 bytes of mem-



ory. Suppose you were writing a program to deal a deck of cards and you defined an array to keep track of which cards have already been dealt. That fifty-two element array could be a string array, a floating point array, or an integer array. Obviously, there is no need for decimals in this example so the obvious choice would be to use integers. The following program was run three times on the VIC20; each time changing the type of array (making the variable type in statement 400 correspond to the array).

```

100 PRINT "clr"
200 DIM X(100)           <===== First run
    DIM X%(100)          <===== Second run
    DIM X$(100)          <===== Third run
300 FOR Z = 0 TO 99
400 X(Z) = Z             <===== First run
    X%(Z) = Z            <===== Second run
    X$(Z) = CHR$(Z)      <===== Third run
500 NEXT Z
600 PRINT FRE(X)

```

The differences in memory use during these three runs is very dramatic:

The first run used a floating point array and occupied 601 bytes; the second run used the string array and occupied 403 bytes; and the final run, which used an integer array, occupied only 300 bytes. Is it worth the cost of 300 bytes in order to save typing in a "%" each time the variable is used?

Each of the memory conserving measures outlined so far would be relatively easy to implement using the editing capabilities of the VIC20 once a program is written and resident in memory. The last few suggestions are harder to implement and the savings are more indefinite.

If your program contains groupings of instructions that are repeated several times, it would be more memory efficient (and better programming practice) to incorporate those instructions once as a subroutine and GOSUB to them. If such statements are readily identifiable, you can implement this fairly easily with the following three steps:

1. Add a RETURN statement after the first group of statements.
2. Place a GOSUB statement whose object line number is the first line number in the group immediately preceding the group.
3. Add a GOTO statement immediately after the inserted GOSUB statement whose object line number is the statement immediately following the RETURN statement.

Naturally you would delete all other occurrences of the same group of statements and replace them with a GOSUB to the newly created subroutine. This sounds very complicated, but actually is quite easy to implement and is illustrated in the fictitious

routine that follows. Assume that the statements starting with line number 650 and ending with line number 710 are repeated several times in the program.

```

640 PRINT "ABC"
650 A = A + 1
660 IF A = 9 THEN 700
670 PRINT "MESSAGE ONE"
680 MC = MC / A
690 GOTO _____
700 PRINT "MESSAGE TWO"
710 MC = MC * A
720 IF Q + 1 = 10

```

You can convert this to a subroutine using the technique described by adding lines 645 and 715 in the following illustration:

```

640 PRINT "ABC"
645 GOSUB 650: GOTO 720 <===== New statement
650 A = A + 1
660 IF A = 9 THEN 700
670 PRINT "MESSAGE ONE"
680 MC = MC / A
690 GOTO _____
700 PRINT "MESSAGE TWO"
710 MC = MC * A
715 RETURN <===== New statement
720 IF Q + 1 = 10

```

Another almost obvious way to decrease the amount of storage your program uses is simply to reduce the size of messages that you display on the television. For example, if your program displays the word "TOTAL", change it to "TOT". If your program contains cards, instead of spelling out "KING," "QUEEN," and "JACK," simply use a "K," "Q," and "J" respectively.

Another item to investigate is the use of more economical instructions to achieve the same results. Shown below is an example of how you can replace multiple IF statements with a single ON statement. The program does the exact same thing either way you write it, but using the ON statement yields a savings of 62 bytes.

The following statements

```

300 IF A = 1 THEN 510
310 IF A = 2 THEN 550
320 IF A = 3 THEN 600
330 IF A = 4 THEN 650
340 IF A = 5 THEN 700
350 IF A = 6 THEN 750

```

could be replaced with the single statement:

```

300 ON A GOTO 510, 550, 600, 650, 700, 750

```

Consider using FOR/NEXT loops instead of repeating instructions wherever possible. Suppose you wanted to print a vertical line down the center of the screen, you could program it as:

```

300 PRINT TAB (10); "I"

```



```

310 PRINT TAB (10);"I"
320 PRINT TAB (10);"I"
330 PRINT TAB (10);"I"
340 PRINT TAB (10);"I"
350 PRINT TAB (10);"I"
360 PRINT TAB (10);"I"
370 PRINT TAB (10);"I"
380 PRINT TAB (10);"I"
390 PRINT TAB (10);"I"
400 PRINT TAB (10);"I"

```

or alternatively you could code:

```

300 FOR X = 1 TO 11
310 PRINT TAB (10);"I"
320 NEXT X

```

Again, the same results are achieved, but the FOR/NEXT loop yields a savings of 118 bytes!

### Overlaying

You can sometimes conserve memory by *overlays*. If your program runs in two separate and distinct phases (that is, one portion of your program completes all its work and then is never executed again) it should be possible to split your program into two sections. Have the last statement in the first section issue the statement "LOAD PHASEII" (assuming your second phase is named "PHASEII"). When the first section completes its job, the last instruction would load the next phase for execution.

This assumes that you have written PHASEII onto the cassette tape immediately after PHASEI. This is called *overlaying*; it lends itself well to a disk-oriented system, but there is no reason not to use it with tape also. You should be aware that any variables used in the first phase will not be available in the second phase. (However, even though this should work, the author has not yet been able to do this successfully.)

If you still need memory, you may be able to put some of the data used in your program on a cassette tape and read the data in during program execution. This technique will involve your writing a special program to create the data tape, but, in some instances this can yield substantial memory savings. The premier issue of *Home and Educational COMPUTING!* contained an excellent article by David Malmberg entitled "Custom Characters for the VIC." A program shown in that article lends itself very well to illustrating this memory-saving technique.

That program contained a number of DATA statements and is shown below:

```

170 READ X:IF X > 0 THEN 190
180 FOR X = X TO X+7
182 READ J
184 POKE I,J
186 NEXT
190 GOTO 170

```

```

340 DATA 7168,24,24,36,60,102,66,66,0
350 DATA 7176,124,34,34,60,34,34,124,0
360 DATA 7184,126,34,34,32,32,32,112,0

```

```

600 DATA 7376,0,0,0,0,0,0,0,0
610 DATA -1

```

You would have to write a program to write the data to cassette tape (and ideally you would write the data immediately after the program you saved that will be using that data). The original program can be easily modified to create the data tape and an example could be:

```

100 OPEN 1,1,2,"DATATAPE"
170 READ X:PRINT#1,X:IF X > 0 THEN 190
180 FOR X = X TO X+7
182 READ J:PRINT#1,J
186 NEXT
190 CLOSE1
200 PRINT "DATA TAPE CREATED"
210 END
340 DATA 7168,24,24,36,60,102,66,66,0
350 DATA 7176,124,34,34,60,34,34,124,0
360 DATA 7184,126,34,34,32,32,32,112,0

```

```

600 DATA 7376,0,0,0,0,0,0,0,0
610 DATA -1

```

You would then substitute INPUT#1 statements for READ statements in the original program, but the results would be the same. If you wanted to use the concept in Mr. Malmberg's article in your own program, but needed additional memory, this technique will provide you with a significant amount of memory.

The modified program would be:

```

165 OPEN 1,1,0,"DATATAPE"
170 INPUT#1,X:IF X > 0 THEN 190
180 FOR X = TO X+7
182 INPUT#1,J
184 POKE I,J
186 NEXT
190 GOTO 170
340 REM -NO DATA STATEMENTS-RESULTS IN
350 REM -A SIGNIFICANT REDUCTION IN
    MEMORY

```

Let's assume that you have exercised all the memory-saving procedures outlined and your program still requires additional memory. The last thing to do is carefully investigate the logic of your program. Are there statements that are never executed (perhaps left over from an initial idea that was abandoned)? Naturally, you should remove them. Are there better ways to implement a procedure that might reduce the number of instructions



necessary to accomplish some objective? Sometimes being able to buy back just five or ten bytes of memory will allow your program to run.

One final point; there are instances when you have worked for hours on a large, complicated program and you decide to save it on a cassette tape. You type in the command "SAVE MY-PROGRAM", press ENTER, and lo and behold, instead of receiving the message to press the play and record buttons, you instead get the message "OUT OF MEMORY." This is one of the most frustrating events possible when writing a program because you do not want to lose the hours of effort already expended. Try issuing the "SAVE" command with a shorter filename; this usually works. Instead of entering "SAVE MY-PROGRAM", use "SAVE A".

It takes some effort, but using the techniques outlined here could mean the difference between being able to do what you want with your VIC20 or being continuously confronted with an out of memory condition. With a little patience you might be able to find that needle in the haystack. If all else fails, don't give up hope. Remember that your VIC20 has the capability of being expanded to 32K!



## 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

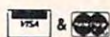
Dealer inquiries welcomed.

## LETTER QUALITY PRINTERS

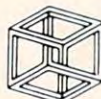
**\$995<sup>00</sup>**  
**BELIEVE IT!**

### AND CHECK THE FEATURES

- WORKS WITH ANY WORD PRO. SOFTWARE
  - 132 COLUMN PLATEN
  - 45 CHARACTERS PER SECOND
  - USES STANDARD DAISY PRINT WHEELS FOR TYPE STYLE VARIETY
  - PROPORTIONAL SPACING
  - TRACTOR AVAILABLE
  - COMPLETELY RECONDITIONED
  - 90 DAY WARRANTY
  - WE PAY SHIPPING IN U.S.
- ADD \$175<sup>00</sup> FOR APPLE OR TRS-80  
INTERFACE (PARALLEL)  
ADD \$279<sup>00</sup> FOR PET/CBM INTERFACE (IEEE-488)



ORDERS ACCEPTED BY PHONE  
NO COD'S (716) 625-8200



**QUBE**  
INTERNATIONAL  
DEPT. C-5

2521 NIAGARA FALLS BLVD.  
PO BOX 151, NO. TON., NY 14120



# Considering Space And Time In The Atari

C. Michael Levy and Grant Levy  
Gainesville, FL

Would you like to know how to speed up some of your programs, or make them use up less memory? As everyone comes to learn as they gain experience in programming, small and often very subtle changes in even very simple programs can have major consequences. Consider the following four programs. Each has exactly 1000 lines. Each makes two references to variable B, 994 assignments to variable A, contain six GOSUBs and six RETURNS, one POKE, one PEEK, one PRINT and one STOP. They all achieve the same end: they determine that the value of variable A is zero and print the number of clock ticks (each lasting 1/60th sec.) required for execution of the program. They are equivalent, right? *Wrong!*

Programs 1 and 2 each require about 15000 bytes of memory, nearly 66% more than the approximately 9100 bytes required for Programs 3 and 4. The reason for this is the way Atari BASIC handles constants and variables. Each and every reference to a constant (zero, in these programs) consumes seven bytes. This is true even if the statements using those constants are *never* executed in the program (as are lines 2-994 in Program 1 and lines 8-1000 in Program 2). Programs 3 and 4 involve the assignment of a constant to only *one* variable (B in line 1); variable A assumes its values by reference to this value. Each such reference requires only one byte.

So the first lesson is to use constants sparingly, if at all. Develop a schema for assigning variable names to constants so that you will not confuse them with "real" variables. For example, consider D0=0, D1=1, or C0=0, C1=1, etc. where the symbol "D" is a mnemonic for *digit* or "C" reminds you that it is a *constant*. Any other combination of letters and/or digits could then represent variables which *vary*.

## Speed Differences

While Programs 1 and 2 are identical in memory

requirements, and Programs 3 and 4 are also identical to each other in terms of memory, these pairs of programs are vastly different in execution speed. Here we find that Programs 2 and 4 each require only one clock tick for completion. In marked contrast are Programs 1 and 3 which are 42 times slower!

The reason for this enormous discrepancy is the way that Atari BASIC seems to locate subroutines. BASIC is apparently incapable of immediately jumping to the desired line referenced in a GOSUB. Rather, it appears to start from the first line of the program, and sequentially search through the list of lines until it finds what it wants. Thus, in Programs 3 and 4, when it encounters in line 1000 a GOSUB 999, it must begin at line one and look at each of the 998 intervening lines until it reaches 999. There, it encounters a GOSUB 998. Back to the top of the list it goes, fruitlessly examining 997 lines. And so on. BASIC performs the same sequence of steps in Programs 2 and 4, but it obviously has to do fewer of them, since all of the subroutines are near the top of the list of line numbers.

Thus the second lesson, is to place subroutines as close as possible to the beginning of each program. Some of your programs that heretofore seemed to drag on unmercifully could now have more zip.

And, finally, we answer the question that has been bothering you for some time. No, we are not masochists. We did *not* type in 1000 lines of code in order to perform these benchmark tests. Instead, a one-line program was written to create a 1000-line skeleton for Programs 1 and 2:

```
10 OPEN #2,0,"D:PROGRAM : FOR J=1 TO 1000
   : ? #2,J; " A=0" : NEXT J : CLOSE #2
```

Then PROGRAM was ENTERed, the appropriate minor changes made to only seven lines, and then RUN. The same procedure was followed for

### Program 1.

```
1 POKE 20,0 : B=0 : GOSUB 1000 : B=PEEK(20)
  : ? B : STOP
2 A=0
3 A=0
4 A=0
5 A=0
6 A=0
7 A=0
8 A=0
9 A=0
  etc.
995 A=0 : RETURN
996 GOSUB 995 : RETURN
997 GOSUB 996 : RETURN
998 GOSUB 997 : RETURN
999 GOSUB 998 : RETURN
1000 GOSUB 999 : RETURN
```



Programs 3 and 4; the sole change was that "A=0" was written as "A=A".

#### Program 2.

```

1 POKE 20,0 : B=0 : GOSUB 7
  : B=PEEK(20)
  : ? B : STOP
2 A=0 : RETURN
3 GOSUB 2 : RETURN
4 GOSUB 3 : RETURN
5 GOSUB 4 : RETURN
6 GOSUB 5 : RETURN
7 GOSUB 6 : RETURN
8 A=0
9 A=0
  etc.
995 A=0
996 A=0
997 A=0
998 A=0
999 A=0
1000 A=0

```

#### Program 3.

```

1 POKE 20,0 : B=0 : GOSUB
  1000 : B=PEEK(20)
  : ? B : STOP
2 A=A
3 A=A
4 A=A
5 A=A
6 A=A
7 A=A
8 A=A
9 A=A
  etc.
995 A=A : RETURN
996 GOSUB 995 : RETURN
997 GOSUB 996 : RETURN
998 GOSUB 997 : RETURN
999 GOSUB 998 : RETURN
1000 GOSUB 999 : RETURN

```

#### Program 4.

```

1 POKE 20,0 : B=0 : GOSUB 7
  : B=PEEK(20)
  : ? B : STOP
2 A=A : RETURN
3 GOSUB 2 : RETURN
4 GOSUB 3 : RETURN
5 GOSUB 4 : RETURN
6 GOSUB 5 : RETURN
7 GOSUB 6 : RETURN
8 A=A
9 A=A
  etc.
995 A=A
996 A=A
997 A=A
998 A=A
999 A=A
1000 A=A

```

## SWIFTY SOFTWARE TOP RATED PRODUCTS FOR ATARI



#### NEW! **HARDWARE** **DISK SENTRY**™

An intelligent digital accessory for your ATARI 810 Disk Drive, lets you selectively write data to both sides of single sided and write protected disks. DISK SENTRY cannot harm your drive or disks. Installs and removes easily; no soldering required. DISK SENTRY's LED signals system status, preventing accidental erasure of data. DISK SENTRY is a convenient push button write-protect override which can pay for itself with your first box of disks. \$39.95 + \$2.50 Shipping and Handling.

#### **ARCADE GAMES**

24K Disk; 16K Cassette; Joystick required. Add these popular HIGH RESOLUTION, REAL-TIME, ANIMATED games to your software arsenal. Get FAST ACTION and FULL SOUND GRAPHICS that take advantage of the unique features of your ATARI. Enjoy challenge that requires strategy and skill.

#### **SPACE CHASE**™

Fly against intelligent invader clones. Arm yourself with Nuclear Defense Charges and play with or without Defense Shields. Enjoy this action-packed multicolor space odyssey. Displays top score, number of planets saved and number of galaxies conquered. \$14.95 cassette; \$19.95 disk

#### **TIMEBOMB**™

Meet the challenge of this fast moving animated race against time, enemy aircraft and enemy bombs as you attempt to disarm timebombs set to explode ammunition depots. Avoid aircraft of varying sizes and speeds — and their bombs. Choose one of ten Day or Night Missions. Use from one to four Joysticks. Any number can play; top players listed on scoreboard. \$14.95 cassette; \$19.95 disk

#### NEW! **AND MORE GAMES.....** **TRIVIA TREK**™

Unlimited fun and lots of laughs for one or two players. Five hundred questions and two thousand multiple choice answers are supplied on the master diskette. A powerful datafile handling program allows creation of your own trivia questions and answers. Features include: Player Missile Graphics, user or random selection of subjects and numerous comical answer choices. This DISK ONLY package comes complete with user instructions. An incredible value for only \$29.95. Requires 32K and disk drive.

#### NEW! **FUN "n" GAMES #1**™

WORDGAMES, POSSIBLE and LEAPFROG giving you hours of fun, challenge and entertainment. WORDGAMES, two games in one, contains GUESSIT - a deductive alphabetic reasoning game for one or two players and WORDJUMBLE - a multiple word scrambling puzzle with play-on-word hints and mystery answers. Instructions show how you can substitute your own words. Use POSSIBLE to help descramble word jumble puzzles or to create your own. All letter/number combinations or permutations of input are printed to screen or optional printer. LEAPFROG is a Chinese-Checker type jumping game in which you try to position two sets of animated jumping frogs in a minimum number of moves. 16K Cassette \$19.95; 24K Disk \$24.95. Disk version of GUESSIT works with VOTRAX Type "n" TALK. A real crowd pleaser.

#### **COMING SOON! Space Shuttle Adventure Series**™

Real-time Space Flight Simulations

#### ★ **PERSONAL DATA MANAGEMENT** **FILE-IT 2**™

Contains all the programs in FILE-IT plus five additional file handling and financial programs. Financial entry and report generator programs create a powerful personal accounting system while two additional utility programs provide random access updating and user controlled record selection. Subfiles may be created, merged and sorted by any field. A monthly Bar Graph program generates a visual picture of financial data on the screen and/or printer. Supports up to four disk drives as well as the AXLON RAMDISK. Minimum requirements are 24K, 1 disk drive and an 80 column printer. Extensive documentation, supplied in a ring binder, provides clear instruction along with a tutorial on computer filing. \$49.95 + \$3.25 Shipping and Handling. AXLON RAMDISK not required.

#### **FILE-IT**™

Use this start up database system to file and manage personal information and data. Create, sort, store and manipulate information such as appointment calendars, address, or telephone data, credit or charge records, stock investments, medical or prescription information, hobby, coupon or other types of collection information...and more. With printer you get 1 or 2 across mailing labels, disk jacket inventory covers and neatly written copy of all your data files. Comes with well documented instruction manual explaining basics of computer filing. Fast and easy to use. Holds over 300 records in 40K. Requires minimum of 24K and 1 disk drive. Printer optional. \$34.95 (Disk Only)

#### **COMING SOON! The Family Financier**™

AN easy to use financial package.

#### **UTILITIES**

##### **DISKETTE INVENTORY SYSTEM**™

Use this system to gain control of your expanding disk/program inventory. Quickly get locations of single or multiple copies of your programs and all your valuable files. An invaluable tool, this system is easy and convenient to use and to update. 24K disk system required. \$24.95 Printer suggested.

##### **SWIFTY UTILITIES**

A valuable collection of programming utilities for the ATARI programmer. This DISK ONLY package includes all of Programming Aids I and additional programs designed to make programming time more efficient. Special MENU program runs both saved and listed programs. REM REMOVER eliminates REM statements so programs take less core and run faster. PRINT 825 and PRINTEPS custom print programs prepare condensed, indented and paginated program listings on your ATARI 825 or EPSON MX-80 printer. Listings identify machine code, graphics and inverse video characters. VARIABLE LIST and VARIABLE PRINT programs help you prepare alphabetized annotated list of your program variables. A delete lines utility provides convenience of line deletion while a DOS CALLER gives you convenient access to many DOS utilities while your program is in core. Disklist prepares disk jacket labels. Many of these programs work coreident with each other and with your program. Disk Drive and minimum of 24K required. \$29.95

##### **PROGRAMMING AIDS PACKAGE I**™

Four utility programs to help increase programming efficiency and learn more about your computer. RENUMBER handles references and even variables. Generates Diagnostic Tables for programming error detection. PROGRAM DECODER, DECIMAL to BCD and BCD to DECIMAL programs give you a practical way of studying internal program representation and ATARI number-conversion procedures. Comes with comprehensive user's manual. 16K cassette \$14.95; 24K disk \$19.95

##### **SWIFTY DATALINK**™

High Quality Smart Terminal Communications program. Easy to use Multi-Option, Menu Driven. Full performance uploading/downloading. Works in Duplex or Simplex modes supporting ASCII and ATASCII transmission. Printer Dump, Screen Dump and Disk Search options. Use as remote terminal. Send/receive and store programs and data files. Saves connect time charges with commercial services. Requires 24K RAM, 810 Disk Drive, 850 Interface or equivalent, 830 or other 300 Baud modem. (Printer optional) \$39.95

##### NEW! **SWIFTY TACH MASTER**™

An accurate disk speed diagnostic utility program designed specifically for ATARI 810 Disk Drives. Provides easy-to-read visual indication of the speed of any drive connected to your system. Using the accuracy of machine language, TACH MASTER displays five RPM readings per second with a working tachometer accurate to 1/4 RPM. Allows you to adjust your drive(s) to factory specs easily and at any time in the convenience of your own home. Comes complete with easy to follow user's manual. \$29.95

##### NEW! **ACCESSORIES**

##### **VINYL DUST COVERS**

New, glove soft, vinyl dust covers for the ATARI 800 Computer, the 400 Computer and the 825 Printer. Custom made from heavy duty upholstery grade vinyl, these covers completely cover the top and sides of your valuable equipment. Do not confuse them with cheap, flimsy plastic covers available elsewhere. Accessory ports and other input/output plugs are exposed for convenience of use. Available in midnight black. Atari 400: \$13.95; Atari 800: \$14.95; Atari 825: \$14.95; Atari 810: \$13.95. Specify model. Please include \$2.50 for Shipping and Handling.

send check or money order to:

### SWIFTY SOFTWARE, INC.

64 BROAD HOLLOW ROAD  
MELVILLE, N.Y. 11747  
(516) 549-9141

N.Y. Residents add 7 1/2 % sales tax

send for free catalogue dealer orders and c.o.d.'s accepted

©1981, 1982 Swift Software, Inc.  
NOTE: ATARI® is a registered trademark of Atari Inc., a Warner Communications Company and all references to ATARI® should be so noted.



# Micro Wholesale

## BEST PRICES



**VIC-20 \$248**

**5 VIC Programs \$19.95**

States & Capitols, Jackpot, Countries & Capitols, Black Jack, and Hangman.

### COMMODORE EQUIPMENT

VIC Printer	\$ 320
VIC Disk	\$ 467
PET 4016	\$ 789
PET 4032	\$ 952
CBM 8032	\$1087
Super PET	\$1636
Datasette	\$ 63
Single Disk	\$ 541
4040 Disk	\$ 983
8050 Disk	\$1333
4022 Printer	\$ 613
8023P Printer	\$ 768
8300P Daisy Printer	\$1732

Xerox 820	\$2447
Atari 400	\$ 347
Atari 800	\$ 697
Atari Disk	\$ 447
Box of 10 Elephant Diskettes No. 1 S.S. S.D. Soft Sector	\$ 21
Anadex 9501 Printer	
200CPS Matrix-Graphics	\$1197
Smith Corona Daisy Printer	
Letter Quality, Parallel or Serial	\$ 787
Comet I, 124 CPS Matrix Printer	
Bidir. Parallel or Ser-Lmt'd Qty	\$ 302
Epson MX70	\$ 344
Epson MX80	\$ 444
Epson MX80FT	\$ 547

**Other Brands Carried:** NEC, ALTOS, NORTH STAR, DIABLO, QUME OKIDATA, TELE-VIDEO, AMDEK, MAXELL, SCOTCH, DY-SAN, CITOH, ZENITH and More.

**Ask For Catalog:** Specify Commodore, Atari, Apple, North Star, or Peripherals.

**Micro Wholesale**

276 N. University, P.O. Box 1243  
Provo, Utah 84603 (801) 373-2901

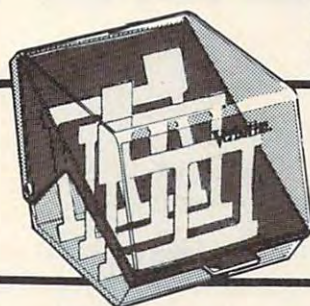
Prices are for prepayment by certified check, add 3% for Bank Cards. Personal checks accepted, but will delay shipment. When possible, all items shipped UPS, freight collect. All sales subject to availability and stock.

# SPRING DISKOUNT SALE

## DUST COVERS for ATARI

Protect your investment! Made of fabric (not cheap vinyl), these dust covers will more than pay for themselves in preventing problems associated with dirt and grime. All seams are machine sewn for durability. Colors are tan with chocolate brown trim.

Cat. No. 3818	Atari 800	\$10.95
Cat. No. 3819	Atari 400	\$8.95
Cat. No. 3820	Atari 410	\$7.95
Cat. No. 3821	Atari 810	\$7.95
Cat. No. 3822	Atari 850	\$8.95
Cat. No. 3823	Atari 825	\$8.95



## DISK PROTECTOR CASES

- \*Holds 50 diskettes
- \*Hi-impact smoked plastic
- \*Desktop-perfect for home/or office

Cat. No. 2956 \$23.00

## SHADOW HAWK ONE

HORIZON SIMULATIONS

The Galactic Empire has just conquered the Noble Free Space Confederation... except for your remote outpost on a moon at the outer limits. Employing the unmatched speed of your warship, SHADOW HAWK ONE, you prey on the Empire's Merchant Fleet to capture enemy raw materials which can be bartered for better weaponry, shielding, missiles, etc. Be Alert! The Empire Wants you.

Cat. No. 3397 Atari, 48K, disk \$49.95

## TT#5 PLAYER/MISSILE GRAPHICS

SANTA CRUZ

This is one of the best of the tutorials from Santa Cruz Educational Software. P/M Graphics is what sets 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 Atari, 32K, cass \$29.95  
Cat. No. 3401 Atari, 32K, disk \$29.95



## VERBATIM DATALIFE DISKETTES

- \*For ATARI, APPLE II, TRS-80, etc.
- \*Single sided, double density
- \*40 track tested
- \*Built-in hub rings

Cat. No. 1147 (box of 10) \$28.00

## HOW TO ORDER

Write or phone. Pay by check, M/C, VISA, or COD (add \$1.50 for COD).  
(800) 423-5387

(213) 886-9200

Offer Expires May 31, 1982  
Mention this ad and we pay shipping (UPS ground only).  
HW Electronics 19511 Business Center Dr. Dept. G5  
Northridge, CA 91324

**WHEN IN SOUTHERN CALIFORNIA,  
VISIT OUR RETAIL STORES**

# HW ELECTRONICS

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

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



*Extensibility means the ability to extend a computer language by adding new, customized commands. The & symbol has been used to extend Apple's BASIC, but it has drawbacks. Here's an alternative approach.*

# Modifying Apple's Floating Point BASIC: An & Interpreter Without the &

H. Cem Kaner and John R. Vokey  
Department of Psychology  
McMaster University  
Hamilton, Ontario

Any computer language has some features which are clumsy and lacks others that you wish you had. If the language is generally satisfactory, you can either put up with these drawbacks or, in some cases, you can "patch" the language, adding new commands or changing old ones to meet your requirements. Applesoft, Apple's floating point BASIC, is quite easy to patch, and there is a growing collection of documentation to help the programmer along. A very direct and popular approach to patching the language uses the &.

## & Interpreter

The presence of this character in a program forces a jump out of BASIC, and can force a jump to the subroutine you have written to handle a new command. As the number of available patches has grown, a further feature has been added, an ampersand interpreter. In this case, the & forces a jump to a program (the &-interpreter) which first determines which new command is being signalled, and then branches to the subroutine appropriate to that command. The &-interpreter provided by Mottala is a fine example of this kind of program. For us, this ability to interpret a wide range of user-defined commands is a very significant enhancement to Applesoft. Unfortunately, any ampersand interpreter has a drawback – that the & itself.

The problem arises if the new command flagged by the & involves a symbol Applesoft would readily treat as a command without the &. An example of this is Smith's & GOTO command (**COMPUTE!**, May, 1981, #12). This modification to GOTO allows the use of labels instead of line numbers in GOTO statements. For example, if X = 1000, then & GOTO X sends program control to line 1000. Forget that &, though, and Applesoft branches to line 0, no matter what X holds.

This quirk of Applesoft (no fault of Smith's) caused us no end of debugging problems, and we set out to find some way to avoid the new & HEAD-ACHE command the Apple seemed determined to

send to us. We found two solutions to the problem. First, use of the & should be restricted to flagging commands which would be gibberish to Applesoft in the absence of a preceding &. Forgetting & still causes the program to crash, but it crashes at the appropriate line number, so the error can be easily found and fixed. Second, when Applesoft commands are themselves modified, the modifying subroutine should be executed whenever that command is encountered, without requiring the presence of the & to signal something new. In this article we will show this second solution can be implemented, on all types of Apple II systems.

It is usually relatively easy to change a BASIC command if Applesoft resides in RAM memory. This occurs in one of two ways: either Applesoft has been loaded onto a 16K memory expansion card, or Ramcard, or Applesoft IIa has been loaded, usually from cassette tape, into the Apple's RAM memory. Ramcard Applesoft is identical to the Applesoft stored in ROM on the Apple II PLUS and on Apple's ROM Applesoft Board. Applesoft IIa is different from these, and we will save discussion of it until later.

If you do use Ramcard Applesoft, then the & approach to modification of BASIC commands is unnecessarily slow along with being very space inefficient. Our labs use the Apple Language System, which includes a Ramcard. By modifying the GOTO code directly, we save 60 of the 67 bytes required for Smith's labelled GOTO/GOSUB program, and do everything his program does, except that our version does not suffer from the one minor error in his (and any ROM Applesoft) version of that routine (details later). Unfortunately, our experience with Apple's rendition of PASCAL (which should not be taken as generalizing to other versions of PASCAL), was quite negative. Partially based on this, other labs in the Department in which we work decided to save their money, and did not buy the Language System or any other RAM memory expansion board. Thus, our programs would not run on their machines, and their &-laden programs would not run on ours. To maintain compatibility of systems, while avoiding the headaches inherent in &-flagged command modifications, we were forced to develop an inter-



preter for ROM Applesoft which works in much the same way as an &-interpreter would, except that it does not require the &.

### How The Applesoft Interpreter Works

Rather than storing the literal characters of BASIC commands, and of many other key words and symbols, Applesoft represents these internally in a tokenized format. Each key word is replaced by a number, between \$80 and \$EA (see the *Applesoft Reference Manual*, p. 121), and can thus be stored in a single byte of memory. This is an extremely space efficient storage system. However, Applesoft now requires an interpreter to decode these tokens, in order to act on the commands, or to evaluate the functions which they represent. All key words are first tokenized, then interpreted, whether the machine is in Immediate Mode, responding to commands as you type them in, or in Deferred Mode, running a program. Our interpreter mimics the Applesoft interpreter. For this reason, and also because we think it would be helpful for anyone writing modifications to Applesoft, we will describe the behavior of the Applesoft interpreter in some detail.

### The Flow Of Control

The Applesoft interpreter starts at location \$D805. We cannot list this copyrighted routine here, but you can see it for yourself if you jump to the MONITOR (via CALL -151) and then type in "D805L". Having the routine in front of you may clarify the flow of control described below.

This program begins, at \$D805, by determining whether the TRACE command is in effect (flagged by the contents of \$F2). If so, and if a Deferred Mode program is being run (flagged by contents of \$76), then, before each command is executed, the "#" sign is printed, followed by the line number. The first location following this print-out is \$D81D, which is branched to directly if the printing is not to be done.

At \$D81D, the CHRGET routine (\$B1-\$C8) is called. This subroutine fetches the next character of the program and sets the Zero flag if that character signals an "end of line," that is, if the character is a carriage return or a colon, ":". The routine then calls, via a JSR, the actual interpretation subroutine, which starts at \$D828. This returns immediately, via the RTS at \$D857, if an end of line is encountered. Otherwise, the program will return from this subroutine later, in a more indirect way. When this subroutine is returned from, the interpreter exits by jumping to a routine called NEWSTT (NEW STaTement) at \$D7D2, which will execute the next program statement, falling back into this interpreter in the process.

If CHRGET did not find an end of line, the

\$D828 subroutine expects to find a command of some sort. If the character fetched by CHRGET is not a token, the interpreter assumes the programmer intended a LET command (eg X=100) and jumps to the LET subroutine at \$DA46. The interpreter determines whether it has a token by subtracting \$80, the value of the smallest token, from the Accumulator (A), which holds the character. If A is still positive, we have a token. This token may represent a command (\$80 through \$BF), or it may be some non-command key word (\$C0 through \$EQ). Since we have already subtracted \$80 from A, we have a command only if A is less than \$40 ( $\$40 + \$80 = \$C0$ ), which is checked by a CMP (compare) instruction. If A is not less than \$40, we do not have a command, which is what should be here, so the interpreter jumps to \$D846, thence to \$DEC9, which produces a "? SYNTAX ERROR".

### The Command Table

If we are dealing with a command, the next job of the interpreter is to determine where to go to execute it. There is an address table, beginning at \$D000, (Applesoft IIa: \$0800) which contains this information. In this table, the starting address of every command, less 1, is stored in order of magnitude of the command's token. Thus the address of END, whose token is \$80, is stored first, from \$D000 to \$D001. FOR's token is next, and the address of the FOR routine, less 1, is stored from \$D002 to \$D003. Since \$80 was subtracted from A, A now stores a number between \$00 and \$3F. Double this number, by rotating A left, add it to \$D000, and you get the location of the two byte address of the command.

The addition is accomplished by indexing \$D001 and \$D000 with register Y, after Y is loaded with the doubled contents of A. The command's address, less 1, is then pushed onto the stack. When the next RTS is encountered, the program will "return" control to the last address on the stack, after adding 1 to that address. Thus, the next RTS we encounter will force a jump to the correct starting address of the command to be executed. The actual location of the interpreter's RTS is hidden. The final instruction of the interpreter is a JMP, rather than a JSR, to CHRGET, which will fetch the first character following the command. The RTS from CHRGET is the one which takes us to the command itself.

Note that the next address on the stack is the address of the routine which called this interpreter. We have already seen what happens when this one is returned to: the interpretation process stops and the program jumps to NEWSTT. As soon as the RTS at the end of the command we will execute is encountered, the program will effectively branch



to NEWSTT.

### What Happens When Applesoft Runs Into An &

The & symbol is tokenized in the same way that BASIC commands are tokenized, so, even though & is not a proper BASIC "command" (see the *Applesoft Reference Manual*), the interpreter will treat it as if it were one. The & address in the token lookup table is \$03F5 (less 1). That is, the CHRGET routine jumped to by the interpreter will return to location \$3F5 when it has fetched the next character following the &. At \$3F5, there are three free locations, which typically contain an instruction to jump to some other address, say \$0300, where the actual &-interpreter begins. The &-interpreter will then typically examine the contents of the accumulator, which holds the latest character fetched by CHRGET, and will operate on these in much the same manner as the Applesoft interpreter did for the character it got from CHRGET. There are various ways to accomplish this, but the effect will be to eventually find yet another address (the start of the command flagged by the &) and to force a JMP to that address. The return at the end of the subroutine jumped to will be a return back to the Applesoft Interpreter and, thence, to NEWSTT, as described above.

### How To Get Rid Of The & in ROM Applesoft

Our goal is to write a command token interpreter which will handle modified Applesoft commands. We are not worried about things flagged by the & which would not otherwise be treated as commands. We could extend the routine below so that it would handle these, but, given the implementation, (i.e. as a patch to CHRGET, which is very frequently called), this would noticeably slow down Applesoft, so we do not recommend it. Further, we could extend the interpreter so that it would handle tokens which are not commands, allowing modification of functions. Again, we do not recommend this. Applesoft provides a USR function with the explicit intention of allowing user defined functions. Our experience with functions is that they are not really modified at all. Rather, they are replaced by something quite different, which the user considers better.

In this case, we feel that the appropriate vehicle for replacement is the one deliberately built into Applesoft, i.e. USR. An &-based alternative, or an alternative using this routine, would be inappropriate as well as quite clumsy. Given these limited objectives, along with a desire (if only to ease our memory burden) to mimic the Applesoft token interpretation approach, the modification to ROM Applesoft in order to allow compatibility with RAM Applesoft without requiring ampersands to flag modified commands is surprisingly straight-

forward.

Nearly all of ROM Applesoft resides in ROM and so cannot be changed. One important routine, CHRGET, does not reside in ROM. CHRGET is loaded into memory whenever Applesoft is initialized (e.g. by the FP command if you have a disk). This is exactly the routine called by the Applesoft interpreter whenever it wants a new command. We modify CHRGET so that it jumps to a routine we call NEWGET, located at \$300, which will replace CHRGET. Along with doing everything CHRGET used to do, this routine checks whether the character it fetches is a token in a list of modified command tokens. If so, it executes that command before returning to the Applesoft interpreter.

All properly written Applesoft commands and programs will leave the TeXT PoiNtEr (TXTPTR) pointing to an end of line following the command, once the command has been executed. User written commands must conform to this as well. If they do, then, when NEWGET finally returns control to BASIC, it will pass the Applesoft interpreter an "end of line." This forces, as we have seen above, a branch to NEWSTT, which is just what we want in order to avoid confusion in Applesoft over what should be executed next.

### Further Notes

Here are a few further notes on the ROM Applesoft token interpreter subroutine listed at the end of this article, which may not otherwise be clear from the discussion above.

1. The CHRGET routine conceptually divides into two subsections. The first increments TXTPTR, to point to the next character. The next section is often called CHRGOT, and this is the routine which actually fetches the contents of the location pointed to by TXTPTR and sets up various internal flags. By loading JMP \$300 into the first three bytes of CHRGET, we effectively destroy the 6-byte segment of code which increments TXTPTR, and thus have to repeat this as the first 6 bytes of NEWGET. In the process, we free the three page 0 locations, \$B4, \$B5, \$B6, which follow the JMP \$300. These are used as temporary locations by NEWGET. The CHRGOT routine is completely unaffected by the jump, so we can still use it to actually fetch the character and to set the appropriate flags (eg. Zero).

2. CHRGET has no effect on registers X and Y. We will use both in searching the token table. If we do not return the original values of X and Y when we return from NEWGET, we will induce errors in the many routines which call CHRGET and which assume that this call will not affect these registers.

3. Variable FLAG checks whether or not a command is currently being executed. If FLAG is 0, we



are not in the midst of handling a command. If FLAG is not 0, then the user has typed in something like GOTO GOTO, which is a syntax error. Since Applesoft syntax is very clear on this point – all commands must be separated by ends of line (colon or carriage return) – we return “? SYNTAX ERROR” if we find a command token while executing a command. Since BASIC is not in direct control of program execution at this point, this program must do its own error checking.

4. Our lookup table differs in format from the one at \$D000. We also store jump locations, less 1, and will get to these via an RTS, as the Applesoft interpreter does. However our table also holds the target tokens themselves. The first and second locations of CMDTBL (CoMmanD TaBLe) hold the low and high bytes of the address of the subroutine which will execute the command whose token is stored in the third location. Similarly for the fourth and fifth locations (addresses) and the sixth location (token), and so on. The command table is stored in this version of the program at location \$0354. There is no need whatever to store it here. You can put it in any convenient place in memory, as long as you change the four places in the program which refer to the starting address of CMDTBL (before JMPGOT and after GOTONE).

5. End of CMDTBL is indicated by a zero value for the target token. We load the token into register Y at \$31D in order to test for end of table. If Y is zero, the command token is not one we are looking for, so we exit.

## The Labelled GOTO/GOSUB Example

### 1. Comments on the patch.

We use Smith's routine as an example partially because it was published in **COMPUTE!**, so you may be familiar with it, and partially because we have found it quite useful. The routine has been completely rewritten in three ways, once for ROM Applesoft, once for RAMCARD Applesoft, and once for Applesoft IIa, or TAPERAM. A more complete discussion of the logic of the routine is in Smith's article.

The effect of the patch is as follows: Taking X to mean any arithmetic expression or variable, (X may, but need not, be a literal number), then if the value of X is 1000, GOTO X will be treated by Applesoft in the same way as GOTO 1000. Similarly, GOSUB X will be treated as GOSUB 1000. Thus labels can be defined for subroutines (as all rational programming languages, including nearly all assemblers, allow) and for GOTO statements (reminiscent of FORTRAN's ASSIGN statement). If X is a *real* number, it is rounded down to the nearest integer, so be wary of arithmetic expressions.

This patch does not affect the behavior of ON...GOTO and ON...GOSUB, so these must still use line numbers rather than labels. However, you can replace these computed GOTO's in your code by computed GOTO's of a very different type, which are reminiscent of PASCAL's CASE handling. As an example, at the start of your program you might DIMension a matrix SELECT(20) and assign the values of 20 different line numbers to the values of SELECT(I). To GOTO these lines, you can compute the value of I, and then GOTO SELECT(I). With decent commenting on the intention and conditions of each choice of SELECT's line numbers (which is best done at the place in the program that the line numbers are actually assigned to SELECT's elements,) your program will probably be much more readable than one with an equally commented ON...GOTO statement. This is our experience with these two different forms of computed GOTO and GOSUB, and we have stopped using ON...GOTO completely.

## Spaghetti Structure

A different method of implementing a computed GOTO is ideal for making your program structure resemble a plate of spaghetti. If you change the value of X at various points in the program and repeatedly use X in GOTO X or GOSUB X statements, then, if you succeed in debugging your program, you will be able to amaze your friends and neighbors with your ability to produce unintelligible, yet functional, code. The labelled GOTO/GOSUB facility as presented here can be used to dramatically increase the readability of your program, or it can be abused to degrade the structure of your program. We strongly recommend that you assign line numbers to specific variables at the start of the program, use informative names for those variables, and never change their values once assigned.

### 2. The actual patches

The ROM Applesoft patch is given in the appended listing, directly after the lookup table. The program is virtually the same as Smith's, with a few more comments. Note that whether you use this program or Smith's, IF...THEN X, IF...GOTO X and IF...THEN GOTO X will not work properly. We have tried and tried, but cannot fix this flaw in a program of reasonable length. A statement of the form IF...THEN: GOTO X (or GOSUB X) will work correctly. There must be a colon between the THEN and the GOTO or the GOSUB.

The RAM Applesoft patch is much simpler. For either RAM Applesoft version, you do not need the subinterpreter. Instead, modify GOTO directly. GOTO is a subroutine of GOSUB, so this modifies both. Change the JSR \$DA0C (the LINE



number GETting subroutine) at \$D93E (Ramcard), or the equivalent command at \$1140 for Taperam to JSR \$300 (or wherever you wish to store this patch). This changes the first command of GOTO, forcing it to treat the patch as its line number getting subroutine. The patch consists of:

```
JSR FRMEVL
JSR GETADR
RTS.
```

The Ramcard locations of these routines are \$DD7B for FRMEVL, which evaluates the expression following the GOTO or GOSUB and deposits this in Applesoft's floating point accumulator, FAC,

and \$E752 for GETADR, which moves the contents of FAC to locations \$50 and \$51, where GOTO expects to find the line number to which to go. The locations for TAPERAM are \$157E, and \$1F49 for the two routines. That's all there is to it. Seven bytes of new code and, to top it off, IF...GOTO X works (IF...THEN X will not), as do IF...THEN GOTO X and IF...THEN GOSUB X though, to preserve program portability, you will probably want to include the colon, entering only IF...THEN: GOTO (or GOSUB) statements when using labels rather than literal line numbers.

*To use this routine, the Applesoft CHRGET routine at \$B1 must be modified. This may be accomplished from the monitor by writing B1:4C 00 03NB6:00 or by calling the initialization patch (included below) from BASIC.*

```
0028 0300      ;EQUATES:
0029 0300
0030 0300      SYNERR = $DEC9      ;GENERATE SYNTAX ERROR
0031 0300      TXTPTR = $B8
0032 0300      CHRGOT = $B7
0033 0300      XTEMP = $B4
0034 0300      YTEMP = $B5
0035 0300      FLAG = $B6          ;FLAG COMMAND IN PROGRESS
0036 0300
0037 0300      ;THE NEW CHRGET ROUTINE:
0038 0300
0039 0300 E6B8    NEWGET INC TXTPTR      ;INCREMENT LOW BYTE
0040 0302 D002      BNE GETCHR          ;GET NEXT CHARACTER
0041 0304 E6B9      INC TXTPTR+1        ;INCREMENT HIGH BYTE
0042 0306 20B700    GETCHR JSR CHRGOT    ;GET CHARACTER
0043 0309
0044 0309      ;THE SUB-INTERPRETER:
0045 0309
0046 0309 C9C0      CMP #$C0          ;IS IT A NON-COMMAND TOKEN?
0047 030B B025      BCS OUT            ;YES, GO
0048 030D 86B4      STX XTEMP          ;ELSE, SAVE X
0049 030F 84B5      STY YTEMP          ;AND Y
0050 0311 A8        TAY                ;TEST N FLAG
0051 0312 1016      BPL JMPGOT         ;NOT A TOKEN?, GO
0052 0314 A4B6      LDY FLAG           ;COMMAND IN PROGRESS?
0053 0316 D01D      BNE ERRORT        ;YES, ERROR
0054 0318 A2FF      LDX #$FF          ;NO, SET X FOR TABLE LOOKUP
0055 031A E8        NPTCMD INX         ;POINT X AT NEXT COMMAND TOKEN
0056 031B E8        INX
0057 031C E8        INX
0058 031D BC5403    LDY CMDTBL,X       ;GET TOKEN
0059 0320 F008      BEQ JMPGOT         ;END OF TABLE? GO
0060 0322 DD5403    CMP CMDTBL,X       ;ELSE, COMPARE TO TABLE TOKEN
0061 0325 D0F3      BNE NPTCMD        ;NO MATCH, GET NEXT COMMAND
0062 0327 203803    JSR GOTONE         ;ELSE SET UP FORCED JUMP
0063 032A A200      JMPGOT LDX #0
0064 032C 86B6      STX FLAG           ;CLEAR FLAG
0065 032E A6B4      LDX XTEMP         ;RECOVER X
0066 0330 A4B5      LDY YTEMP         ;RECOVER Y
```



```

0067 0332 4CB700 OUT      JMP CHRGOT      ;GET CHAR AT TXTPTR
0068 0335          ; .                  AND RETURN TO APPLESOFT
0069 0335
0070 0335 4CC9DE ERRORT JMP SYNERR      ;DO SYNTAX ERROR
0071 0338
0072 0338          ;SET UP FORCED JMP VIA RTS AND SET FLAG
0073 0338
0074 0338 85B6      GOTONE STA FLAG      ;FLAG COMMAND IN PROGRESS
0075 033A BD5303      LDA CMDTBL-1,X ;GET HIGH BYTE
0076 033D 48          PHA              ;AND DEPOSIT ON STACK
0077 033E BD5203      LDA CMDTBL-2,X ;GET LOW BYTE
0078 0341 48          PHA              ;AND DEPOSIT ON STACK
0079 0342 60          RTS              ;EXECUTE USER PATCH
0080 0343
0081 0343          ;INITIALIZATION PATCH:
0082 0343          ;A CALL 835 FROM BASIC WILL INITIALIZE
0083 0343          ;THE ROM SUB-INTERPRETER.
0084 0343
0085 0343 A94C      INIT   LDA #$4C      ;LOAD A 'JMP' AND
0086 0345 85B1      STA $B1      ;STORE AT CHRGOT
0087 0347 A900      LDA #<NEWGET ;LOW BYTE OF INTERPRETER
0088 0349 85B2      STA $B2
0089 034B A903      LDA #>NEWGET ;HIGH BYTE
0090 034D 85B3      STA $B3
0091 034F A900      LDA #0        ;CLEAR THE
0092 0351 85B6      STA FLAG      ;COMMAND IN PROGRESS FLAG.
0093 0353 60          RTS          ;RETURN TO BASIC
0094 0354
0095 0354          ;COMMAND TABLE:
0096 0354          ;COMMANDS AND THEIR PATCH ADDRESSES
0097 0354          ;ARE STORED TOGETHER IN LOW-BYTE, HIGH BYTE,
0098 0354          ;COMMAND TOKEN ORDER. THE LAST THREE BYTES
0099 0354          ;OF THE TABLE MUST BE ZEROS.
0100 0354
0101 0354 5C03      CMDTBL .WOR GOSUB-1 ;GOSUB PATCH ADDRESS
0102 0356 B0          .BYT $B0      ;GOSUB TOKEN
0103 0357 7603      .WOR GOTO-1     ;GOTO PATCH ADDRESS
0104 0359 AB        .BYT $AB       ;GOTO TOKEN
0105 035A 00        .BYT 0,0,0     ;END OF TABLE
0105 035B 00
0105 035C 00
0106 035D
0107 035D          ;AS AN EXAMPLE OF USING THE ROM TOKEN INTERPRETER,
0108 035D          ;WE INCLUDE A LISTING OF A PATCH THAT PROVIDES
0109 035D          ;LABELLED GOSUBS AND GOTOS IN APPLESOFT BASIC.
0110 035D          ;THIS CODE IS FROM M.R. SMITH (COMPUTE, 12, 1981).
0111 035D          ;FOR THE GOSUB PATCH, IT IS EFFECTIVELY A
0112 035D          ;RELOCATION OF THE INITIAL PORTION OF THE
0113 035D          ;APPLESOFT GOSUB CODE. THIS ENABLES A
0114 035D          ;MODIFICATION OF THE SECTION OF CODE THAT
0115 035D          ;JUMPS TO THE APPLESOFT GOTO CODE, WHERE
0116 035D          ;THE EFFECTIVE CHANGE IS MADE.
0117 035D
0118 035D          ;APPLESOFT POINTERS AND ROUTINES:
0119 035D          CURLIN = $75      ;CURRENT LINE NUMBER
0120 035D          NGOSUB = $D7D2   ;NORMAL GOSUB

```



```

0121 035D      NGOTO  = $D941
      ;NORMAL GOTO
0122 035D      FRMEVL = $DD7B
      ;EVALUATE EXPRESSION AT TXTPTR
0123 035D      STACK  = $D3D6
      ;CHECK ON STACK POINTER
0124 035D      GETADR = $E752
      ;TRANSFER FAC TO LINNUM
0125 035D
0126 035D      ;GOSUB PATCH:
0127 035D
0128 035D A903  GOSUB  LDA #3
      ;NORMAL GOSUB RELOCATED
0129 035F 20D6D3 JSR STACK
      ;FROM $D921
0130 0362 A5B9    LDA TXTPTR+1
      ;STORE TXTPTR
0131 0364 48      PHA
      ;ON STACK
0132 0365 A5B8    LDA TXTPTR
0133 0367 48      PHA
0134 0368 A576    LDA CURLIN+1
      ;STORE CURRENT LINE NUMBER
0135 036A 48      PHA
      ;ON STACK
0136 036B A575    LDA CURLIN
0137 036D 48      PHA
0138 036E A9B0    LDA #$B0
      ;MARK GOSUB
0139 0370 48      PHA
      ;ON STACK
0140 0371 207703  JSR GOTO
      ;DO A MODIFIED GOTO
0141 0374 4CD2D7  JMP NGOSUB
      ;FINISH NORMAL GOSUB
0142 0377
0143 0377      ;GOTO PATCH:
0144 0377      ;THIS IS WHERE THE EFFECTIVE CHANGE OCCURS.
0145 0377      ;BY REDIRECTING THE EVALUATION OF WHAT FOLLOWS
0146 0377      ;THE GOTO OR GOSUB TOKEN TO FRMEVL (WHICH
0147 0377      ;EVALUATES THE EXPRESSION FOLLOWING THE TOKEN),
0148 0377      ;AND THEN TRANSFERING THE RESULT FROM FAC
0149 0377      ; (THE FLOATING POINT ACCUMULATOR) TO LINNUM
0150 0377      ; (WHERE THE REMAINDER OF THE ACTUAL GOTO
0151 0377      ;ROUTINE EXPECTS TO FIND THE LINE NUMBER),
0152 0377      ;LABEL AND EXPRESSION EVALUATION FOLLOWING
0153 0377      ;GOTO OR GOSUB IS EFFECTED.
0154 0377
0155 0377 200003  GOTO  JSR NEWGET      ;GET NEXT CHARACTER
0156 037A 207BDD  JSR FRMEVL      ;EVALUATE EXPRESSION
0157 037D 2052E7  JSR GETADR      ;TRANSFER FAC TO LINNUM
0158 0380 4C41D9  JMP NGOTO      ;FINISH NORMAL GOTO
0159 0383
0160 0383      .END                ;END ASSEMBLY

```

## ...PET/CBM/VIC? SEE SKYLES...

**PET owners everywhere sing**  
*♪ Thanks for the Memories ♪*  
**to good old Bob Skyles**

... they should ... because Bob Skyles is the only complete source for memory boards for *any* PET ever sold. Old Bob won't forget you.

And the Skyles memory systems have the highest quality control of any computer product ever. Over 100 million bits of Skyles memory boards are already in the field. First quality static and dynamic RAMS, solid soldered on first quality glass epoxy. That is why they are **guaranteed**—in spite of the new lower prices—for a full two years.

The boards, inside the PET/CBM, install in minutes without special tools or equipment ... just a screwdriver.

Because of our new dynamic memory design, and to celebrate old Bob's 30<sup>th</sup> birthday, here are the smashing new prices:

8K Memory System	orig. \$250.00	now \$200.00	Save \$ 50.00
16K Memory System	orig. \$450.00	now \$300.00	Save \$150.00
24K Memory System	orig. \$650.00	now \$400.00	Save \$250.00

... For any PET ever made. When ordering, just describe your PET by model number and indicate the amount and type (or brand) of memory currently in the unit.

Shipping and Handling ... (USA/Canada) \$3.50 (Europe/Asia) \$15.00

California residents must add 6% / 6 1/2 % sales tax, as required.

Visa/Mastercard orders: call tollfree (800) 227-9998 (except California). California orders: please call (415) 965-1735.



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

## ...PET/CBM/VIC? SEE SKYLES...



*"Garbage collection" refers to the long delays which can occur while the computer rearranges strings. This article shows how to avoid these delays when you are working with significant numbers of strings. The technique here is most useful for PET/CBM owners who have older machines. Newer machines with BASIC 4.0 avoid these problems.*

# Screen Input On The PET

Elizabeth Deal  
Malvern, PA

Using the information placed on the screen as a source of *input* seems like a contradiction in terms. Why should one bother inputting something when one already knows what it is? I stumbled upon one good reason and worked with it to a happy ending: no garbage collection delays.

This article runs through a series of small experiments. Both tests and conclusions are based on work in the Upgrade PET. Users of original ROMs and pre-Fat 40 BASIC 4, 40 and 80 column systems are invited to try the tests. I suspect that the results will be the same. However, simple as it is, I just don't know how the whole thing will behave in another PET. The concept of null input might be handled differently and, together with POKEing two system locations, it might crash the PET. This might mean it will need to be reset. There is no way of telling until you join the fun.

Screen input, as described in this article, is most valuable in the systems prior to 4.0 BASIC as it shows yet another way of minimizing character string handling in our quest towards a garbageless PET. My method is limited, however, to only certain applications. The entire problem was explained and a more general procedure was proposed by Jim Butterfield in **COMPUTE!**, September, 1981, #10. BASIC 4 users, of course, don't suffer from these problems.

The reason I became interested in screen input is twofold. First, the suggestion appears in the POWER™ manual, and it made me curious about the method's utility. [POWER is a chip which adds several commands to BASIC. It is sold by Professional Software.] Secondly, I have a very nice disk utility which displays all the sorts of data about the contents of the floppy. The program carefully displays that information by use of ordinary PRINT statements (no reverse field, no cursor controls, nothing,

just plain letters). I needed to lift this data, put it into variables, and use it for other things, not excluding a sorted listing. It seemed like a simple task, until a seven minute long garbage collection zapped me during testing.

The garbage collection occurred because, although the program originally concatenated (added) pieces of strings by use of PRINT statements in a loop, as in

```
PRINT#4,CHR$(X+48);
```

I had to change the code to

```
V$(3,I)=V$(3,I)+CHR$(X+48)
```

So the concatenation would run its course eight times before I could touch the value. (Putting each character into an array is out of the question). This invited trouble. It didn't really happen, as the number of strings wasn't that large. But, when I let the program do the work several hundred times while simulating a small PET and a large disk drive, the PET got clogged up with strings and produced an annoying, slow motion spectacle, eventually ending in the interminable pause for housekeeping. That did it.

Clearly, there are many approaches to such a simple task, but the challenging aspect in this case was that the screen already contained neatly formatted data in the exact form I wanted. If I could only, somehow, pick up these fields and stuff them into the elements of a string array V\$(record,field) I'd be home free. I couldn't resist trying the screen input idea. It just seemed right for the task.

## Testing

Before actually using a new scheme in a program it pays to understand how the process really works, remembering Murphy's Law: "nothing is ever as simple as it looks." Let's run through experiments that seem to provide several definitive answers.

A note to non-Upgrade BASIC users: I don't know what these tests will do to you. Several conversion values are provided, but if your system works differently, it is up to you to discover it. The amount of typing is insignificant until the very end. By then you will know enough to avoid serious trouble.

(1) Type a line: 1050 OPEN3,3:INPUT#3:CLOSE3. Clear the screen and RUN it. The only redeeming feature of this result is that we haven't killed the PET. The error tells us that some bad values have landed in the OS area. Give up or think. In the remaining exercises you do not need to clear the screen. As a matter of fact, it's more enlightening if you don't.

(2) Add these lines:

```
100 PRINT"ONE";:GOSUB1000
200 ??:PRINT A$,LEN(A$);ASC(A$+CHR$(0));
    VAL(A$)
210 END
```



1000 :  
1500 RETURN

And RUN it. Well, it doesn't do anything. A\$ is a useless string, a long chain of blanks, not exactly our intention.

(3) Let's position the cursor over the string. Move it back to "0" on "ONE" by inserting three left cursors within the quotes after "E". RUN it. There is hope, but note the unacceptable length of the string.

(4) We can limit PET's wish for characters by printing a comma after the "ONE". But that's the last thing I need in my neat display. Let's convince the PET, instead, that its screen width is not 40 characters (or 80 in 80xx), but only three. Butterfield's memory map indicates that information is held in 213. I don't know what it is in Original PETs, those maps were never printed in **COMPUTE!** (Perhaps an equivalent of 213 does not exist. If anyone would like to contribute an accurate cross-reference type map for all these systems, please send one in). Type two lines:

```
1020 POKE213,3 to squeeze the screen width
1080 POKE213,39 to restore it back.
(PET counts from zero)
```

This works. Incidentally, line 1080 may not be needed, but out of overabundance of caution I'll keep it. PET's self preservation instinct seems to restore 213 for subsequent screen work. Query whether 80-column window facility might be useful.

(5) We're not out of the woods yet. What will happen if "ONE" were followed by a string of blanks, a typical sight in a left adjusted alphabetic information? Modify two lines to be:

```
100 ?LEFT$("ONE" + "10 sp",10) + "10 left";:
GOSUB1000
```

```
1020 POKE213,10
```

RUN it. Trailing blanks are handled correctly.

(6) What about leading blanks you might see in a right-justified number? Change one line:

```
100 ?RIGHT$("10 sp" + "472",10) + "10 left";:
GOSUB1000
```

and RUN it. Oops! The value is intact, but PET stripped the leading spaces, as it always does in INPUT. This result may be satisfactory for many applications, too sloppy for others. Let's handle it.

(7) One fix might involve printing a phony character in the first position. But then, if we wanted to use the value in a numeric variable, we'd have to strip the character. Instead, we'll use a harmless quote by typing 100 PRINT CHR\$(34) + RIGHT\$ ... etc. and increase the screen width to 11 in line 1020. Unfortunately this places a character on the screen. Unavoidable. RUN it. Not bad. The leading blanks are there, the value is all right, but the original string is useless with the in-quote cursors.

In fact, any formatting command on the same line, i.e. prior to going into 1000, will print this

way. TAB(x) will yield x right cursors. That's the way PET is built. We could POKE 205 to disable the quote mode, but that gets messy in some applications. We could close the quote, but that increases the space penalty on the line by another character. A less limiting way is needed.

(8) Remember that all we really want is to move the cursor left. PET doesn't care how. The memory map indicates the current cursor position is held in 198 (Original 226?). To move the cursor into the first position we'll put a zero there. In line 100 delete 10 left cursors and its surrounding quotes. Type 1005 POKE198,0 to put cursor in the first position, and 1040 POKE 198,11 so that a subsequent field prints in a correct position. RUN it. It really works now. We can handle leading and trailing blanks and anything in between. Right?

(9) Not quite. How about a null string? It can happen. If a condition is true you might be printing a "\*", if false you'll print nothing and use TAB to position to the next item, or skip line if it is at the end. Our 1000 subroutine can't predict the future, and will attempt to process a null string with all its inherent troubles. Let's see what happens. Type 100 PRINT"";:GOSUB1000 and don't type, but imagine, 110 PRINT TAB(10)"whatever". RUN it. Very nasty. There is actually more wrong than the eye can see. If the PET is alive you have just learned, for instance, the easiest way of causing active files to vanish into thin air (look at the table in \$0251-026F on the Upgrade and 4.0 systems). Tragic results, to say the least.

(10) A quote may help. Type 100 PRINT CHR\$(34) + "";:GOSUB1000. Once again, this gives a correct result, but subsequent TAB or cursor characters will cause trouble. This was not very important with numbers. It's vital that it doesn't happen here.

(11) While we correct this mishap, let's make the routine a bit more general for further testing. Type NEW and the code in lines 10-260, adjusting 198, 213 and screen width of 39 to your system. P1 is the first position of the field. P2 is the final position after a string, null or otherwise, is printed. You can try the code on various things within the quotes in line 30. When done, type two quotes there (" ") and convince yourself that it can't work. Then remove REM from line 190 and try it. This works. Replace REM in 190 and remove REM in 200. This also works with some space penalty.

Either method should work for a particular application – the choice is a function of further use for the field. Lines 190 or 200 simply check if the cursor has moved. If not, the subroutine detects a null string and makes the needed adjustments by faking a different screen width. In case of 200, it must also reset 198, since the PRINT statement



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

## WHO CAN USE ASERT?

libraries  
personnel departments  
dating services



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**

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. [www.abcomputers.com](http://www.abcomputers.com)



changes it. Unless Murphy has other plans, this completes the tests.

You should be able to guess now what would happen if the PRINT statement contained cursor controls, forced quotes, commas, colons, a clear screen command, or reverse fields (both at the beginning and the end of the space) within the quotes. All the information needed to answer the questions is here and in the known facts of an ordinary INPUT statement. If you can't guess you may want to try these things by repeating step 11. We don't need those answers for the simple application below, but we may need them for other tasks.

### The Problem Solved

Since we now understand some of the PET's rules in this game we can try a slightly more ambitious project: correctly picking up multifield records, our original intent. A routine in lines 510-1060 is used. By way of illustration, we use unformatted data from DATA lines. Two PRINT statements format the data on the screen, avoiding garbage collection problems in the string construction. (The strings are actually constructed in high memory, but that does not contribute to the garbage.) PRINT C\$ does not format the data, therefore a null string will result on output/input, a slight complication if you want to leave the program intact. The shifted-space method is used. I leave it as an exercise to the reader to try null strings elsewhere on a line, while positioning subsequent fields correctly. An example is given in the two REM lines, which introduce another variable. Admittedly these complications are unnecessary, they are included to make the task harder.

As each part of a line is printed, the subroutine in 1010 places its image into an element of an array A\$(line,field). Note that in some applications you may not even have to go after each field. You may wish to pick up one, two, three, or all four fields in a single procedure by placing GOSUB 1010 where you want it.

There are no new concepts in the code. The key variables are:

A\$(I,J)	destination array
I	record or screen line number
J	field number, counted in subroutine from left to right
P(J-1)	cursor position before the field is printed
P(J)	cursor position after the field is printed
W	screen width varying with P(J)
WS	system screen width address where we adjust screen size
CP	system address of cursor position on a line
PS	forces tabbing on a null string if such must exist. PS default is 1.
P	flags when we're dealing with a null string.

The two P arrays could have been coded as single variables. They are provided in array form, for we

may need those values for something else.

When you run this demo program, you will see that the method works and is not all that complicated. Here, we have purposely set up certain roadblocks to see how far we can push the PET.

There are more uses of screen input, just as there are more aspects to the method. We just scratched the surface. But once we have the basics we can go on to bigger things.

Screen input is useful for working with strings without the usual penalty. I thank Jim Butterfield for hinting at this in the POWER documentation and, more importantly, for offering a valuable warning about trouble spots.

```

10 REM SCREEN INPUT ELIZABETH DEAL
20 P1=POS(0)
30 PRINT"ANYTHING";:GOSUB180
40 PRINT"":PRINT
50 PRINTA$,LEN(A$)ASC(A$+CHR$(0))
60 END
180 P2=POS(0):P=0
190 :REM IFP2=P1THENP=1
200 :REM IFP2=P1THENP=1:PRINTCHR$(160);
210 POKE198,P1
220 POKE213,P2+P
230 OPEN3,3:INPUT#3,A$:CLOSE3
240 POKE213,39
250 POKE198,P2+P
260 RETURN
500 :
510 W=39:WS=213:CP=198
520 R=3:NV=5:DIM A$(R,NV),P(NV)
530 BL$="":PF$=CHR$(160):PS=1
540 FORI=1TOR:J=0:READ A$,B,C$
550 : REM PRINT""::IFI=2THENPRINT"*";
560 : REM PS=2:GOSUB1010
570 PRINT LEFT$(A$+BL$,8);:GOSUB1010
580 PRINT PF$+RIGHT$(BL$+STR$(B),6)
590 PRINT C$;:PS=1:GOSUB1010
600 PRINT:NEXTI:PRINT
610 FORI=1TOR:FORJ=1TONV
620 PRINTA$(I,J);
630 NEXTJ:PRINT:NEXTI:PRINT:END
640 DATA FIRST,12345,TEXT 1
650 DATA SECOND,12,TEXT 11
660 DATA THIRD,,,
1000 :
1010 J=J+1:P(J)=POS(0):P=0
1020 IFP(J)=P(J-1)THENP=1:PRINT PF$;
1030 POKECP,P(J-1):POKEWS,P(J)+P*PS
1040 P(J)=P(J)+P*PS
1050 OPEN3,3:INPUT#3,A$(I,J):CLOSE3
1060 POKEWS,W:POKECP,P(J):RETURN ©

```





# the retailer™

cash register program/inventory management  
on the Commodore CBM

The **retailer™** contains the following programs:

## Cash Register

- Records sales transactions
- Prints register tape or invoice
- Automatically computes discounts and/or sales tax
- Computes change due
- Prints daily sales summaries

## Inventory Manager

- Creates and maintains complete information on every item in inventory (over 3000 items)
- Adds, changes, deletes inventory on command
- Automatically updates inventory from register transactions
- Maintains stock balance, records dollar amounts sold as well as cost of items sold
- Prints price tags

## Report Program

Reports produced:

- Listing of:
  - All Inventory items
  - Items sold in past week
  - Items on order
  - Items on hand by season
  - Items with zero stock balance
- Dollar listing by stock category of:
  - Weekly sales
  - Cost of weekly sales
  - Merchandise on hand
  - Merchandise on order
  - Merchandise at mark down
  - % sales to total sales
  - % stock to total stock
  - % gross profit
  - % stock at mark down
- Daily summary of sales by month

available now  
**\$650.00**

Dealer inquiries invited



RETCOM Systems, Inc., 61-B Mountain Blvd., Warren, New Jersey 07060 (201) 561-3112

[www.commodore.ca](http://www.commodore.ca)



# Extra Colors For Atari Through Artifacting

Bill Mohn  
Danville, CA

How can you get up to four colors in GRAPHICS 8? The technique is called "artifacting"; it makes use of an otherwise unintended characteristic of a color display tube.

The highest resolution mode for the Atari is GRAPHICS 8. This mode includes 51,200 pixels (picture elements) arranged in a rectangular area extending 320 units horizontally and 160 units vertically. Unlike modes 2 through 7, GRAPHICS 8 normally allows only one color at two brightnesses. The background hue and luminance is determined by Color Register 2. "SETCOLOR 2,hue,lum" establishes Color Register 2. If left unset, it defaults to dark blue.

Color Register 1, which is set by SETCOLOR 1,hue,lum is used only to determine the luminance of graphics points. The hue of graphics points is determined by Color Register 2. A point is put on the screen by BASIC's PLOT or DRAWTO commands. To be effective, a COLOR 1 must precede the first PLOT.

In order to understand artifacting, look closely at a color television screen. You will see closely spaced vertical stripes of red, green and blue phosphor elements. If these are illuminated equally, a white picture is produced. Unequal illumination produces all of the other colors.

## GRAPHICS 7 Versus 8

A pixel in GRAPHICS 8 is the size of one half of one set of three color phosphor elements. If Color Register 1 luminance is relatively high and a graphics point is plotted at an odd valued horizontal coordinate, only the blue phosphor will be lighted. If a shape is drawn, taking care to use only odd values for X, the entire shape will be blue.

On the other hand, if only even values of X are chosen, the red and green phosphors will be selected. The proportion of these will depend on the background hue (Color Register 2) and the resulting "even-only" figure can range from red

through brown to green. If both even and odd points are plotted, the background hue will result at a luminance specified by Color Register 1. Of course if neither even nor odd points are plotted, the result will be background hue and luminance. An interesting side effect occurs when an "even figure" overlaps an "odd figure" — the area of overlap will be clearly visible.

GRAPHICS 7 has pixels four times as large as GRAPHICS 8. That is, they are twice as wide and twice as high. Since artifacting requires skipping half the points in the horizontal direction, the resolution in that direction with GRAPHICS 8 approximately equals that of GRAPHICS 7. However, the *vertical* resolution of GRAPHICS 8 is twice that of GRAPHICS 7.

The program here is a simple demonstration of these principles. It first uses GRAPHICS 8 to draw three overlapping disks, the upper left using odd points only, the upper right using even points only, and the lower using all points. Note that the lower disk does have better horizontal resolution than the other two. After a delay, the program plots three similar figures in GRAPHICS 7. You will see that these disks have horizontal resolution equal to and vertical resolution worse than the previous upper disks.

You may explore the variety of colors possible with artifacting by changing the two SETCOLOR statements. Statements 4000 through 4120 may be inserted to cause all 960 possible SETCOLOR combinations to be displayed. This will step through all combinations as long as START is depressed.

## Notes On The Sample Program

- 1000-1200 Make an array of Y-values corresponding to X-values for circles.
- 1300-1500 Setup for the first display in GRAPHICS 8.
- 1600-1900 Draw upper-left figure using odd points only.
- 2000-2300 Draw upper-right figure using even points only.
- 3000-3300 Draw lower figure using even and odd points.
- 3400 A delay loop to allow viewing the first display.
- 5000-5600 Setup and draw first figure in GRAPHICS 7.
- 5700-7300 Draw second and third figures.
- 9000-9100 Another delay loop followed by a return to repeat.
- 4000-4120 Three nested loops stepping through all possible hues and luminances.
- 4060 The "arrow" is entered by typing ESC, CTRL, CLEAR
- 4090 This loops until either START, SELECT, or OPTION is depressed.

```

100 REM *****
200 REM *
300 REM * DEMONSTRATION OF
400 REM * ARTIFACTING IN GRAPHICS 8 *
500 REM *
600 REM * BY BILL MOHN
700 REM *
```



```

800 REM *
900 REM *****
1000 DIM Y(79)
1050 ? "SHORT DELAY WHILE CALCULATING"
1060 ? "COORDINATES FOR CIRCLE....."
1100 FOR I=1 TO 79
1200 Y(I)=SQR(1600-(40-I)*(40-I)):NEXT I

1300 GRAPHICS 8:COLOR 1
1400 SETCOLOR 1,0,14:SETCOLOR 2,10,0
1500 ? "GRAPHICS MODE 8 WITH 'ARTIFACTIN G'"
1600 FOR X=1 TO 79 STEP 2
1700 PLOT 80+X,45-Y(X)
1800 DRAWTO 80+X,45+Y(X)
1900 NEXT X
2000 FOR X=2 TO 78 STEP 2
2100 PLOT 150+X,45-Y(X)
2200 DRAWTO 150+X,45+Y(X)
2300 NEXT X
3000 FOR X=1 TO 79
3100 PLOT 120+X,110-Y(X)
3200 DRAWTO 120+X,110+Y(X)
3300 NEXT X
3400 FOR DLY=1 TO 1000:NEXT DLY
4000 REM INSERT FOR AUTOMATIC SEQUENCING

4010 FOR HUE=0 TO 15
4020 FOR LUM2=0 TO 14 STEP 2
4030 SETCOLOR 2,HUE,LUM2
4040 FOR LUM1=0 TO 14 STEP 2
4050 SETCOLOR 1,0,LUM1
4060 ? " "; "SETCOLOR 2, "; HUE; ", "; LUM2
4070 ? "SETCOLOR 1,0, "; LUM1
4080 FOR DLY=1 TO 50:NEXT DLY
4090 IF PEEK(53279)=7 THEN 4090
4100 NEXT LUM1
4110 NEXT LUM2
4120 NEXT HUE
5000 GRAPHICS 7
5100 ? "GRAPHICS MODE 7"
5200 COLOR 1
5300 FOR X=1 TO 79 STEP 2
5400 PLOT 40+X/2,23-Y(X)/2
5500 DRAWTO 40+X/2,23+Y(X)/2
5600 NEXT X
5700 COLOR 2
6000 FOR X=1 TO 79 STEP 2
6100 PLOT 75+X/2,23-Y(X)/2
6200 DRAWTO 75+X/2,23+Y(X)/2
6300 NEXT X
6400 COLOR 3
7000 FOR X=1 TO 79 STEP 2
7100 PLOT 60+X/2,55-Y(X)/2
7200 DRAWTO 60+X/2,55+Y(X)/2
7300 NEXT X
9000 FOR DLY=1 TO 1000:NEXT DLY
9100 GOTO 1300

```



**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 station! 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.

<b>Atari Microworld</b>	Atari 400 and 800
32K Cassette	\$19.95
Atari 400 and 800	
32K Diskette	\$22.95
<b>TRS-80 Microworld</b>	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

<input type="checkbox"/> MICROWORLD	\$	
SHIPPING		1.00
TOTAL	\$	
<input type="checkbox"/> TRS-80 <input type="checkbox"/> ATARI		
<input type="checkbox"/> CASSETTE <input type="checkbox"/> DISKETTE		
NAME		
STREET		
CITY	STATE	ZIP
<input type="checkbox"/> CHECK	<input type="checkbox"/> VISA	<input type="checkbox"/> MASTERCARD
#	EXP. DATE	

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



# Insight: Atari

Bill Wilkinson  
Optimized Systems Software  
Cupertino, CA

The major program for this month is perhaps the most exciting one to appear in this column to date. We will take advantage of Atari's modular software construction to define a set of *soft keys*, a concept that is marketed for real \$\$\$ on some machines. The most obvious use of soft keys is in writing BASIC programs. Even with the abbreviations allowed by BASIC, wouldn't it be convenient to be able to use a single keystroke to get a disk directory listing? And, of course, when programming in assembly language there are certain character combinations that are repeated often enough to justify the use of soft keys (e.g., "Y" or ".BYTE").

The techniques presented in the soft key program include how to "steal" the system's default I/O devices and adapt them for your own purposes. It might be worth your while to re-read my column on adding the "M:" driver, (**COMPUTE!**, January, 1982, #20, pg. 120) since I will be assuming your knowledge of some of the points made there.

For the BASIC user, the soft keys can be made truly "soft" — even to the point of allowing a running BASIC program to change the definition of what a soft key means. And, of course, there will be the usual set of tidbits for those who don't feel up to tackling the soft keys project.

## An Announcement

As most of you probably know, magazine articles and columns are written months before they actually appear. As I write this in mid-February, my company (Optimized Systems Software, Inc.) and **COMPUTE!** are frantically engaged in getting a new book ready for publication. *Inside Atari DOS* will presumably have made its appearance by the time you read this. Now, for the first time, Atari users will have access to the listing of the File Manager System of Atari DOS 2.0S, the current version of

Atari DOS. Besides the listings, there is a complete description of each major subroutine, complete with entry and exit parameters and error conditions.

The book is not complete in and of itself: you would still need Atari's listings of the OS ROMs and DUP to have access to all of the DOS secrets. But this book will tie together many loose ends.

Let me leave you with one caveat (don't I always?): the book assumes that the reader has at least a working knowledge of 6502 assembly language. The book is of most value if you would like to see how such a complex organism as a DOS is built.

The terminology "soft" keys refers to keyboard keys that may change "meaning" as desired by the user. The Atari keyboard keys which we will make "soft" include the characters "control-A" through "control-Z" (that is, what are normally graphic characters produced by holding down the control key while hitting one of the letter keys).

The keys will be made soft in a very flexible fashion: each of the 26 keystrokes may be defined in such a way that entering one of them will "fool" the Atari OS (and hence BASIC, etc.) into thinking that a sequence of one or more ordinary keys have been depressed. The phrase "one or more" is literal: there is no effective limit on the number of characters a soft key may represent.

As this program is written, there are some limitations. Only characters with an ATASCII value of 1 to 127 decimal (\$01 to \$7F hex) may be placed in the string. The CR (RETURN or End-Of-Line) character is thus not permitted (since its value is \$9B hex, 155 decimal). Since lack of CR seems to me to be a major flaw, each zero (\$00) byte in the string is converted so that OS sees a CR instead.



The reason that only the values from 0 to 127 are acceptable is that a byte with its most significant bit (MSB, \$80 or 128) turned on is our signal that this byte is to be the last character in the soft key string. Obviously, you can rewrite this part, if you desire, so that some other means is used to designate the end of string (a preceding length byte or trailing zero byte are obvious alternatives). However, the method chosen is simple and seems adequate for most purposes.

### For the BASIC user, the soft keys can be made truly 'soft'...

One more note before we get into implementation details: since there are times when you might really want the graphics character "hidden" by a soft key, I have designed an "escape" sequence. Pressing "control-comma" (normally the graphics heart character) signals to the soft key routine that the next character is *not* to be translated. Thus, even the heart may be generated by pressing control-comma twice.

#### The Nifty-Gritty

Program 1 shows the complete source of Easykey, the program which implements all the features mentioned above. The program is composed of five primary parts.

The first part, with line numbers in the 2000-2999 range, is used to hook the routine into Atari's OS. First, we search the Handler Table looking for the E: device. When we find it, we hold onto its address and put the address of our replacement driver in the table instead. Recall that the address in HATABS must be the address of the Handler Routine Table. We copy the current table (presumably the Atari default table, from ROM) to our NEWETBL (new E. table) and replace the entry point for the get-a-single-character routine with the address of our new routine (NEWGETCH) less one (always required, see commentary on the M: driver in my January column). We then change LOMEM so BASIC won't wipe us out and exit.

The second part of this process is the new get-a-single-character routine for the screen editor (E:) device. Most of this code is copied directly from the OS ROMs, the only exceptions being the branches to locations in ROM and the call to the keyboard

get-a-character routine (KGETCH).

The third part, NEWGETCH (NEW Key-board GET single CHaracter routine), is the heart of this whole process. Here is where individual keystrokes are actually interpreted from their hardware codes and characteristics to a more palatable ATASCII code. But here, also, is where the system is vulnerable to our machinations. Since nothing "downstream" of the keyboard handler (e.g., the rest of the E: driver, CIO, BASIC, etc.) knows what happened at the physical keyboard level, the calling routines will believe the keyboard handler no matter what it tells them.

Actually, NEWGETCH is fairly simple. First it checks to see if it is already processing a soft key. If so, it simply hands the caller the next key of the soft key's string. If not, it goes and gets a real key from the keyboard. If that key is a heart, it simply gets the next real keyboard key and passes that back to the caller (our "escape" clause). Otherwise, if the keyboard key was not control-A through Control-Z, then the key is returned to the caller unchanged.

If the keyboard key was one of the definable soft keys, its value is used to index into a table of soft key string pointers. Here one last validity check is made: if the string pointer is zero, the keyboard key is returned to the caller and no soft key string handling occurs. If a string pointer is encountered, its address is placed in KPTR and used to access all the characters in the string.

Note that zero bytes are translated to \$9B (CR) characters and that any character with its most significant bit on terminates the string (by zeroing the high order byte of KPTR).

The fourth part of this routine is simply the above-mentioned table of soft key string pointers. Note that we take advantage of the fact that the assembler places zeroes into .WORDS (or .BYTES) which are undefined.

The last part, of course, consists of the actual strings. Note the flexibility here. Control-D (label SD), for example, includes the modified CR character (\$00) as also its last character by simply turning on the MSBit, producing a code of \$80.

#### Soft Keys: Using Them From BASIC

To conserve space and to show the flexibility of the soft key system, I included strings for only three keys: control-D, which causes a disk directory listing, control-S for "SETCOLOR," and control-P for "PRINT#." The simplest way to add more soft key strings is to put them into the source given (with labels "SA" through "SZ," as appropriate) and assemble the whole thing at an address appropriate for your system.

But you can add or change soft key strings



dynamically from BASIC via POKEs, etc. Note that Easykey, as given here, reserves over 450 bytes for soft strings. Note also the addresses of the labels "STABLE" and "STRINGS." If you assemble your own copy of Easykey, be sure and note the addresses of these two labels and convert them to decimal if you intend to use dynamic soft keys.

### ...if you control these seven pointers, you control BASIC...

Program 2, the BASIC program, is a sample which will allow you to redefine all 26 soft keys to any string you like and then save the resulting definitions in a disk file for later use. Study the technique, and you should be able to produce any kind of soft keys you might want. The program fragment (Program 3) will allow the reloading of predefined sets produced by the previous program.

#### Inside Atari BASIC: Part 4

This month we will feature a short discussion of the various "tables" used by Atari BASIC and how to find them. Some of this material is well covered by some of the articles in *COMPUTE!'s First Book of Atari*, so if you are too impatient to wait for next month you can run out and buy the book. Next month we will begin to use the information we discover this month to "fool" BASIC into letting us do things it was never designed for. We begin:

When an Atari BASIC program is SAVED to disk or cassette, there are only 14 bytes of zero page written out along with the main tables and program. These 14 bytes consist of seven two-byte pointers (in the traditional low-byte, high-byte form) which tell BASIC where everything is in the particular program being SAVED (or later being LOADED). All the other important zero page locations (and there are over 50 of them) are regenerated and/or recalculated by BASIC anytime you type NEW or SAVE (or, for some locations, RUN, GOTO, etc.). The implication is that, if you control these seven pointers, you control BASIC so let's examine their names and functions. Table 1 gives a summary thereof.

The first thing you may note about this table is that some of the locations (indicated by asterisks)

have duplicate labels. If you examine the mnemonic meanings, you will probably see why: the pointer can mean different things in different contexts. For example, the space pointed to by location \$80 (decimal 128) is used for different purposes, depending on whether BASIC is currently working on entering a new line (it uses OUTBUFF) or executing an expression within a program (when it uses ARGOPS).

You might also note that I provided a list of more than seven pointers. The locations \$8E and \$90 are not SAVED and reLOADED because they are always dependent on the current state of the program (i.e., whether it is RUNning, whether it has executed a DIM statement, etc.). They are included here for completeness: aside from the zero page locations (and the \$600 page locations with BASIC A+), these pointers completely define BASIC's usage of the Atari computer's memory space. So now let's go into detail about what each of these pointers is used for.

**Table 1: BASIC's Critical Zero-Page Pointers**

Location	Mnemonic	Which means:
Hex	Decimal	Label:
80	128	LOMEN
80	128 *	ARGOPS
80	128 *	OUTBUFF
82	130	VNTP
84	132	VNTD
86	134	VVTP
88	136 *	STMTAB
8A	138	STMCUR
8C	140	STARP
--	---	
8E	142	ENDSTAR
8E	142 *	RUNSTK
90	144	TOPRSTK
90	144 *	MEMTOP

We already noted that ARGOPS is used in expression evaluation. That is, whenever BASIC sees any kind of expression to be evaluated [e.g.,  $3*A + B$  or  $SIN(30)$  or  $2^{(LOG(4/EXP(Y*Z^3)) - 1/(Z^{2.5} + ATN(Z)))}$  or even 1.25], it must put intermediate results and/or operators on a "stack." ARGOPS points to a 256 byte area reserved for both the argument stack and the operator stack. (What actually happens in Execute Expression is extremely complex and far beyond the scope of this article.) Since expression evaluation and program entry cannot occur at the same time, OUTBUFF shares this same 256 byte space. When a program line is entered, BASIC checks it for syntax and converts it to internal tokens, placing these tokens temporarily into this 256 byte buffer (before moving them into the appropriate place in the



program, depending on the line number). Again, this process is complex, but the results have been documented here in prior columns and in such places as *De Re Atari* and *COMPUTE!'s First Book of Atari*.

VNTP and VNTD point to the Variable Name Table. In Atari BASIC and BASIC A+, only the first occurrence of a name causes an entry to be added to this table. Within the tokenized program, the name(s) are replaced by a "variable number" which refers to the name's position within the name table. The names are simply placed in this table one after the other, with no intervening bytes, and the end of a name is signaled by turning on the significant bit (\$80, 128 decimal) of its last character. Note that the dollar sign on the end of a string name and the left parenthesis on the end of an array name *are* included in this table.

VVTP and ENDDVVT define the limits of the Variable Value Table. Aside from the actual tokenized BASIC program, this is probably the most interesting of the tables. Each variable occupies eight bytes in this table, so the variable number token need only be shifted left three times to index to the proper location herein. In Part 5 of this series, we will delve into this table in depth, finding many ways to fool BASIC, but there is no room in this issue for more on the subject.

STMTAB defines the beginning of the tokenized program; and, since there is no proper label to refer to, we may consider that STARP defines the end of same. Again, I refer to previous parts of this series for details on the structure of tokenized lines. STMCUR is interesting because it normally points to the actual line currently being executed. This would be one way of implementing special "statements" in Atari BASIC; a USR call would cause the subroutine to use STMCUR to examine the rest of the line for variables, etc. But my comments on ease of use, etc., from last month still apply: I don't think this is really practical.

STARP is the last of the seven pointers that are SAVED and LOADED. Actually, it is included only to point to the end of the program area. The string/array space is *not* SAVED or LOADED (but see Al Baker's article on "Atari Tape Data Files" in *COMPUTE!'s First Book of Atari* for some tricky techniques which I may expand on in future columns). STARP and ENDSTAR define the limits of the string/array space. Atari BASIC is different from Microsoft-style BASICs in that arrays and strings are allocated from this space in the order they are DIMensioned and are not moved around relative to each other after that. Thus, if you code "DIM A\$(100),B(3,3)" then you can use ADR(A\$) + 100 as the address of B(0,0).

Finally, there is the run-time stack, defined by

RUNSTK and TOPRSTK. When a GOSUB or FOR (or WHILE in BASIC A+) is encountered, the current "address" (consisting of the line number and statement offset within the line) must be "pushed" onto a stack to wait for the corresponding RETURN or NEXT (or ENDWHILE) to "pop" it off, so that the loop or mainline routine may continue where it left off. This stack thus expands and contracts as necessary while a program is running. Again, full details of how the stack is accessed can't be discussed here. In any case, the mechanism is relatively simple for GOSUB and WHILE; only FOR...NEXT presents some interesting problems.

Before we leave this topic for this month, we should note that when a program is SAVED all seven pointers are "relativized" to zero. That is, each pointer has the value of LOMEM (which is also the first pointer) subtracted from it. Then when the program is LOADED, the current value of LOMEM is added to each pointer, thus allowing self-relocating BASIC programs. A side effect of this process is that the first pointer is thus always zero (actually two bytes of zero), and BASIC uses this fact as a self-check when LOADING: it assumes that any file which does *not* start with two zero bytes cannot be a BASIC SAVED program.

### Tidbit #1: Structured Programs

An often desirable construct within properly structured programs is this one:

```
1. IF <expression> THEN <procedure-1>
   ELSE <procedure-2>
```

Since BASIC doesn't support procedures, we will modify this to the more familiar-looking form:

```
2. IF <expression> THEN GOSUB <line-1>
   ELSE GOSUB <line-2> or, using BASIC A+,
3. IF <expression> : GOSUB <line-1> ELSE :
   GOSUB <line-2> : ENDIF
```

But still, the Atari BASIC programmer cannot use either of these forms. Take heart! There is a solution which is a logical replacement for 2., above:

```
4. ON <logical-expression> + 1 GOSUB <line-1>, <line-2>
```

Note that there is a subtle difference: where the IF allowed "expression," we now require "logical-expression." The reason is fairly obvious if you recall that a logical expression in Atari BASIC (e.g., A<B or B=0 or A\$<>B\$) always evaluates to a one (true) or zero (false). By adding one (the "+ 1" in 4.) to a logical expression's value, we have a value of either one or two, something which ON...GOSUB is quite happy with since it GOSUBs to line-1 if the value is one and line-2 if the value is two.



If you really do have an "expression" to replace (e.g., IF A THEN...), simply change it into a logical expression by comparing it to zero, thus:

```
IF A THEN ...
becomes
IF (A<>0) THEN ...
which becomes
ON (A<>0) + 1 THEN ...
```

**...we are going to let Atari's  
CLEAR-SCREEN character  
do all the work for us.**

P.S.: If you want some structuring, but not too much, notice that the GOSUBs in 2. and 4. may be changed to GOTOs with similar effects.

### Tidbit #2: A Bug in DOS 2.0S

DOS 2.0S and OS/A+ have an improvement which allows much faster disk reads and writes. When DOS detects that a large data transfer is about to take place, it drops into what is called *Burst I/O Mode*. However, when a file is opened for update (OPEN #1,12,...), burst I/O should not take place. DOS handles update writes correctly, but will often blow it on update reads. The following two, one-byte patches may be made and then DOS should be re-written to the disk (with INIT under OS/A+, with menu option "H" under Atari DOS 2.0S). Caution: do *not* apply these patches to any other versions of DOS!

from BASIC:	from DEBUG:
POKE 2596,144	C A24<90
	C AD5<IF
POKE 2773,31	('C' is the Change command in BUG.)

### Tidbit #3: Clearing Memory (Revisited)

My thanks to Jerry White for permission to share his ideas on this with you. This concept is actually the result of a series of coincidences. Coincidence #1: a zero byte in screen memory is displayed as a space on the screen (not true on most machines, where \$20—decimal 32—is the space character). Coincidence #2: the Atari CLEAR-SCREEN character (SHIFT-CLEAR or CTRL-CLEAR) is not subjected to most of the cursor range checks that other characters must go through. Coincidence #3: the code to clear the screen doesn't just clear

one line 24 times (as does, for example, the Apple II's code); instead it simply starts at what it thinks is the lowest address being displayed and continues to the top of memory.

By now, it should be obvious that we are going to let the Atari's CLEAR-SCREEN character do all the work for us. The only thing we must do is fool it into believing that the "screen" is where we want it and is the size we want it.

CLEAR-SCREEN starts clearing at the location pointed to by \$58 (88 decimal) and continues until one-byte short of the page pointed to by \$6A (106 decimal). That is, it always stops clearing at location \$xxFF, where xx is one less than the contents of \$6A. So our memory clear program fragment looks something like this:

```
LOWADDR = ????: REM the lowest address to clear
HIADDR = ????: REM the highest address to clear
!! must end on xxFF boundary !!
* SVLOW1 = PEEK(88) : SVLOW2 = PEEK(89)
SVHI = PEEK(106)
POKE 106,INT( (HIADDR + 1) / 256 )
TEMP = INT(LOWADDR/256) : POKE 89,TEMP
POKE 88,LOWADDR-256*TEMP
PRINT CHR$(125); : REM this does the actual clear
POKE 106,SVHI
* POKE 88,SVLOW1 : POKE 89,SVLOW2
```

Some *cautions* are in order (as usual): 1) The screen editor thinks that it really has cleared the screen and homed the cursor. For safety's sake, it is probably best to follow that code fragment with either a GRAPHICS statement or a real screen clear. 2) Since you can only specify the high (ending) address to the nearest page boundary, you have to be careful you aren't wiping something else out.

For once, though, *caution* number (1) has a good side effect. If you follow the program fragment given above with a GRAPHICS statement, then locations 88 and 89 are going to get recalculated anyway! So the lines marked with asterisks may be omitted in such cases.

P.S.: If you have BASIC A+, there is a much easier method, related to the way strings may be cleared. Given that you know LOWADDR and HIADDR, as in the fragment given above, you may clear the area via the following:

```
poke lowaddr,0 : move lowaddr,lowaddr + 1,
hiaddr-lowaddr (And, wouldn't you know it, another caution: the system gets very unhappy if hiaddr = lowaddr.)
```

Next month, we will teach you how to have your Atari take over the entire Bell telephone system. All you need is 37 billion dollars, 25000 miles of #0000 gauge copper wire, and a toothpick. *Caution:* if you try this you musssssssssssssstttttt tabbbbbbbbbbbbeeeeeee suuuuuurrrrrr.....asdf aresetasyghxvnbaer6q23uqerngt1357 etaoin shrdlu



# Program 1.

```

0000      0990      .PAGE "      EQUATES"
1000 ;;;;;;;;;;;;;;
1010 ;
1020 ; EASYKEY --
1030 ; A program to ease repetitive typing
1040 ; when using Atari BASIC, etc.
1050 ;
1060 ; Written by Bill Wilkinson
1070 ; Optimized Systems Software
1080 ;
1090 ; for the May, 1982, issue of COMPUTE!
1100 ;
1110 ;;;;;;;;;;;;;;
1120 ;
1130 ;
1140 ; Equates to subroutines, etc., located
1150 ; in the Atari OS ROMs
1160 ;
1170 ; CAUTION: these equates are for the
1180 ; revision 'A' ROMs.
1190 ;
FCB3      1200 SWAP = $FCB3
FA88      1210 ERANGE = $FA88
006B      1220 BUFCNT = $6B
0054      1230 ROWCRS = $54
0055      1240 COLCRS = $55
006C      1250 BUFSTR = $6C
F6E2      1260 KGETCH = $F6E2
004C      1270 DSTAT = $4C
02FB      1280 ATACHR = $2FB
009B      1290 CR = $9B
F66E      1300 EGETC2 = $F66E
F67C      1310 EGETC3 = $F67C
0063      1320 LOGCOL = $63
F90A      1330 BELL = $F90A
F6AD      1340 DOSS = $F6AD
F634      1350 RETUR1 = $F634
1360 ;
1370 ;
031A      1380 HATABS = $31A
02E7      1390 SYSTEMLOMEM = $2E7
0000      1490      .PAGE
1500 ;
1510 ; EQUATES UNIQUE TO THIS ROUTINE
1520 ;
0022      1530 QUOTE = $22 ; The " character
0000      1540 NUL = 0 ; A nul...which becomes a CR
1550 ;
00E6      1560 ZTEMP1 = $E6 ; shared with fltg pt routines
1570 ;
00FF      1580 LOW = $FF
0100      1590 HIGH = $100
0080      1600 KQUIT = $80 ;MSBit says quit
1800 ;
1810 ;;;;;;;;;;;;;;
1820 ;
1830 ; CHOOSE THIS ORIGIN TO FIT YOUR SYSTEM !!
1840 ;
1850 ;;;;;;;;;;;;;;
1860 ;
1F00      1900 ORIGIN = $1F00
2200      1910 NEWLOMEM = ORIGIN+$300
0000      1920 * = ORIGIN
1F00      1990      .PAGE "      Hooking the driver into OS"
2000 ;
2010 ;;;;;;;;;;;;;;
2020 ;
2030 ; The 'HOOKUP' routine --
2040 ;
2050 ; this portion of the code simply hooks

```

```

2060 ; our replacement driver into the E:
2070 ; handler vector table.
2080 ;
2090 ;
2100 ;;;;;;;;;;;;;;
2110 ;
2120 HOOKUP
2130      LDX #0 ;start at beginning
2140 LOOKLOOP
2150      LDA HATABS,X
2160      CMP #'E ; looking for the E: driver
2170      BEQ EFND ;got it
2180      INX
2190      INX
2200      INX ; to next name
2210      BNE LOOKLOOP ; keep looking
2220      BEQ HOOKUP ; actually, this is fatal error
2230 ;
2240 ; found the E driver
2250 EFND
1F10 BD1B03 2260      LDA HATABS+1,X ; get LSB of addr
1F13 85E6 2270      STA ZTEMP1
1F15 BD1C03 2280      LDA HATABS+2,X ; and MSB
1F18 85E7 2290      STA ZTEMP1+1
1F1A A9C7 2300      LDA #NEWETBL&LOW
1F1C 9D1B03 2310      STA HATABS+1,X ; replace with our table
1F1F A91F 2320      LDA #NEWETBL/HIGH
1F21 9D1C03 2330      STA HATABS+2,X
2340 ;
1F24 A00F 2350      LDY #15 ; all bytes of table
2360 EMVLP
2370      LDA (ZTEMP1),Y ; get a byte of old tbl
2380      STA NEWETBL,Y ; to our table
2390      DEY
2400      BPL EMVLP ; and do more
2410 ;
2420 ; E: handler vector table is moved
2430 ;
1F2E A942 2440      LDA #NEWGETCH-1&LOW
1F30 8DCB1F 2450      STA NEWETBLGC ; change the get character ptr
1F33 A91F 2460      LDA #NEWGETCH-1/HIGH
1F35 8DCC1F 2470      STA NEWETBLGC+1
2480 ;
2500 ; tables, etc. are corrected
2510 ;
2520 ; move lomem
2530 ;
1F38 A900 2540      LDA #NEWLOMEM&LOW
1F3A 8DE702 2550      STA SYSTEMLOMEM
1F3D A922 2560      LDA #NEWLOMEM/HIGH
1F3F 8DE802 2570      STA SYSTEMLOMEM+1
1F42 60 2580      RTS
1F43      2990      .PAGE "      The replacement E: driver"
3000 ;
3010 ;;;;;;;;;;;;;;
3020 ;
3030 ; Begin the actual E: replacement routine
3040 ;
3050 ; This routine replaces E:'s get-a-character
3060 ; entry point.
3070 ;
3080 ;
3090 ;;;;;;;;;;;;;;
3100 ;
3110 NEWGETCH
1F43 20B3FC 3120      JSR SWAP ;<< lines with comments starting
1F46 2088FA 3130      JSR ERANGE ;<< with "<<" are copied from
1F49 A56B 3140      LDA BUFCNT ;<< without change from ROM
3150 ;
1F4B D026 3160      BNE GOEGETC3 ; was just EGETC3
3170 ;
1F4D A554 3190      LDA ROWCRS ;<<

```



```

1F4F 856C 3200 STA BUFSTR ;<<
1F51 A555 3210 LDA COLCRS ;<<
1F53 856D 3220 STA BUFSTR+1 ;<<
3230 EGETC1
1F55 20791F 3240 JSR NEWKGETCH ; What we really wanted to replace
1F58 844C 3250 STY DSTAT ;<<
1F5A ADFB02 3260 LDA ATACHR ;<<
1F5D C99B 3270 CMP #CR ;<<
3280 ;
1F5F F015 3290 BEQ GOEGETC2 ; was just EGETC2
3300 ;
1F61 20ADF6 3310 JSR DOSS ;<<
1F64 20B3FC 3320 JSR SWAP ;<<
1F67 A563 3330 LDA LOGCOL ;<<
1F69 C971 3340 CMP #113 ;<<
1F6B D003 3350 BNE EGETC6 ;<<
1F6D 200AF9 3360 JSR BELL ;<<
3370 EGETC6
1F70 4C551F 3380 JMP EGETC1 ;<<
3390 ;
3400 GOEGETC3
1F73 4C7CF6 3410 JMP EGETC3 ;becuz branch can't get there
3420 ;
3430 GOEGETC2
1F76 4C6EF6 3440 JMP EGETC2 ;becuz branch can't get there
3450 ;
1F79 3990 .PAGE " The replacement KGETCH routine"
4000 ;
4010 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
4020 ;
4030 ; This is the routine that replaces the
4040 ; ROM-based KGETCH (Keyboard GET Character)
4050 ; routine. This routine is designed
4060 ; especially for NEWKGETCH, but could be
4070 ; called in place of KGETCH if the user
4080 ; wished.
4090 ;
4100 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
4110 ;
4120 NEWKGETCH
1F79 ADC61F 4130 LDA KPTR+1 ; msb of ptr to multikey string
1F7C F024 4140 BEQ NOSTRING ; but no string, so normal KGETCH
1F7E 85E7 4150 STA ZTEMP1+1 ; need to put it in zero page
1F80 ADC51F 4160 LDA KPTR ; ditto the LSB
1F83 85E6 4170 STA ZTEMP1
1F85 EEC51F 4180 INC KPTR ; bump ptr for next time
1F88 D003 4190 BNE NEWK1
1F8A EEC61F 4200 INC KPTR+1 ; MSB also if needed
4210 NEWK1
4220 ;
4230 ;
4240 ;
1F8D A000 4230 LDY #0
1F8F B1E6 4240 LDA (ZTEMP1),Y ; get next char in string
1F91 1003 4250 BPL NEWK2 ; last char has MSB on
1F93 8CC61F 4260 STY KPTR+1 ; so reset pointer & flag
4270 NEWK2
1F96 297F 4280 AND #$7F ; ensure only 7 bits
1F98 D002 4290 BNE NEWK3 ; except...
1F9A A99B 4300 LDA #CR ; null becomes CR
4310 NEWK3
1F9C 8DFB02 4320 STA ATACHR ; this is what KGETCH does
1F9F 4C34F6 4330 GORETURN JMP RETUR1 ; seems silly, but it works
4340 ;
4350 ; no string waiting for us...get a real key
4360 ;
4370 NOSTRING
1FA2 20E2F6 4380 JSR KGETCH ; a real key
1FA5 844C 4390 STY DSTAT ; why??? just in case
1FA7 C900 4400 CMP #0 ; control-comma, the heart
1FA9 F017 4410 BEQ REALCTL ; handled special
1FAB C91B 4420 CMP #27 ; one more than control-Z
1FAD B0F0 4430 BCS GORETURN ; a regular key
4440 ;

```

```

4450 ; if here, we have control-A through control-Z
4460 ;
1FAF 0A 4470 ASL A
1FB0 A8 4480 TAY ; doubled value used as index
1FB1 B9FE1F 4490 LDA STABLE-2,Y ; -2 becuz control-A is
1FB4 8DC51F 4500 STA KPTR ; value of 1 instead of zero
1FB7 B9FF1F 4510 LDA STABLE-1,Y
1FBA 8DC61F 4520 STA KPTR+1 ; MSB of addr of string
1FBD F0E0 4530 BEQ GORETURN ; oops...no string for this key
4540 ;
1FBF 4C791F 4550 JMP NEWKGETCH ; and pass back first char of string
4560 ;
4570 ;
4580 ; special handling for control-comma !!!!
4590 ;
4600 REALCTL
1FC2 4CE2F6 4610 JMP KGETCH ; it is used as an escape key
4620 ; ; to allow real control chars !
4630 ;
4900 ;
4910 ; MISCELLANEOUS RAM USAGE
4920 ;
1FC5 0000 4930 KPTR .WORD 0 ;master ptr to next char
4940 ;
1FC7 0000 4950 NEWETBL .WORD 0,0 ;to be filled in
1FC9 0000
1FCB 0000 4960 NEWETBLGC .WORD 0,0,0,0,0
1FCD 0000
1FCF 0000
1FD1 0000
1FD3 0000
1FD5 0000
4970 ;
1FD7 4990 .PAGE " The pointer table"
5000 ;
5010 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
5020 ;
5030 ; The strings and String TABLE
5040 ;
5050 ; Each string in use has its address placed
5060 ; in the corresponding location in STABLE
5070 ; Any control character which should not be
5080 ; translated should have $0000 as its
5090 ; string address in STABLE.
5100 ;
5110 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
5120 ; (first, get on even boundary)
5130 * = ORIGIN+$100
5140 STABLE
5150 .WORD SA,SB,SC,SD,SE,SF,SG,SH,SI
2000 0000
2002 0000
2004 0000
2006 3420
2008 0000
200A 0000
200C 0000
200E 0000
2010 0000
}***ERROR - 5-UNDEFINED
2012 0000 5160 .WORD SJ,SK,SL,SM,SN,SO,SP,SQ,SR
2014 0000
2016 0000
2018 0000
201A 0000
201C 0000
201E 7F20
2020 0000
2022 0000
}***ERROR - 5-UNDEFINED
2024 8620 5170 .WORD SS,ST,SU,SV,SW,SX,SY,SZ
2026 0000
2028 0000

```



```

202A 0000
202C 0000
202E 0000
2030 0000
2032 0000
}***ERROR - 5-UNDEFINED
2034      .PAGE "          The soft key strings"
        5990
        6000 ;
        6010 ;;;;;;;;;;;;;;
        6020 ;
        6030 ; The actual strings
        6040 ;
        6050 ; we need supply strings only for
        6060 ; the desired softkeys
        6070 ;
        6080 ; Note that inverse video is not
        6090 ; allowed, except that the last
        6100 ; character of a string has its
        6110 ; MSBit on, same as inverse video
        6120 ;
        6130 ; Note that zero bytes ('NUL') get
        6140 ; converted to RETURN ($9B) characters
        6150 ;
        6160 ;;;;;;;;;;;;;;
        6170 ;
        6200 SD .BYTE "CLOSE #7:OPEN #7,6,0,"
        6210 .BYTE QUOTE,"D:*.","QUOTE
        6220 .BYTE ":FOR I=0 TO 256:GET #7,I"
        6230 .BYTE ":PRINT CHR$(I);:NEXT I"
        6240 .BYTE NUL+KQUIT
        6250 ;
        6260 SP .BYTE "PRINT ",KQUIT+'#
        6270 ;
        6280 SS .BYTE "SETCOLO",KQUIT+'R
        6290 ;
        6300 ; Note the methods of getting quoted strings
        6310 ; in .BYTE statements and turning on the
        6320 ; MSBits of a character
        7000 ;
        7010 ; finally, we set up for LOAD-AND-GO
        7020 ;
        7030      *= $2E0
        7040      .WORD HOOKUP
        7050 ;
        9999 .END

```

## Program 2.

```

1000 REM *****
1010 REM *
1020 REM * EASYLOAD --
1030 REM *      A BASIC PROGRAM THAT ALLOWS
1040 REM *      THE USER TO MAKE HIS/HER OWN
1050 REM *      SET OF "SOFTKEYS" FOR USE
1060 REM *      WITH THE "EASYKEY" PROGRAM
1070 REM *
1080 REM *****
1090 REM
1100 REM :BEFORE RUNNING THIS PROGRAM, BE SURE
1110 REM :THAT "EASYKEY" HAS BEEN LOADED AND
1120 REM :RUN. YOU MAY VERIFY THIS BY CHECKING
1130 REM :THE VALUE OF PEEK(744) -- IT SHOULD
1140 REM :MATCH THE NEWLOMEM PAGE VALUE IN
1150 REM :THE LISTING OF EASYKEY
1160 REM :[ PEEK(744) = 34 IF EASYKEY IS
1170 REM :ASSEMBLED AS GIVEN HERE ]
1180 REM
1190 REM == FIRST, SET UP ADDRESSES, ETC. ==
1200 STABLE=8192:REM STABLE = $2000
1210 STRINGS=STABLE+(2*26)
1220 REM
1230 DIM KEY$(130)

```

```

1240 REM
1250 REM FIRST, CLEAR OUT OLD SOFTKEY DEFINITIONS
1260 REM
1270 FOR ADDR=STABLE TO STRINGS-1:POKE ADDR,0:NEXT ADDR
1280 REM
1290 PRINT CHR$(125);
1300 PRINT "When prompted, enter a softkey string"
1310 PRINT "for the given control-key."
1320 PRINT
1330 PRINT "Remember: no inverse keys and use"
1340 PRINT "control-comma (the heart) to request"
1350 PRINT "a RETURN key code. Actual use of the"
1360 PRINT "Return key terminates the string for"
1370 PRINT "the given softkey."
1380 PRINT :PRINT "[just RETURN will undefine that key]"
1390 PRINT :PRINT
1400 ADDR=STRINGS:REM WHERE WE START STORING SOFTKEY STRINGS
2000 REM MAIN LOOP
2010 FOR KEY=0 TO 25
2020 PRINT "ConTrol-";CHR$(65+KEY);:INPUT KEY$
2030 KLEN=LEN(KEY$):IF NOT KLEN THEN 2290
2040 KEY$(KLEN)=CHR$(ASC(KEY$(KLEN))+128):REM SET MSB OF LAST CHARACTER
2050 IF ADDR+KLEN>=STABLE+512 THEN PRINT "too much defined!";CHR$(253):END
2070 REM SET ADDRESS OF STRING IN STABLE
2080 TEMP=INT(ADDR/256):POKE STABLE+KEY+KEY+1,TEMP
2090 POKE STABLE+KEY+KEY,ADDR-256*TEMP
2100 FOR KPT=1 TO KLEN
2110 POKE ADDR,ASC(KEY$(KPT))
2120 ADDR=ADDR+1
2130 NEXT KPT
2200 REM END OF PROCESSING FOR THAT KEY
2290 NEXT KEY
3000 REM *****
3010 REM
3020 REM NOW WE ALLOW YOU TO SAVE THIS NEWLY DEFINED SET ON DISK
3030 REM
3040 REM SIMPLY HIT BREAK AT THE QUESTION IF YOU DON'T WANT TO SAVE
3050 REM
3100 PRINT :PRINT :PRINT "What set of softkeys should we save"
3110 PRINT " this definition under (1-999) ";:INPUT SET
3120 IF SET<1 OR SET>999 OR SET<>INT(SET) THEN 3100
3150 KEY$="D:SOFTKEY.":KEY$(LEN(KEY$)+1)=STR$(SET)
3200 OPEN #1,8,0,KEY$
3210 FOR KPT=STABLE TO ADDR
3220 PUT #1,PEEK(KPT)
3230 NEXT KPT
3250 CLOSE #1
3280 PRINT :PRINT "==== normal end ===="
3290 END

```

## Program 3.

```

30000 REM *****
30010 REM
30020 REM THIS PROGRAM OR PROGRAM FRAGMENT IS
30030 REM INTENDED TO BE USED TO RELOAD ONE
30040 REM THE SETS OF SOFTKEYS CREATED BY
30050 REM "EASYLOAD".
30060 REM
30100 STABLE=8192:REM STABLE=$2000
30110 DIM NAME$(20)
30200 PRINT "What set of softkeys should be"
30210 PRINT " reloaded from disk (1-999) ";:INPUT SET
30220 NAME$="D:SOFTKEY.":NAME$(LEN(NAME$)+1)=STR$(SET)
30250 OPEN #1,4,0,NAME$
30290 TRAP 30400
30300 FOR KPT=STABLE TO STABLE+512
30310 GET #1,KEY:POKE KPT,KEY
30320 NEXT KPT
30400 CLOSE #1
30500 REM COULD RETURN FROM SUBROUTINE, IF DESIRED

```



*For people who would find it very difficult to enter commands or use the keyboard, the program below prints messages (or BASIC commands) on screen and allows the user to move a pointer to the desired word and enter it into the computer. It can accept input from either the numeric keypad or from a joystick. It is designed to work on a PET with the Graphics keyboard, but the central idea could be adapted to any computer.*

# Handicapped Programming

Hilton B. Souther  
Lynchburg, VA

This program allows a person to build messages on the screen or could even be used to program. The input is made by pressing a button on a joystick or by pressing keyboard numeric pad no. 5.

When the program first comes up, it gives you an option of moving a pointer with the numeric pad or using the joystick; from then on, it responds based on the input. The program is set up for the joystick on the high order bits on the user port. If there is no joystick, all commands come from the numeric pad.

Most of the PET commands, except seldom used ones such as CMD, are displayed on the screen. A pointer will be shown opposite the various commands or BASIC words and the pointer can be moved up, down, right, or left until it is opposite the desired word. The user then pushes the button or the no. 5, depending on the mode of operation, and the word that is being indicated by the pointer is printed at the bottom of the screen. After the item is printed, the pointer is again activated for further entry. At the bottom of the screen where the message is being built, a vertical pointer indicates the next location for the next character location or word location. If one wishes to enter information not shown in the listing of BASIC words, put the pointer opposite the word "SCREEN" and enter. The word will print and then erase; a pointer at the top of the screen will then activate and will be pointing at the alphabet and numbers located across the top of the screen. The up arrow is used for the pointer; it can only move left or right; it will wrap around at the end. When the pointer is beneath the desired character, push the no. 5 key or button and the single character will be added to the message. To change the top of the screen to special characters, enter the greater than or less than sign. The line will change, and then you can enter the special characters. To return to

the words, enter the @ symbol and the pointer will be back at the words.

When the message is completed or you wish to print on a printer at Device No. 4, enter the % symbol; the message or basic statement will go to the printer. The program will clear the array, present the screen again and allow the next message or statement to be entered. To prevent the screen from scrolling, there is a limitation of 240 characters. The program checks for the limit and, if reached, prints the message and continues.

The program presently goes to a printer; however, it could be changed to write to a disk or to the cassette, or even to push out to a modem and communicate with another computer.

There is sound with each movement of the pointers using the CB2 convention. All the computer needs for this program to function is the ability to close switches so a handicapped person could make it work. Of course, the necessary switches would have to be used in place of the joystick. I did not mention that to enter blanks, enter the shaded space. To delete characters, enter the right bracket and the message will decrease by one character. The program will run on an 8K machine, Original or Upgrade ROMs; however, the REMARKS will have to be deleted first. If you have a wedge in the 8K and try to load, it won't fit. After you load it on the 8K, you can't save unless you delete one line since you will get an out-of-memory message.

I have not stated in this article all of the line numbers and their functions. I think the REMARKS do a pretty good job of that. The sort for sorting the words is the SHELL sort. I read all of the variables first to count them and then dimension the array that size. The program could be changed by using new data statements. It is presently set up for 64 words on the screen in four rows. The program



could be changed to calculate the number of words being used and adjust the various locations of the pokes to accommodate the different values. I did not try that since I was trying to accomplish something that might be useful for someone who cannot access the computer any other way.

If anyone wishes a copy of the program, I will make a copy from the original for a cost of \$3.00. Please enclose a tape and SASE mailer.

Hilton B. Souther  
115 Windingway Rd.  
Lynchburg, VA 24502

```

16 BEG=1:REM HOUSEKEEPING
18 GOTO112
20 DI=1:PRINT"{HOME}{15 DOWN}":REM
  [HM][DN15]
24 GB=3:POKETR,0:ONDIGOSUB32,220
28 POKEK4,16:POKEK5,15:POKEK6,51:P
  OKEK5,0:POKEK4,0:POKEK6,0
30 GOTO24
32 POKEI,ASC(">"):FORJ=1TO10:NEXT:
  POKEE6,0
33 S9=0:FORJ=I+1TOI+9:IFPEEK(J)=32
  ANDPEEK(J+1)=32THENS9=J-1:
  GOTO37
36 POKEJ,PEEK(J)OR128:NEXT
37 IFPEEK(TR)=255THEN37
38 GOTO40
39 FORJ=I+1TOS9:POKEJ,PEEK(J)-128:
  NEXT:RETURN
40 IFPEEK(TR)=E4THENGOSUB39:POKEI,
  32:GOSUB88:I=I-40:RETURN
44 IFPEEK(TR)=E5THENGOSUB39:POKEI,
  32:GOSUB94:I=I+40:RETURN
48 IFPEEK(TR)=E3THENGOSUB39:POKEI,
  32:GOSUB100:I=I+10:RETURN
52 IFPEEK(TR)=E2THENGOSUB39:POKEI,
  32:GOSUB106:I=I-10:RETURN
54 IFPEEK(TR)=E1THEN64
60 RETURN
64 GOSUB39:FORJ=I+1TOS9:FR=PEEK(J)
  :IFPEEK(J)=32THENCT=CT+1:I
  FCT=2THEN80
68 IFPEEK(J)<>32THENCT=0
72 POKELO,FR:POKELN,32:LN=LN+1:POK
  ELN,30:LO=LO+1:E$=E$+CHR$(
  PEEK(J)+64)
74 NEXT:IFCT=0THENPOKEL0,32:POKELN
  ,32:LN=LN+1:POKELN,30:LO=L
  O+1:E$=E$+" "
78 IFLEFT$(E$,3)="SCR"THENGOSUB394
  :RETURN
80 REM ADD TO STRING AND MOVE THE ~

```

```

  POINTER TO THE NEXT POSITI
  ON
82 IFLEN(C$)+LEN(E$)>240THENPT=10
  1:GOTO316
84 C$=C$+E$:E$="":RETURN
88 IFI=T3ORI=T2ORI=T1ORI=TTHENI=I+
  K-T+40
90 RETURN
94 IFI=K3ORI=K2ORI=K1ORI=KTHENI=I-
  (K-T)-40
96 RETURN
100 IF((I-8)+10)/40=INT((I+10)/40)T
  HENI=I-40
102 RETURN
106 IF((I-8)/40)=INT((I-8)/40)THENI
  =I+40
108 RETURN
112 PRINT"{CLEAR}{03 DOWN}"TAB(10)"
  {REV}READING VARIABLES":PR
  INT:REM[CL][DN3]
114 K=33448:K1=K+10:K2=K+20:K3=K+30
  :BEG=1:W=0:K4=59467:K5=K4-
  1:K6=K4-3:K7=K4-8
116 READW$:IFW$<>"9999"THENW=W+1:GO
  TOL16
118 I=32848:T=I:T1=T+10:T2=T+20:T3=
  T+30:DIMW$(W):LO=33488:LN=
  LO+40
122 RESTORE:FORP=0TOW-1:READW$(P):P
  RINT".":W$(P)=W$(P)+" ":N
  EXT:P=0:GOTO138
128 DATAPEEK(,INPUT,COS(,GET,ASC(,E
  XP(,ATN(,LOG(,RND(,SGN(,SQ
  R(,CHR$(,LEFT$(
130 DATALEN(,MID$(,RIGHT$(,STR$(,VA
  L(,POS(,TAB(,FRE(,SYS,TIS,
  TI,USR,"DEF FN"
132 DATACLR,SAVE,CONT,LIST,LOAD,NEW
  ,RUN,VERIFY,DIM,ON,GOSUB,G
  OTO,IF,THEN,ABS(
134 DATAREM,RESTORE,RETURN,STOP,WAI
  T,SIN(,PRINT,SCREEN,AND,CL
  OSE,FOR,INT(,NEXT
136 DATANOT,OPEN,OR,POKE,READ,SPC(,
  STEP,TO,END,DATA,9999
138 TA=W-1:PRINT:PRINTTAB(52)"{REV}
  SORTING":PRINT
140 TC=TA
142 TC=INT(TC/2):IFTC>=1THEN146
144 GOTO172
146 REM SORT THE LIST
148 FORU=0TOTC:FORT9=UTOTA-TCSTEPTC
  :T0=T9:T$=W$(T9+TC)
150 IFT$>W$(T0)THEN154
152 W$(T0+TC)=W$(T0):T0=T0-TC:IFT0=
  >1THEN150
154 W$(T0+TC)=T$:PRINT"";:NEXTT9,U
  :GOTO142

```



```

162 P=0:PRINT "{HOME}{02 DOWN}";:FOR
    Y=0TO3:FOR Y1=0TO15:REM [HM]
    [DN2]
164 PRINTTAB(Y*10+1);WS(P):P=P+1:IF
    P>W+1THEN170
168 NEXTY1:PRINT "{HOME}{02 DOWN}";:
    NEXTY:PRINT:REM [HM] [DN2]

170 GOTO20
172 CR$=CHR$(13):REM SET CARRIAGE ~
    CONTROL
176 M$="{REV}4{OFF} FOR LEFT {REV}6
    {OFF} FOR RIGHT {REV}8{OFF
    OFF} UP AND {REV}2{OFF} DO
    WN":M1$="THE NUMBER {REV}5
    "
180 S=32808:TR=515:E1=34:E2=42:E3=4
    1:E4=50:E5=18:E6=525
182 INPUT "{CLEAR}{02 DOWN} USING A J
    OYSTICK {REV}Y{OFF}ES OR {
    REV}N{OFF}O";JS:IFPEEK(500
    03)THEN TR=151:E6=158
186 IFJS="Y"THEN TR=59471:E1=63:E2=2
    23:E3=239:E4=127:E5=191
188 IFJS="Y"THEN M$="JOYSTICK":M1$="
    {REV}RED{OFF} BUTTON"
192 GOTO322
196 IFS=32807THENPOKE32808,32:S=S+4
    0:POKES,30:RETURN
200 POKES+1,32:POKES,30:RETURN
204 IFS=32848THENPOKE32847,32:S=S-4
    0:POKES,30:RETURN
208 POKES-1,32:POKES,30:RETURN
212 POKEK7,0
216 GOSUB220:RETURN
220 POKES-40,PEEK(S-40)OR128
221 IFPEEK(TR)=255THEN221
224 GOSUB304:GOSUB227:IFPEEK(TR)=E1
    THENPT=PEEK(S-40):GOSUB266
    :GOSUB230
226 GOTO228
227 IFPEEK(S-40)>127THENPOKES-40,PE
    EK(S-40)-128:RETURN
228 POKEE6,0:RETURN
230 REM
232 TB=LEN(C$):IFLEN(C$)=0THENRETUR
    N
234 IFPT=0THENRETURN
238 IFASC(RIGHT$(C$,1))=101THENC$=L
    EFT$(C$,LEN(C$)-1):GOTO288

242 IFPT=60ORPT=62THEN262
244 IFPT=29ANDLEN(C$)=0THENRETURN
246 IFPT=29ANDLEN(C$)=1THENPOKELN,3
    2:POKELO-1,32:GOSUB398:RET
    URN
250 IFPT=29THENPOKELN,32:LN=LN-1:PO
    KELN,30:POKELO-1,32:LO=LO-
    1:GOSUB400
252 IFPT<>29ANDPT<>32THENPOKELO,PT:
    POKELN,32:LN=LN+1:LO=LO+1:
    POKELN,30
256 IFPT=32THENPOKELO,230:LO=LO+1:P
    OKELN,32:LN=LN+1:POKELN,30

262 FORU=1TODL:NEXT:RETURN
266 IFPT=0THENDI=1:POKES,32:RETURN
270 IFPT=62THENGOSUB388:GOSUB392:PO
    KES,30:RETURN
272 IFPT=60THENGOSUB388:GOSUB390:PO
    KES,30:RETURN
276 IFPT=102THENPT=32:C$=C$+CHR$(PT
    ):RETURN
278 IFPT<>94ANDPT<>224ANDPT<>29THEN
    C$=C$+CHR$(PT+64):GOTO280
280 IFPT=94THENC$=C$+CHR$(255)
284 RETURN
288 POKEE6,0:PRINT:PRINT "{CLEAR}{07
    DOWN}YOUR MESSAGE IS":PRI
    NTC$:REM[CL] [DN7]
292 ML=1:OPEN4,4:CMD4:PRINTC$
296 PRINT#4:CLOSE4
300 IFML=1THENLO=LO-LEN(C$):LN=LO+4
    0:L=0:P=0:C$="":PRINT "{CLE
    CLEAR}":P=0:GOTO372
304 IFPEEK(TR)=E2THENGOSUB227:S=S-1
    :GOSUB196
308 IFPEEK(TR)=E3THENGOSUB227:S=S+1
    :GOSUB204
310 RETURN
314 PRINT
316 PRINT "{CLEAR} YOUR MESSAGE LE
    NGTH HAS REACHED THE MAXI
    MUM FOR THIS ONE, ";
318 PRINT "PRINTING AND THENCONTINU
    ING.":GOSUB262
320 DL=2000:GOSUB262:ML=1:GOTO288
322 PRINT "{CLEAR}{03 DOWN}":REM[CL]
    [DN3]
324 PRINT "YOU WILL BE PRESENTED WIT
    H A LIST OF "
326 PRINT "WORDS, OPPOSITE THE WORDS
    WILL BE A >
328 PRINT "TO MOVE THE > PUSH THE
330 PRINTM$
332 PRINT "WHEN THE > POINTS TO THE ~
    PHRASE
334 PRINT "YOU WISH THEN PUSH "M1$
336 PRINT "TO CHANGE THE > FROM THE ~
    PHRASES TO
338 PRINT "CHARACTERS, PUSH "
340 IFE5<>191THENPRINTRIGHT$(M1$,9)
    "{OFF} WHEN OPPOSITE
342 IFE5=191THENPRINTCR$;M1$"{OFF} ~
    WHEN OPPOSITE
344 PRINT "THE WORD SCREEN AND THEN ~
    YOU WILL HAVE

```



```

346 PRINT"THIS POINTER ^ AT THE TOP
    OF THE SCREEN
348 PRINT"ONLY MOVE LEFT OR RIGHT W
    ITH THIS ^
350 PRINT"WHEN UNDER THE DESIRED CH
    ARACTER PUSH
352 PRINTM1$
354 PRINT"TO CHANGE THE LINE DISPLA
    Y AT THE TOP
356 PRINT"ENTER THE '<' OR THE '>' W
    HEN THE ^ IS {REV}UNDER{
    OFF} THEM
358 PRINT"TO DELETE A CHARACTER, E
    NTER THE ']'
360 PRINT"TO ENTER A SPACE ENTER TH
    E &.
362 PRINT"TO PRINT OUT ENTER THE '%'
    SYMBOL
364 PRINT"TO RETURN TO THE WORDS EN
    TER THE '@' SYMBOL. ALL
    ITEMS POINTED TO
365 PRINT"WILL BE IN {REV}REVERSE F
    IELD{OFF}"
366 PRINT"TO START PUSH ";M1$
368 IFPEEK(TR)<>E1THEN368
370 FORU=1TO500:NEXT
372 PRINT"{CLEAR}";

```

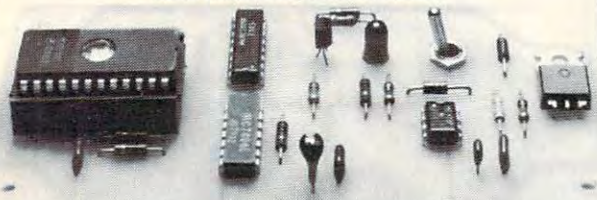
```

374 A1$="<.,;:~$'\()+-=?@%&]<
376 A$=">ABCDEFGHIJKLMNOSTUVWXYZ
    0123456789@]&":PRINTA1$;
378 IFBEG=1THENDIMA%(LEN(A$)),A1%(L
    EN(A1$)+1):TY=32768
380 IFBEG=1THENFORJA=TYTOTY+LEN(A1$
    ):A1%(JA-TY)=PEEK(JA):NEXT
    :A1%(JA-TY-1)=34
382 IFBEG=1THENPRINT"{CLEAR}";A$:FO
    RJA=TYTOTY+LEN(A$):A%(JA-T
    Y)=PEEK(JA):NEXT
384 IFBEG=1THENBEG=2:GOTO162
386 PRINT:GOTO162
388 PRINT"{HOME}";:FORU6=1TO40:POKE
    U6+32767,32:NEXT:RETURN
390 FORU6=TYTOTY+LEN(A$):POKEU6,A%(
    U6-TY):NEXT:RETURN
392 FORU6=TYTOTY+LEN(A1$):POKEU6,A1
    %(U6-TY):NEXT:RETURN
394 DI=2:POKEI,32:POKES,30:FORY9=1T
    OLEN(E$):POKELO,32:LO=LO-1
    :NEXT
396 POKELO,32:POKELN,32:LN=LN-LEN(E
    $):POKELN,30:E$="":RETURN
398 LO=LO-1:LN=LO+40:POKELN,30:C$="
    ":RETURN
400 C$=LEFT$(C$,LEN(C$)-1):RETURN ©

```

## 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:

6502  
6800  
6809  
8080/8085/Z80

### Interface to:

6820 PIA or 6522 VIA  
6820 PIA  
6820 PIA  
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.



**Micro Technical Products, Inc.**

814 W. Keating Ave., Dept. M  
Mesa, Arizona 85202 • 602-839-8902



Make fuller use of your PET with

## Programming the PET/CBM

by Ray West

504 pages/ 17 chapters + index/ many programs, tables, and diagrams. Paperback/ 19 x 16 x 2 1/2 cms/ ISBN 0 9507650 0 7.

"*Programming The PET/CBM* is a massive reference work, containing almost five hundred pages of closely-packed information on the computer's inner workings. The data covers BASIC, Machine Language, PET/CBM architecture and associated commercial products to an astonishing degree of detail. The book is not suited to the beginner, who may be bewildered by the wealth of detail; but it's unquestionably the most comprehensive and accurate reference I have seen to date." — Jim Butterfield

"Your book is EXCELLENT!" — Jim Strasma

Mail-order from:

Ray West, Level Ltd., 1001 Dunwoody Chace,  
Atlanta, GA 30328.

\$26.00 (Delivery in about 6 weeks)

\$32.00 (Delivery in about 2 weeks)



# We have ways...

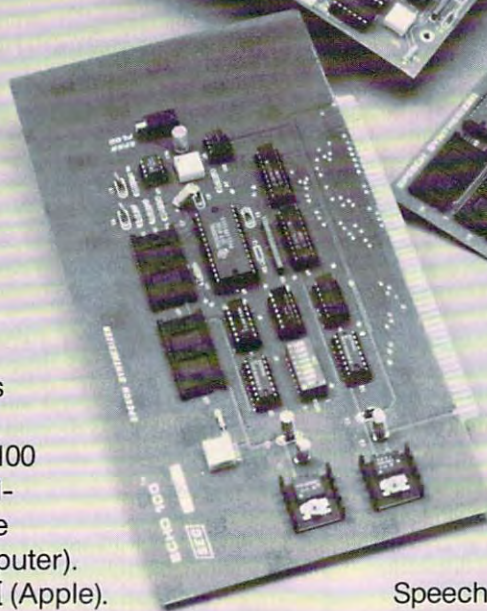
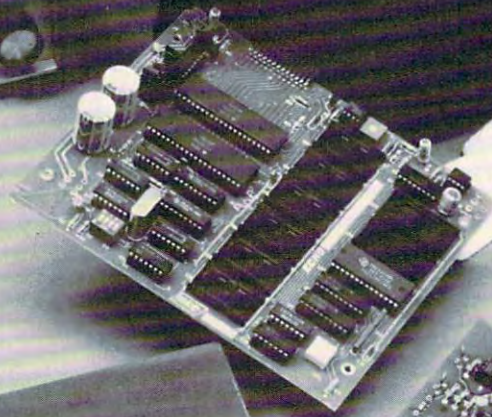
## to make it talk!

Give your computer the magic of speech with an ECHO™ Speech Synthesizer!

There are four new models to choose from: the ECHO 80 (TRS-80 models I & III), ECHO 100 (S-100), the ECHO GP (general-purpose, serial/parallel) and the ECHO SBC (single board computer). These join the popular ECHO II (Apple).

ECHO Speech Synthesizers use LPC synthesis pioneered by Texas Instruments, combined with a phoneme-based operating system to provide any vocabulary you desire while using a minimum of your computer's memory.

The TEXTALKER Speech Generator, standard with all ECHOs except the SBC, translates normal English text to speech automatically.



Simple commands allow you to select from:

- 63 different pitch levels
- Words spoken monotonically or with intonation
- Fast or slow speech output
- Entire words pronounced or spelled letter by letter
- Different volumes
- Spoken punctuation if desired

Speech applications are virtually unlimited, ranging from education to games to aiding the handicapped. The flexibility and low price of ECHO Speech Synthesizers make them the logical choice when adding speech to your system.

For more information see your dealer or contact:



**STREET ELECTRONICS CORPORATION**  
3152 E. La Palma Avenue, Suite D  
Anaheim, CA 92806 Telephone: 714/632-9950

[www.commodore.ca](http://www.commodore.ca)



# New Products



## ComputerTown International To Offer New Services

ComputerTown, USA!, a grass-roots computer literacy network sponsored by the National Science Foundation, offers guidance and support to anyone who wishes to learn about computers on a grass-roots, community-based level. Under its NSF grant, CTUSA! has fostered ComputerTown projects across the US and abroad, and is currently completing a book-length "Implementation Package" outlining the details of starting and conducting a ComputerTown project.

Funded until October, 1983, CTUSA! has proven so effective that an on-going project, ComputerTown International (CTI) has now come to life. The purpose of CTI is to build on the knowledge and resources gained by ComputerTown, USA! and to keep the communication channels open after the NSF funding ends. CTUSA! and CTI will both operate from the offices of People's Computer Company in

Menlo Park, California, and most of the staff will overlap.

Three initial services are planned by ComputerTown International;

- 1) A *Speakers' List* will be developed and made available to interested groups or individuals. This list will be comprised of qualified individuals willing to speak on computer literacy and related subjects at locally organized seminars and events.
- 2) CTI is organizing *Two Teams of Specialists* – teachers and consultants, who are expert in the theory and practice of public access computer literacy and informal education. While these services will be offered on a fee basis, this will not decrease the free support and advice offered under the NSF grant, but will enable CTI to provide a depth of assistance and instruction impossible on a complimentary basis.
- 3) *The ComputerTown Bulletin* has proven an invaluable vehicle of communication among the international network of people and organizations involved in computer literacy. As of March, 1982, the *Bulletin* will no longer

be funded by the National Science Foundation. In order to continue publication, ComputerTown International seeks individuals or businesses willing to sponsor one issue of the *Bulletin*. In exchange, the *Bulletin* will devote two pages to news and information about the sponsor's products and services and will acknowledge the sponsorship in print. Sponsors are encouraged to print extra copies for their own use as mailing and handout materials. Six thousand bi-monthly issues of each sponsored *Bulletin* will continue to be offered free of charge to anyone with an interest in computer literacy.

For further information about ComputerTown International and its services, write:

ComputerTown International  
P.O. Box E  
Menlo Park, CA 94025

## Time Bomb Develops New Language Skills

A new educational game, Time Bomb, has been released by

## WORD LISTS for your ATARI™

All the 2 through 10 letter words from the Merriam Webster Dictionary. Over 23,000 total entries, arranged in easy to enter and use numbered data statements.

CREATE YOUR OWN  
Proofreaders, Word Challengers, Word Mazes  
or Searchers and many other uses.

Package includes 2 discs, documentation  
and example programs.

JAVJR CO.  
P. O. Box 233  
Altus, OK 73521

TO ORDER SEND \$29.95 TO:



## MEMOREX FLEXIBLE DISCS

WE WILL NOT BE UNDER-  
SOLD!! 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





# ATARI HOME COMPUTERS



## ATARI 800

16K ... \$679

32K ... \$749

48K ... \$819

## ATARI 400

16K... \$329

32K... \$478

48K... \$555

410 Recorder	\$76.00
810 Disc Drive	\$449.00
822 Printer	\$269.00
825 Printer	\$629.00
830 Modem	\$159.00
820 Printer	\$269.00
850 Interface	\$169.00
New DOS 2 System	\$29.00

### PACKAGES

481 Entertainer	\$83.00
482 Educator	\$130.00
483 Programmer	\$57.00
484 Communicator	\$344.00

### ATARI HOME COMPUTER PROGRAMS

#### Home Office

CX404 ATARI Word Processor	\$119.00
CX8102 Calculator	\$29.00
CX412 Dow Jones Investment Evaluator	\$99.00
CX4109 Graph It. Joystick optional	\$17.00
CX4104 Mailing List	\$20.00
CX4115 Mortgage & Loan Analysis	\$13.00
CX406 Personal Financial Management System	\$59.00
CX4103 Statistics 1	\$20.00
CX8107 Stock Analysis	\$20.00
CXL4015 TeleLink 1	\$23.00

#### Home Study

CX4101 An Invitation to Programming 1	\$20.00
CX4106 An Invitation to Programming 2	\$23.00
CX4107 Biorhythm	\$13.00
Conversational Languages (ea.)	\$46.00
CX4121 Energy Czar	\$13.00
CX4114 European Countries & Capitals	\$13.00
CX4108 Hangman. Joystick optional	\$13.00
CX4102 Kingdom	\$13.00
CXL4007 Music Composer	\$47.00
CX4123 Scram. uses joystick	\$20.00
CX4112 States & Capitals	\$13.00
CX4110 Touch Typing	\$20.00

#### Home Entertainment

PAC MAN ..... \$35.00

Centipede ..... \$35.00

Caverns of Mars ..... \$32.00

CXL4013 Asteroids	\$35.00
CXL4004 Basketball	\$27.00
CX4105 Blackjack	\$13.00
CXL4009 Computer Chess	\$33.00
CXL4012 Missile Command	\$35.00
CXL4008 Space Invaders	\$35.00
CXL4011 Star Raiders	\$42.00
CXL4006 Super Breakout	\$33.00
CXL4010 3-D Tic-Tac-Toe	\$27.00
CXL4005 Video Easel	\$24.00

#### Programming Languages and Aids

CXL4003 Assembler Editor	\$47.00
CXL4002 ATARI BASIC	\$47.00
CX8126 ATARI Microsoft BASIC	\$70.00
CXL4018 PILOT	\$72.00
CX405 PILOT	\$105.00

CX30 Paddle	\$18.00
CX40 Joy Stick	\$18.00
CX853 16K RAM	\$89.00
Microtek 16K RAM	\$75.00
Microtek 32K RAM	\$159.00
Ramdisk (128K)	\$539.00
Inteck 48K Board	\$249.00
One year extended warranty	\$50.00

### THIRD PARTY PROGRAMS

#### ATARI Program Exchange:

Eastern Front 41	\$25.50
Avalanche	\$15.50
Outlaw	\$15.50
747 Landing Simulation	\$15.50
Babel	\$15.50
Dog Daze	\$15.50
Downhill	\$15.50
Attack!	\$15.50
Blackjack-Casino	\$15.50
Reversi II	\$15.50
Domination	\$15.50
Solitaire	\$15.50
Disk Fixer	\$15.50
Supersort	\$15.50
Data Management	\$15.50
Chameleon	\$15.50
Instedit	\$15.50
Insomnia	\$15.50
My First Alphabet	\$25.50
Mapware	\$18.00
Number Blast	\$11.50
Family Cash Flow	\$15.50
Weekly Planner	\$15.50
Bowler's Data Base	\$13.00
Banner Generator	\$11.50

#### Crystal Software

Bermuda Fantasy	\$26.00
Beneath Pyramids	\$20.00
Galactic Quest	\$26.00
House of Usher	\$20.00
Forgotten Island	\$26.00
Haunted Palace	\$33.00
Compumax (Acct. Rec., Gen. Ledger, Inventory, Payroll, ea.)	\$110.00
Visicalc	\$169.00
Letterperfect (Word Processor)	\$109.00
Ricochet	\$14.50
Crush Crumble & Chomp (cassette or disk)	\$24.00
Star Warrior (cassette or disk)	\$29.00
Rescue at Rigel (cassette or disk)	\$24.00
Datstones (cassette or disk)	\$16.00
Invasion Orion (cassette or disk)	\$18.50
Mission Asteroid	\$22.00
MouskATTACK	\$31.00
The Next Step	\$34.00
Softporn	\$27.00
Wizard & Princess	\$29.00
K-BYTE Krazy Shoot Out (ROM)	\$39.00
Protector (Disk 32K)	\$32.00
Jaw Breaker (on line disk)	\$27.00
Ghost Hunter (cassette)	\$24.00
Ghost Hunter (disk)	\$30.00

# PRINTERS

Centronics 739-3	\$619.00
Centronics 739-1	\$519.00
Diablo 630 Special	\$1799.00
Epson	
MX70	\$359.00
MX80	\$469.00
MX80FT	Call
MX100	Call
NEC	
8023	\$549.00
7730	Call
7720	Call
7710	Call
Okidata	
82A	\$499.00
83A	\$769.00
84	\$1129.00
Citron Starwriter	
F10-40 CPS	\$1469.00
F10-55 CPS	Call
Paper Tiger	
445G	\$699.00
460G	\$899.00
560G	\$1129.00
Talley	
8024-7	\$1399.00
8024 L	\$1629.00
MPC Apple Parallel Board & Cable	\$69.00

# XEROX®

Xerox 820	
System I 5 1/4"	\$2450.00
System II 8"	\$2950.00
CPM 5 1/4"	\$169.00
Word Processing	\$429.00
Super Calc	\$269.00

# Texas Instruments



**TI-99/4A \$299**

PHA 2100 R.F. Modulator	\$43.00
PHP 1600 Telephone Coupler	\$179.00
PHP 1700 RS 232 Accessories Interface	\$179.00
PHP 1800 Disk Drive Controller	\$239.00
PHP 1C50 Disk Memory Drive	\$389.00
PHP 2200 Memory Expansion (32K RAM)	\$319.00
PHP 1100 Wired Remote Controllers(Pair)	\$31.00
32K Expansion	\$329.00
PHP Printer Solid State	\$319.00
PHM 3006 Home Financial Decisions	\$26.00
PHM 3013 Personal Record Keeping	\$43.00
PHD 5001 Mailing List	\$60.00
PHD 5021 Checkbook Manager	\$18.00
PHM 3008 Video Chess	\$60.00
PHM 3010 Physical Fitness	\$26.00
PHM 3009 Football	\$26.00
PHM 3018 Video Games I	\$26.00
PHM 3024 Indoor Soccer	\$26.00
PHM 3025 Mind Challengers	\$22.00
PHM 3031 The Attack	\$35.00
PHM 3032 Blast	\$22.00
PHM 3033 Blackjack and Poker	\$22.00
PHM 3034 Hustle	\$22.00
PHM.3036 Zero Zap	\$18.00
PHM 3037 Hangman	\$18.00
PHM 3038 Connect Four	\$13.00
PHM 3039 Yahtzee	\$22.00
Tombstone City 21st Century	\$34.00
Munch Man	\$34.00
TI INVADERS	\$34.00
CAR WARS	\$34.00

All items subject to availability and price change.

# computer mail order west

CALL TOLL FREE

# 800-648-3311

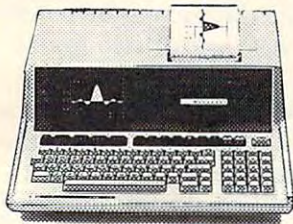
Franco  
Habla  
Espanol

IN NEVADA, CALL (702) 588-5654

P.O. BOX 6689, STATE LINE, NEVADA 89449

www.commodore.ca





## HP-85 \$1999

80 Column Printer	\$799.00
HP-125	\$1999.00
HP-83	\$1699.00
HP-85 16K Memory Module	\$169.00
5 1/4" Dual Master Disc Drive	\$1929.00
NEW! HP-87	\$1999.00
Hard Disk w/Floppy	\$4349.00
Hard Disk	\$3440.00
"Sweet Lips" Plotter	\$1149.00

## HP-41CV Calculator . . . \$249.00

41 C	\$189.00
11 C	\$119.00
12 C	\$129.00
34 C	\$117.00
38 C	\$119.00
HP-41 Printer	\$340.00

### HPIL CALCULATOR PERIPHERALS

IL Modul	\$104.00
Digital Cassette	\$449.00
Printer/Plotter	\$419.00
Card Reader	\$164.00
Optical Wand	\$99.00

## NEC

8001-A	\$669.00
8031	\$669.00
8012	\$549.00

## Disks

Maxell	
MD I (box of 10)	\$36.00
MD II (box of 10)	\$46.00
MFD I (8")	\$44.00
MFD II (8" Double Density)	\$54.00
Syncom (box of 10)	\$29.00

## Apple

Call for availability and prices  
on all Apple computers and peripherals.

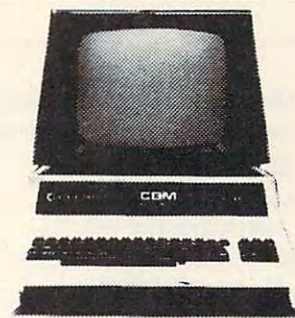
## Monitors

Amdex 12" B&W	\$129.00
12" Green	\$139.00
13" Color	\$349.00
NEC	
12" B&W	\$169.00
12" Color	\$339.00
TI 10" Color	\$349.00

## commodore BUSINESS MACHINES

### SOFTWARE

Word Pro 5 Plus	\$319.00
WordPro4 Plus	\$299.00
WordPro3 Plus	\$199.00
Commodore Tax Package	\$589.00
Visicalc	\$169.00
Medical Billing	\$449.00
The Source	\$89.00
OZZ Information System	\$289.00
Dow Jones Portfolio	\$129.00
Pascal	\$239.00
Legal Time Accounting	\$449.00
Word Craft 80	\$289.00
Power	\$79.00
Socket-2-Me	\$20.00
Jinsam	\$Call
MAGIS	\$Call
The Manager	\$209.00
Softrom	\$129.00
Real Estate Package	\$799.00
BPI Inventory Control	\$319.00
BPI Job Costing	\$319.00
BPI Payroll	\$319.00
BPI General Ledger	\$329.00
Creative I SAM	\$79.00
Creative General Ledger	\$229.00
Creative Accounts Receivable	\$229.00
Creative Inventory	\$229.00

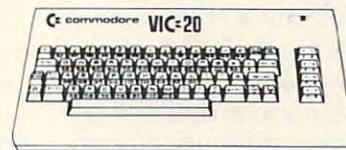


## CBM 8032 \$1069

4032	\$969.00
4016	\$769.00
8096 Upgrade Kit	\$399.00
Super Pet	\$1599.00
2031	\$529.00
8050	\$1299.00
4040	\$969.00
8300 (Letter Quality)	\$1799.00
8023	\$769.00
4022	\$599.00
Pet to IEEE Cable	\$37.00
IEEE to IEEE Cable	\$46.00
Tractor Feed for 8300	\$240.00
8010 Modem	\$229.00

## VIC 20 \$259

### COMPLETE



VIC 6 Pack Program	\$44.00
VIC 1530 Commodore Datasheet	\$69.00
VIC 1540 Disk Drive	\$499.00
VIC 1515 VIC Graphic Printer	\$339.00
VIC 1210 3K Memory Expander	\$32.00
VIC 110 8K Memory Expander	\$53.00
VIC 1011 RS232C Terminal Interface	\$43.00
VIC 1112 VIC IEEE-488 Interface	\$86.00
VIC 1211 VIC 20 Super Expander	\$53.00
VT 232 VICTerm I Terminal Emulator	\$9.00
VIC 1212 Programmers Aid Cartridge	\$45.00
VIC 1213 VICMON Machine Language Monitor	\$45.00
VIC 1901 VIC AVENGERS	\$23.00
VIC 1904 SUPERSLOT	\$23.00
VIC 1906 SUPER ALIEN	\$19.00
VIC 1907 SUPER LANDER	\$23.00
VIC 1908 DRAW POKER	\$23.00
VIC 1909 MIDNIGHT DRIVE	\$23.00

Terminal	\$13.00
Un Word	\$13.00
Grafix Menagerie	\$11.00
VIC PICS	\$15.00
Ticker Tape	\$13.00
Banner Headliner	\$13.00
RS 232	\$39.00
VT 106A Recreation Pack A	\$44.00
VT 107A Home Calculation Pack A	\$44.00
VT 164 Programmable Character/Graphics	\$12.00
Household Finance	\$27.00
VIC Games	\$19.00
VIC Home Inventory	\$13.00
VIC Rec/Ed II	\$13.00
VL101 Introduction to Computing	\$19.00
VL 102 Introduction to BASIC Programming	\$19.00
VM110 VIC 20 Programmers Reference Guide	\$15.00

In-stock items shipped same day you call.

No risk, no deposit on C.O.D. credit card or phone orders. No waiting period for certified checks or money orders. All prices shown are cash prices, add 3% for Master card and Visa. Pre-paid orders receive free shipping in the continental United States.

## Terminals

Teletype	
910	\$579.00
912C	\$699.00
920C	\$749.00
950	\$939.00
Call for computers	
Zenith 219	\$749.00
Adds	\$549.00

## Modems

Novation Auto	\$239.00
D Cat	\$169.00
Cat	\$159.00
Hayes	
Smart	\$239.00
Livermore Star	\$119.00

Nev. & Pa. residents add sales tax.

# computer mail order east

CALL TOLL FREE **800-233-8950**

INTERNATIONAL CALLS AND IN PA. CALL (717) 327-9575  
477 E. THIRD ST., WILLIAMSPORT, PA 17701

Patricio  
Habla  
Espanol

[www.commodore.ca](http://www.commodore.ca)



Program Design, the Greenwich, Connecticut firm that specializes in the design, development, and marketing of educational and game software for micro-computers.

Time Bomb is a version of the pen-and-pencil game Hangman. The computer chooses at random a secret word from a list of hundreds of words. The player is told the number of letters in the secret word, then asks the computer if certain letters appear in the word. After the computer shows if the letters are present, the player tries to guess the word. Every wrong guess shortens the fuse attached to a large bomb. The player must guess the secret word before the bomb goes off.

Some of the secret words are well known to all. Others are less common, or consist of unusual letter combinations that make guessing them difficult. There are two separate games, each with their own word lists. In addition, the disk version allows the user to add his or her own word lists.

Time Bomb was written by Dr. Dean Victor, author of Mini-crossword, AstroQuotes, and several other PDI programs. The program uses high resolution and player/missile graphics, and presents a challenge to both children and adults.

Time Bomb is available for use on Atari 400/800 computers

with a memory of at least 16K. The cassette version retails for \$16.95, the disk version for \$23.95.

*Program Design, Inc.  
11 Idar Court  
Greenwich, CT 06830  
(203)661-8799 John Victor  
(203)792-8382 Laurie Hall*

## Atari Sponsors Summer Computer Camps

Atari, Inc., will conduct eight camp sessions this summer, two in each of four locations, for 10 to 18-year olds interested in computers.

The camp sessions will each last four weeks. Atari Computer Camp sessions will begin in late June or early July, and will be conducted on school and university campuses in the northeast, southeast, midwest and west. Day-to-day operation of the camps will be handled by Specialty Camps, Inc., an organization with some 25 years of experience in running theme and traditional camps. Atari is designing their own curriculum for the camps under the direction of Robert A. Kohn who has been involved with computers and education for the past 15 years. They will recruit and train their own instructors, many of whom will be professional educators.

While the formal instruction sessions will last for two hours each day, all of the computers and software will be available to campers during their free time. The daily schedule will also include traditional summer camp activities.

Teaching sessions will be limited in size, with one instructor for every five to six campers. There will be one computer for every two campers, since it has been Atari's observation that computer learning is enhanced when people work together on computer projects.

Equipment used will be Atari 400 and 800 Home Computers.

*Atari, Inc.  
1265 Borregas Ave.  
P.O. Box 427  
Sunnyvale, CA 94086*

## VIC 20 Programmers Reference Guide

The new VIC 20 *Programmers Reference Guide* is now available from Commodore Business Machines, Inc. Designed for use by first-time computerists as well as experienced programmers, the *Programmers Reference Guide* sells for \$16.95 and provides complete information about the programming of Commodore's VIC 20 home computer. Nearly 300 pages, the

## 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**



Dealer Inquiries  
Invited



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



**ALL ATARI® HARDWARE 15%-25%  
OFF LIST PRICE**

Atari 800 16K .....	740.00
Atari 400 16K .....	359.00
Atari 410 Cassette .....	80.00
Atari 810 Disk .....	480.00

### ATARI® ACCESSORIES 10%-20% OFF LIST PRICE

8K Memory Board .....	40.00
16K Memory Board .....	80.00
Joysticks (pair) .....	19.00
Paddles (pair) .....	19.00



To order: Call 617-964-3080  
Ask for mail order, or write:

**BBI Mail Order**

P.O. Box 365  
Newton Highlands, MA 02161  
(617) 964-3080

**PLUS 10%-20% OFF  
ALL ATARI® SOFTWARE  
ALSO 3RD. PARTY HARDWARE  
AND SOFTWARE AT  
COMPARABLE SAVINGS**



ECLECTIC SYSTEMS CORPORATION  
Order TOLL FREE 1-800-527-3135



P.O. Box 1166 • 16260 Midway Road  
Addison, TX 75001 • (214) 661-1370

ANNOUNCES

# EM<sup>©</sup> for the SuperPET

## EM<sup>©</sup> is ANSI standard MUMPS

### Make your SuperPET Super-Load in EM<sup>©</sup> and Dramatically Improve your Programming Productivity



EM<sup>©</sup> is more than just a programming language. It is a well integrated data management system combining with one syntax what other operating systems would call 1) an application programming language; 2) a job control language; 3) a linkage editor; 4) a database management system; and 5) a communications monitor.

#### PROGRAM MANAGEMENT:

EM<sup>©</sup> provides all programming management facilities needed to manage programs and program files. Programs can be created, edited, cataloged and debugged from within EM<sup>©</sup>. Programs can be as large as disk capacity. A resident algorithm rids memory of least frequently used variables and program modules so that what you need off-disk normally resides in memory.

#### STRING POWER:

EM<sup>©</sup> makes string handling easy with its extensive set of string operations and functions. Variable length strings can be used routinely without the obstacles presented by most other programming languages.

#### PATTERN MATCHING:

EM<sup>©</sup> can "filter" user input with a useful pattern matching that will result in fewer user or device errors. For example: dates, zip codes and names can be tested for validity with a single statement.

#### GLOBALS:

EM<sup>©</sup> obviates the need for traditional read and write operations on secondary storage devices by allowing data elements to be directly referenced as a set of subscripts; all the details of file organization and retrieval are handled by the system.

#### TIMING:

EM<sup>©</sup> enables a programmer to associate timing constraints with several operations. This feature allows testing for terminal malfunctions as well as prompting users in time-critical dialogue.

#### DATA BASE MANAGEMENT:

Sorts and merges are not necessary as EM<sup>©</sup> automatically stores data in a dynamically allocated balanced tree structure. Random access to any data item requires at most three disk reads.

#### EM<sup>©</sup> UNMATCHED IN PROGRAMMING PRODUCTIVITY:

System houses that program in EM<sup>©</sup> (MUMPS) find that their costs are lower than those of their competitors using other languages. Fewer lines of code are necessary per application. Dimension statements are not required. Subscripts may be alpha, numeric or any legal string. Data types need not be defined and can change freely throughout as EM<sup>©</sup> can recognize when it is dealing with alpha, numeric, integer or floating-point data types. EM<sup>©</sup> gives the professional programmer a full set of software tools designed for real-life tasks and problems he consistently encounters in the production and maintenance of application software. EM<sup>©</sup> adheres rigidly to ANSI MUMPS standards, which make it transportable to larger processors manufactured by DEC, TANDON, DATA GENERAL, HARRIS and others. Additionally EM<sup>©</sup> gives the less-experienced programmer the tools to do a professional job on formidable programming applications.

**You may order EM<sup>©</sup> or SuperPET by calling ECLECTIC SYSTEMS toll-free at 1-800-527-3135 from 10 AM to 4 PM CDT Monday through Friday, or you can order by mail using the form below. Texas residents call 1-214-661-1370.**

#### ECLECTIC SYSTEMS CORPORATION

P.O. Box 1166, 16260 Midway Road, Addison, Texas 75001

Here's my order for EM<sup>©</sup> @ \$299 plus \$3.75 for shipping and handling (UPS surface unless specified otherwise). Residents of Texas, Louisiana, Oklahoma City and Tulsa, Oklahoma must add applicable taxes.

☐ My certified check or money order is enclosed.

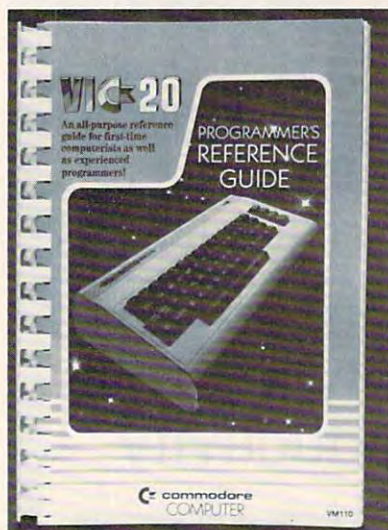
☐ Please charge my VISA # \_\_\_\_\_ or  
MasterCard # \_\_\_\_\_ Expiration date \_\_\_\_\_

Name \_\_\_\_\_ Signature \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_





guide includes illustrations, instructions, charts and programs, as well as a schematic of the VIC 20.

Commodore Business Machines, Inc.  
Computers Systems Division  
681 Moore Road  
King of Prussia, PA 19406  
(215)337-7100

## New Blue Book Available For The Apple Computer

WIDL Video, Chicago, publisher of the Apple Directories, has released the new 2nd edition of *The Apple II Blue Book*. The Blue Book is a master directory of software, hardware, peripherals, and information for the Apple II Computer. It gives Apple users a complete "where to find it" guide to available software and also includes a directory of hardware, boards, peripherals, and accessories. The meaty, 400 page Blue Book is loaded with useful information including over 5,000 software and hardware listings and more than 750 software and hardware producers.

The Software Section of the Blue Book contains program



listings for every application and features business, games, and educational software. Also included are special interest sections featuring word processing programs, graphics software, and data base management systems for the serious Apple user.

The Hardware section of the Blue Book contains new devices that interface with the Apple II Computer. There are manufacturers listed along with product information photos, price

## 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
<b>NEW</b> 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




**ECLECTIC SYSTEMS CORPORATION**  
**Order TOLL FREE 1-800-527-3135**



**P.O. Box 1166 16260 Midway Road**  
**Addison, TX 75001 (214) 661-1370**



**Authorized  Commodore service center**  
**Repair of the complete line of Commodore products**  
**In a hurry? Check our modular exchange program**

**HARDWARE:**

CBM 8032 Computer, 80 Column	\$1095
CBM 8050 Disk Drive	1340
CBM 4032 Computer, 40 Column	995
CBM 4040 Disk Drive	995
CBM 4022 Printer	649
CBM VIC 20 Computer	263
CBM VIC 1515 Printer	355
CBM 8300P (DIABLO)	1925
CBM VS100 Cassette	68
PET to IEEE Cable	33
IEEE to IEEE Cable	39

**SOFTWARE:**

OZZ	\$299
Wordcraft 80	299
The Manager (Data Base)	240
Wordpro 5+	329
Wordpro 4+	329
VISIC ALC	175
MUMPS for SuperPET	299
CMS Accounting System	call
Assembler Development	
Package	77
BASF Diskette, Box of 10	30

ANS MUMPS Programmer's Reference Manual	\$17.50
MUMPS Pocket Guide	2.00
Computer Programming in ANS MUMPS	17.50
RS232 Interface for Commodore VIC 20	39.95
EIO has: two serial asynchronous RS232 ports, two parallel ports with handshaking, one shift register, two 16 bit timers, and room for two optional buffer IC's.	188.00
EIO-C Alternate character generator ROM board for screen display allows you to display characters or graphics of your choice. Alternate characters are soft selectable. EIO board required. Call for price	
EIO-TX Terminal ROM for EIO firmware to turn your CBM into a communications terminal. Store and transmit from disk to remote host and terminal. EIO board required.	49.95
EIO-RS232 Cable for EIO board.	28.00

**BOOKS:**

Commodore Software Encyclopedia	
2nd edition	9.95
The PET Revealed	19.95
Library of PET Subroutines	19.95
PET Interfacing	16.95
PET Basic	12.95
PET and IEEE 488 Buss	15.00

**OTHER:**

EPROM Burner for CBM "burns 2716, 2732 & 2532"	89.95
Software for EPROM Burner on CBM	15.95
D.C. Hayes Smart Modem	249.00

ESC-100 RS232 interface manual selector—switch select between device A, B, or C; switching 25 conductors. Requires no input power and uses receptacle type DB25S.	140.00
ESC-120 M43 TTY-EIA/CURRENT LOOP interface unit—converts TTL level signals from the M43 to EIA or 20ma outputs. Convert EIA or 20ma inputs back to TTL level signals for the M43. Single printed circuit mounts inside the M43. Comes with interface cable.	65.00
ESC-140 RS232 standard interface cable kit. Build your own cable.	
7 conductor	16.00
12 conductor	19.95
ESC-150 The Electronic Switch/Poller allows a single computer serial port to selectively communicate with an assortment of peripheral devices. As many as seven serial asynchronous and six parallel devices may be selected. This concept effectively creates a simplified multi-drop, polled environment. The computer maintains control of the network and may	

seize any of the peripherals by transmitting a simple escape sequence to the "ESC-150".	488.00
ESC-170 RICKETYMETER—Indispensable hand-held tool for troubleshooting, checkout and installation of RS232 type serial line communications systems.	69.95
ESC-180 RICKETYTRAP is a small hand-held device which when interposed in an EIA/RS232 line can monitor and trap on any specific ASCII, binary, or control character. The RICKETYTRAP can also monitor and trap on any specific range of characters bit selectable. This device can also check for type of parity being sent and proper framing. The RICKETYTRAP is switch selectable for 1 or 2 stop bits and can operate at speeds of 150 to 9600 baud.	Call for price
ESC-200 PREDITOR is a small microprocessor with ROM firmware, RAM, and 8 RS232 asynchronous serial input/output ports. The PREDITOR is available in either stand-alone, self-	

contained, or rack-mounted models. The PREDITOR can be programmed by firmware to function as a prompter and editor in a distributed processing network, transforming 4 dumb asynchronous terminals into 4 intelligent terminals communicating with 4 computer asynchronous input/output ports.	Call for price
ESC-861M The DATA CONCENTRATOR accepts data from a multitude of inputs—printers, CRTs, parallel ports, status lines, etc.—then transmits the composite (multiplexed) data stream down a high speed synchronous serial line to a remote DATA CONCENTRATOR. The remote DATA CONCENTRATOR separates (demuxes) the composite data into its original form to be transmitted to the computer ports. Alternately, multiple computer port data is multiplexed, transmitted down the high speed synchronous serial line, demuxed, and finally dispatched to the individual terminals.	850.00

**Order TOLL FREE 1+800-527-3135**

**10 AM to 4 PM CDT Monday through Friday**

**Texas residents call 1+214-661-1370**

**VISA, MASTER CHARGE, MONEY ORDERS, AND C.O.D. "Certified Check" accepted.**

**Units in stock shipped within 24 hours, F.O.B. Dallas, Texas.**

**All equipment shipped with manufacturer's warranty.**

**Residents of Texas, Louisiana, Oklahoma City and Tulsa, Oklahoma must add applicable taxes.**

**Be sure to write to the address at the top of this ad for more information. Dealer inquiries welcome.**

[www.commodore.ca](http://www.commodore.ca)



information, and complete descriptions.

In addition, the Blue Book has a Resource Section that lists reference manuals, publications, newsletters, Apple user groups, clubs, time sharing systems, and more.

The suggested retail price of the new 2nd edition Apple Blue Book is \$24.95 and is available from most computer shops, bookstores, or direct from:

*WIDL Video*  
5245 W. Diversey  
Chicago, IL 60639  
(312)622-9606

## Scott Adams' Graphic Adventure Series

Scott Adams' Adventure series is being released with high-resolution graphics for the Apple II, adding a new dimension to the game. The graphics are compressed, and drawn using a special palette of over 100 colors. In addition, the new programs support the Votrax Type 'N Talk voice synthesizer, giving a full-color Adventure that talks.

Adventureland, the first adventure in the series, is available now. You will wander through an enchanted world trying to recover 13 lost treasures, encountering wild animals, magical beings, and many other perils and puzzles. It retails for \$29.95.

*Adventure International*  
Dept. G, Box 3435  
Longwood, FL 32750

## Daisy Wheel Printer From Transtar

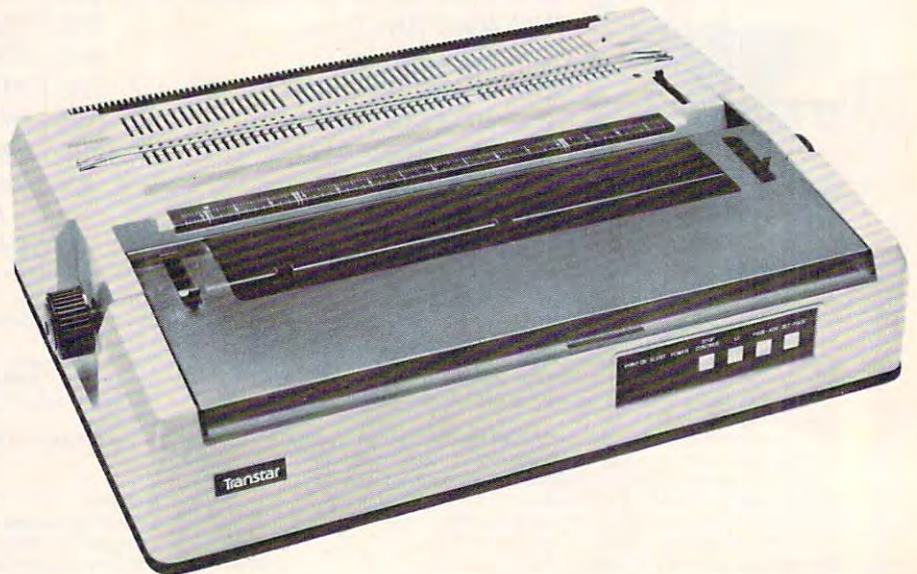
The new Transtar 140 daisy wheel printer combines 40 cps (38 Shannon) letter quality performance and reliability with a list price of \$1695. The new serial printer, built to specification by one of Japan's largest

printer manufacturers, is Diablo code compatible for plug-and-go use with Magic Wand and Wordstar. Ribbons and printwheels are also industry standard. Transtar offers a 6-month end-user warranty. Transtar's low profile package is only 6 inches high, and shipping weight is under 50 pounds for UPS

delivery. The Transtar 140 printer is available now from:

*Micro Distributors*  
11794 Parklawn Drive  
Rockville, MD 20852  
(800-638-6621)

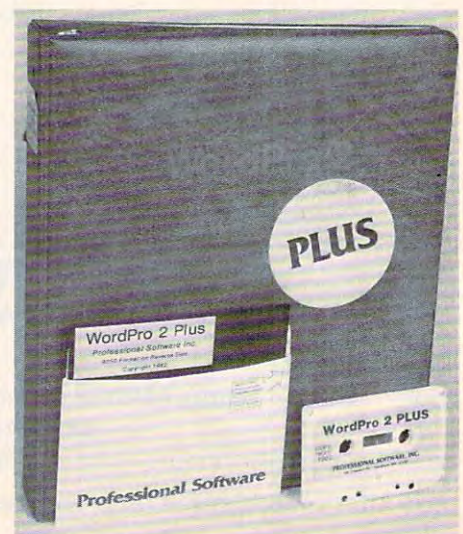
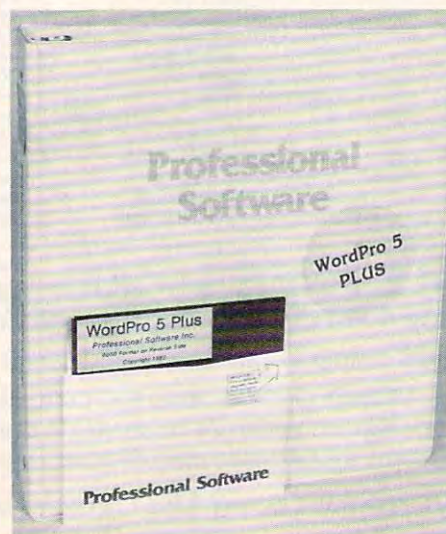
*Sigma Distributing*  
2110-116th Avenue N.E.  
Bellevue, WA  
(800-426-1412)



## Professional Software Releases Two More WordPro Packages

WordPro 5 Plus transforms your Commodore 8096 computer into

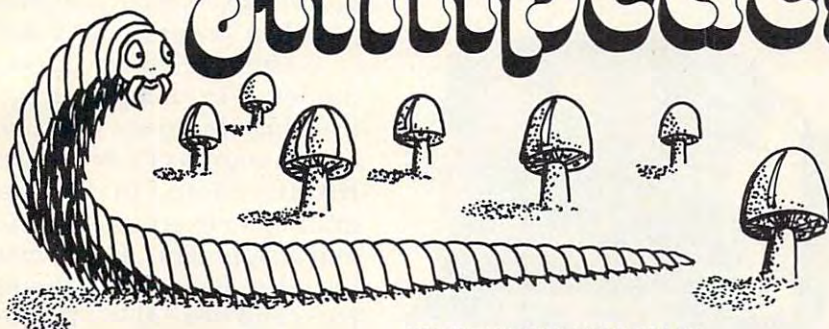
an easy to use, commercial level word processing system. It creates a system with separate text areas to make word processing easier. Multi-user capability (up to 8 workstations) is available via the addition of a multi-user interface device. It operates with many business oriented programs





# SOFTWARE STREET PRESENTS

## Millipedes



ATARI 24 K DISC ONLY \$34.95

BY

BILL BOGENREIF



800 Computer 16K ..... \$689.00

410 Recorder ..... \$79.00

850 Intf. Mod ..... \$159.00

830 Modem ..... \$149.00

### Special Software Packages

The Entertainer ..... \$84.00  
Star Raiders, Missile Command,  
and Joysticks.

The Programmer ..... \$54.00  
Basic Programming language,  
Self teaching guide to Basic  
and Basic reference manual

The Educator ..... \$115.00  
410 Recorder, Basic programming  
language, and States and Capitals.

### TOP SELLING PROGRAMS

Missile Command .....	\$33.96
Asteroids .....	\$33.96
Invitation to Prog. 2 .....	\$16.96
Invitation to Prog. 3 .....	\$16.96
Assembler/Editor .....	\$48.96
Jaw Breaker .....	\$25.95
Cranston Manor .....	\$24.95
Dodge Racer .....	\$19.95
Star Raiders .....	\$39.96
Sammy the Sea Serpent .....	\$14.95
Krazy Shootout .....	\$35.95
Space Invaders .....	\$35.95



810 Disk Drive ..... \$429.00

Prices subject to change

THE ABOVE PRICES ARE FOR PRE-PAID ORDERS.

We also feature tremendous savings from:

Atari  
Atari Program Exchange  
Adventure International  
Crystal  
P.D.I.  
L.J.K.  
Dynacomp  
Quality Software  
Avalon Hill  
Epyx  
Computer Consultants  
Synapse  
Datasoft  
United Software  
On-Line  
O.S.S.  
Software Street

Software Street is your mail order ATARI  
discount center.

Shipping costs:

Software - minimum \$2.00

Hardware - prices will vary (please call)



Call or write for your  
FREE catalog.

DEALERS INQUIRIES WELCOME

Software Street  
3392 Clipper Dr.  
Chino, CA 91710  
(714) 591-3061



ATARI IS A REGISTERED TRADEMARK



including DataPlus, Professional Software's new Information Management program. WordPro 5 Plus is designed for use on Commodore's 8032 computer with Commodore's 64K Memory Expansion Board installed. Any CBM Disk Drive may be used for document storage and any properly interfaced ASCII letter quality or dot matrix printer may be used.

WordPro 5 Plus will be available from Professional Software dealers beginning in January 1982 and will retail for approximately \$450.00.

WordPro 2 Plus is compatible with almost any CBM computer available. It requires a minimum of 16K and is sold complete with both cassette and diskette versions and is fully compatible with most CBM computers. WordPro 2 Plus will also operate on all Commodore disk drives. A wide range of popular dot matrix and letter quality printers are supported directly from the program.

WordPro 2 Plus will be available from Professional Software's dealers beginning in the first quarter of 1982, and will sell for \$199.95.

*Professional Software, Inc.  
166 Crescent Road  
Needham, MA 02194  
(617)444-5224*

## Universal Data Systems Offers Two New Modems

Universal Data Systems, Inc., a data communications manufacturer has announced a full-featured, Bell-compatible 212A modem which will be priced below present market for comparable devices.

At \$695 the UDS 212A offers a saving to data communicators utilizing full-duplex 300 and 1200 bps channels in the same system. The device is FCC certified for direct connection to

the dial-up telephone network and is fully compatible with 212As offered by Bell and other modem suppliers.



The company has also announced the addition of the Model 212 LP to their family of modems. Powered from the telephone line, this manual answer unit requires no external AC power, but offers full duplex 1200 only bps asynchronous operation. The 212 LP is FCC certified for direct connection to the dial-up network, and is compatible with the high speed 1200 bps asynchronous channel of the Western Electric 212A. Packaged in a low profile housing, the unit is designed for desk top applications. It is being offered at \$495.00 in single quantities.

Detailed technical specifications and quantity pricing may be obtained by contacting:

*Universal Data Systems,  
5000 Bradford Drive  
Huntsville, AL 35805-9990  
(205)837-8100*

## Hayden Announces Computer Literacy Package

I Speak BASIC is a machine specific computer literacy course that introduces students to

BASIC programming. The course provides student instruction in the BASIC language for the Apple, TRS-80 and PET and includes a Teacher's Manual, Student Text, and Exam Set for each machine.

Written by Aubrey Jones, I Speak BASIC is designed for teachers regardless of their knowledge of microcomputers and their programming skill.

The core of the course is the Student Text that features learning objectives, definitions and examples of key terms and BASIC concepts in class programming exercises, practices, and assignments. Each version includes chapters explaining the parts and operation of the microcomputer. Chapters cover BASIC programming topics such as mathematical operations, scientific notations, conditional and unconditional branching, input statements, loops, reading data, video display graphics, arrays and subroutines.

The Teacher's Manual provides techniques for presenting the material and emphasizing particular concepts, annotations to aid in lesson planning, suggestions for implementing the course and answers to all practice exams.

The Exam Set contains 12 quizzes on spirit duplicating masters to check student understanding and reinforce learning. The quizzes can be easily reproduced for class use. A classroom set of I Speak BASIC contains one Teacher's Manual, 20 Student Texts and one Exam Set.

Prices for all three machines are:

Student Text	\$ 7.45
Teacher's Manual	16.20
Exam Set	12.50
Classroom Set	156.25

*Hayden Book Company, Inc.  
50 Essex Street  
Rochelle Park, NJ 07662  
Toll Free (800)631-0856  
In NJ (201)843-0550*





# commodore

## SHOW SPECTACULAR

8032-32K 80 COL CRT

REG \$1495 **\$1065**

64K ADD-ON MEMORY

REG \$500 **\$395**

9000 134K SUPER PET

REG \$1995 **\$1795**

4032 32K 40 COL CRT

REG \$1295 **\$965**

4016 16K 40 COL CRT

REG \$995 **\$765**

8050-DUAL DISK 950K

REG \$1795 **\$1295**

4040-DUAL DISK 343K

REG \$1295 **\$995**

2031-SINGLE DISK 170K

REG \$695 **\$525**

C2N-CASSETTE DRIVE

REG \$75 **\$65**

4022-80 COL PRINTER

REG \$795 **\$595**

8023P-136 COL PRINTER

REG \$995 **\$849**

8300P-40CPS LTR QLT

REG \$2250 **\$1995**

8024-MANNESMAN TALLEY

REG \$1995 **\$1595**

8024L-LETTER TALLEY

REG \$2495 **\$1995**

25CPS-STARWRITER

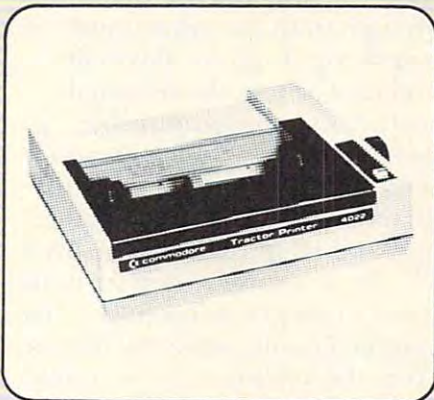
REG \$1895 **\$1445**

CBM-IEEE MODEM

REG \$279 **\$229**

VOICE SYNTHESIZER

REG \$395 **\$329**



PET TO IEEE CABLE

REG \$39.95 **\$34**

IEEE TO IEEE CABLE

REG \$49.95 **\$39**

VIC 20

REG \$299 **\$265**

VIC 1540 DISK 170K

REG \$599 **\$499**

VIC 1515 30CPS PRINTER

REG \$395 **\$349**

VIC 1011 RS 232 INTER

REG \$49.95 **\$39**

VIC 1112 IEEE INTER

REG \$99.95 **\$79**

### MATOR SHARK HARD DISK DRIVE

\*Winchester Disk  
\*Commodore DOS 2.5  
\*24 Megabyte  
**\$6475**

Strobe 100 Plotter  
\*PET Interface  
\*Software Pack  
**\$975**

SCHOOL SPECIALS  
3 for 2 Pricing  
CALL US!

20% Off on Software

*Special pricing on Atari, Apple, Digilog, Epson, Hayes Modem, Printwheels*

## PHILADELPHIA COMPUTER DISCOUNT ©

P.O. Box 170 St. Davids, PA 19087 (215)687-8540

MAIL ORDER PHONE 1-800-345-1289

PREPAID ORDERS SHIPPED FREE

VISA & MASTERCARD ADD 3%

COD - UPS

PA RESIDENTS ADD 6% SALES TAX

[www.commodore.ca](http://www.commodore.ca)



## VidCom, Inc. Announces New Product

VidCom, Inc. has developed a device which plugs into the memory channel of a Personal Computer such as the Atari 800, Apple II, PET and others, making possible the interaction of recorded video taped programs.

The VidCom System consists of a Vidram board, a VTR with connecting cable to the computer Vidram board, a personal computer with an audio program recorder, a color TV, and a computer program to interact with the VTR.

The price for the Vidram board is \$3.95; for the VCR modification, \$195.00. An Atari 800, 48K, or an Apple II, 48K with an audio cassette recorder, including the Vidram Board, a VTR such as a PANASONIC modified with wiring and cable to connect to the computer Vidram board would be \$3,595.00.

VidCom, Inc. will prepare the program instructions for the computer with direct or multiple-choice questions to be asked the student from the video taped materials supplied by customer at the average price of \$5.00 per minute running time, or from video taped materials supplied by us at the average price of \$10.00 per minute.

*VidCom, Inc.  
1234 Watchung Ave.  
Plainfield, NJ 07060  
(201) 754-2377*

## Personal Information Terminals Introduced By Tymshare

A new line of quiet, compact, personal information terminals,

with prices starting at under \$500, is being introduced by Tymshare, Inc. Called Scanset, the terminals are designed specifically for one-button information access by non-computer professionals.

The Model 410, with automatic computer log-in, sells for \$495. The Model 415, with built-in modem, automatic telephone dialer, and automatic computer log-in, is priced at \$649.

The Scanset units have the same basic abilities to communicate with computers as larger, more expensive terminals. Yet they only take about one square foot of space, and are small enough to fit inconspicuously on a desk top. Built for flexibility and ease of use, the terminals feature user programmable function keys, a 9-inch diagonal screen with 24 lines of text and 40 or 80 character line lengths, and limited graphics capability.

Both Scansets have six multi-function keys which can be programmed easily, either by the user from the keyboard, or remotely from the computer. Up to 12 user-defined tasks can be assigned to the programmable keys, giving the user easy access to host computers or frequently used data bases. They can also be programmed to handle other repetitive jobs.

The autodialer feature of the Model 415 can dial up to 36 phone numbers stored in the terminal's memory, automatically connecting the terminal to computers or data bases. The first four numbers can also be used for automatic computer log-in. In addition, this feature can also be used as an autodialer for a regular telephone. A directory of all numbers, with descriptions, is stored in the terminal and is available on-screen at the press of a key.

The user can view a full 24 lines of text at one time, with a choice of either 40 or 80 character

line lengths. Automatic scrolling takes place after the 24th line of text is filled. A 25th line appears at the bottom of the screen, immediately above the function keys, and displays easy-to-read labels for the programmable keys.

The P4 phosphor screen features a flicker-free display. The integral keyboard includes a 69-key standard layout, including four cursor control keys for up, down, left, and right movement of the cursor.



Limited graphics capability, including forms and simple line or bar graphs, is also included. A printer or other device can be connected easily to the Scanset through an intelligent peripheral port. This allows a printer to print out information either directly as it is received through the terminal from a computer, or as it appears on the terminal screen. A buffer management system prevents loss of data.

**TYMSHARE**  
20705 Valley Green Dr.  
Cupertino, CA 95014  
(408)446-6000

*New Product releases are selected from submissions for reasons of timeliness, available space, and general interest to our readers. We regret that we are unable to select all new product submissions for publication. Readers should be aware that we present here some edited version of material submitted by vendors and are unable to vouch for its accuracy at time of publication.*

©



# Lyc0 Computer Marketing & Consultants

We Specialize in Quality, Knowledge, Service, and Microcomputers

717-435-5197



A Warner Communications Company

© 1981 ATARI INC

## MAY SPECIALS

ATARI 800 16K ..... \$675.00

ATARI 400 16K ..... \$329.00

ATARI 825 PRINTER ..... \$585.00

32K MEMORY BOARDS ..... \$149.00

16K MEMORY BOARDS ..... \$75.00

for ATARI 400 & 800 with 1 year warranty!!

### PROGRAMMING SOFTWARE

CX4101	INVITATION TO PROG 1 ....	\$ 21.00
CX4106	INVITATION TO PROG 2 ....	\$ 24.00
CX4117	INVITATION TO PROG 3 ....	\$ 24.00
CX8126	MICROSOFT BASIC .....	\$ 69.00
CXL4002	ATARI BASIC CART .....	\$ 45.00
CX405	PILOT EDUCATOR .....	\$109.00
CX4018	PILOT HOME .....	\$ 65.00
CXL4003	ASSEMBLER EDITOR .....	\$ 45.00
CXL4015	TELELINK .....	\$ 24.00

### EDUCATIONAL SOFTWARE

CX4112	STATES & CAPITALS .....	\$ 12.00
CX4114	EUROPEAN COUNTRIES ...	\$ 13.00
CX4108	HANGMAN .....	\$ 13.00
CX4102	KINGDOM .....	\$ 13.00
CX4121	ENERGY CZAR .....	\$ 13.00
CX4123	SCRAM .....	\$ 19.00
CX4119	FRENCH .....	\$ 45.00
CX4125	SPANISH .....	\$ 45.00
CXL4007	MUSIC COMPOSER .....	\$ 45.00
CX4110	TOUCH TYPING .....	\$ 19.00
CX4103	STATISTICS .....	\$ 19.00

CX 404	WORD PROCESSOR .....	\$129.00
CX406	PERSONAL FINANCE .....	\$ 62.00
	TECHNICALUSERSNOTES...	\$ 25.00

### ACCESSORIES

CX30	PADDLES .....	\$ 18.00
CX40	JOYSTICKS .....	\$ 18.00
CX853	16 K RAM .....	\$ 85.00

### ATARI HARDWARE

810	DISK DRIVE .....	\$455.00
410	CASSETTE RECORDER ....	\$ 75.00
850	INTERFACE .....	\$164.00
830	PHONE MODEM .....	\$159.00
825	PRINTER .....	\$585.00

### ATARI PACKAGES

CX481	ENTERTAINER .....	\$ 85.00
CX482	EDUCATOR .....	\$125.00
CX483	PROGRAMMER .....	\$ 55.00
CX484	COMMUNICATOR.....	\$325.00

### ENTERTAINMENT SOFTWARE

CXL4012	MISSILE COMMAND .....	\$ 35.00
CXL4013	ASTEROIDS .....	\$ 35.00
CXL4011	STAR RAIDERS .....	\$ 37.00
CXL4006	SUPER BREAKOUT .....	\$ 35.00
CXL4009	CHESS .....	\$ 35.00
CXL4004	BASKETBALL .....	\$ 30.00
CXL4005	VIDEO EASEL .....	\$ 30.00
CX4015	BLACKJACK .....	\$ 13.00
CX4107	BIORHYTHM .....	\$ 13.00
CX4111	SPACE INVADERS CASS ...	\$ 17.00
CXL4008	SPACE INVADERS ROM ....	\$ 35.00

WE CARRY MANY OTHER LINES OF  
MICROCOMPUTERS YOU CAN  
CALL FOR PRICES ON:

At Lyc0 Computers we offer our expert services to help customers make their first computer purchase, schools establish a computer program, or evaluate multiterminal systems.

to help evaluate your needs or  
if you wish to make a purchase

CALL US AT 717-435-5197

LYCO COMPUTERS

P.O. BOX 10

COGAN STATION, PA 17728

IN STOCK ORDERS

SHIPPED SAME DAY

WE PAY FREIGHT ON PRE-PAID ORDERS

NO EXTRA CHARGE

ON C.O.D. ORDERS

[www.commodore.ca](http://www.commodore.ca)



# CAPUTE!:

## Corrections And Amplifications

1. "Renumbering An Appended Routine Only," **COMPUTE!**, January, 1982, #20, pg. 144: the direct BASIC command in column one should read  $AD = (PEEK(42) + PEEK(43) * 256) - 2$ :  $AH\% = AD / 256$

2. "Starfight3," **COMPUTE!**, March, 1982, #22, pg. 112: [These changes and hints were sent in by the author.] Change line 590 to  $IF T > 0 \text{ AND } KC > 0 \text{ THEN } 360$ . Under item five of the program directions: a. 120 should be 180, b. TIS should read  $TI\$$ , 450 is 410, 545 is 500, and 1530 is 1300.

This program was written before any memory expansions were available. To use Starfight3 on a VIC with expanded memory, delete lines 30

through 100, substitute an "E" for the "@" in line 920 and substitute a "K" for the "#" in line 980. There will be no little ships now, but the program will run.

If you are receiving an OUT OF MEMORY error message, it is probably due to lines 30-100. Once the program has been RUN, these lines set aside memory that cannot be touched. So, if you make a typo and try to rerun the program, you will run out of memory. The following procedure will let you make corrections:

1. Make the correction.
1. SAVE the program.
3. Turn the VIC off, then back on.
4. LOAD the corrected program.
5. RUN.

©

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



## ATARI SPECIAL PACKAGE

800 COMPUTER WITH DISK DRIVE \$ 1,119.00

### ATARI SOFTWARE

Invt. to programg. 2	24.95
Invt. to programg. 3	24.95
Microsoft Basic	74.95
Pilot	64.95
Personal Finance	59.95
Atari Word Processor	119.95
Missile Command	34.95
Asteroids	34.95
Caverns of Mars	31.95
LJK Letter Perfect	119.95
LJK Mailmerge/Utility	24.95
LJK Data Perfect	79.95
OS/A+	69.00
Basic A+	69.00
Synapse Filemgr. 800	79.95
Synapse Disk Mgr.	24.95
Wiz & Princess	26.95
The Next Step	31.95

Softporn Adventure	24.95
Crush, Crumble, Chomp	25.95
Rescue at Rigel	25.95
Compu-Math/Decimals	31.95
Compu-Math/Fractions	31.95
Compu-Read 3.0	24.95
LISP 2.0	119.95
Text Wizard	79.95
AliBaba & 40 Thieves	25.95
Q5 Forth	69.00
Match Racers	25.95
Pathfinder	29.95
Ghost Hunter	26.95
16K RAM	69.95
32K RAM	139.95

### COMPUTERS

Altos 8000-2	2895.00
Northstar Advantage	call
Zenith Z-89	2149.00

### PRINTERS

Epson MX-80	459.00
Okidata 82-A	449.00
NEC 8023A	489.00
Diablo 630	2195.00
NEC 3510	1795.00

### MODEMS

Hayes Micromodem II	269.00
Hayes Smartmodem	229.00
Novation DCAT	159.00

### MONITORS

NEC 12" Green	159.00
NEC 13" Color	329.00
Zenith 12" Green	119.00

### MEDIA

Verbatim	25.00/box
Dysan	37.00/box

**uni** **COMM** **INC.**

THE PURCHASING SERVICE FOR HOME & BUSINESS COMPUTERS  
 1247 LINCOLN BLVD. F Dept. I SANTA MONICA, CA. 90401 (213) 451-8089

FREE CATALOG... offering the most complete line of computer products in the country. Your satisfaction is backed by our 30 day guarantee. ALL PRODUCTS DISCOUNTED UP TO 30%. COMPARE PRICES & KNOWLEDGE. No minimum orders. Please add 3% for shipping, handling & insurance. Showroom by appt. only Monday to Friday 9:30-5:30. (PRICES SUBJECT TO CHANGE).



# the SOFTWARE connection For Atari Only

## EXTRA SPECIAL SPECIALS

### K-razy Shootout Retail \$49.95 Now \$37.50

K-BYTE breaks into the market with this innovative game. ROM cartridge 16K.

### Pathfinder Retail \$39.95 Now \$25.95

Blast your way through the maze filled with nuclear waste and radioactive monsters in this never before released game by Gebelli Software.

### Raster Blaster Retail \$29.95 Now \$22.50

By Bill Budge. Detailed simulation of pinball, with full-color hi-res graphics, animation, and sound effects.

### Apple Panic Retail \$29.95 Now \$22.50

One of Apple's most exciting games is now available for your Atari.

## GAMES

	Retail	Now
Mouskattack ..... D 32K	\$34.95	\$27.95
Star Blazer ..... D	31.95	26.50
Empire of the Overmind C 40K	30.00	24.00
The Datestones		
of Ryn ..... C/D 32K*	19.95	15.95
Rescue at Rigel ..... C/D 16K*	29.95	23.95
Temple of		
Apshai Dujonquest .. C/D 32K*	39.95	31.95
Star Warrior ..... C/D 32K*	39.95	31.95
Crush, Crumble		
& Chomp ..... C/D 32K*	29.95	23.95
Invasion Orion ..... C/D 32K*	24.95	19.95
Track Attack ..... D	29.95	23.95
Computer Acquire .... C 32K	20.00	16.00
Lords of Karma ..... C 40K	20.00	16.00
Reversi ..... C 8K	19.95	15.95
Deflection ..... C 8K	14.95	11.95
Sunday Golf ..... C 16K	14.95	11.95
Adventure Series ..... C 24K	19.95	15.95
Mountain Shoot ..... C 16K	14.95	11.95
Galactic Trader ..... C 32K	19.95	15.95
Poker Solitaire ..... C 8K	14.95	11.95
Galactic Empire ..... C 32K	19.95	15.95
Star Trek 3.5 ..... C 32K	19.95	15.95
Gomoku ..... C 8K	19.95	15.95
Ricochet ..... C 16K*	19.95	15.95
David's Midnight Magic D	34.95	27.95
Stone of Sisyphus ..... D 40K	34.95	27.95
Ali Baba ..... D 32K	32.95	26.50

### The following require joystick controllers

Angle Worm ..... C 8K	14.95	11.95
Lunar Lander ..... C 24K	14.95	11.95
Stocks and Bonds ..... D 40K*	25.00	20.00
Cypher Bowl ..... C/D 16K	49.95	39.95
Kayos ..... C/D 8K	34.95	27.95
Ghost Hunter ..... C 16K	29.95	23.95
Pool 1.5 ..... D 16K	34.95	27.95
Ghosthunter ..... D 16K	34.95	27.95
Jaw Breaker ..... D 16K	29.95	23.95
Andromeda 2 ..... D 24K	34.95	27.95
Match Racer ..... D 16K	29.95	23.95

COD and Chargecard orders may call (916) 989-3174. Subject to stock on hand.

Prices subject to change.

APPLE USERS CHECK OUR CATALOG

## MISCELLANEOUS

	Retail	Now
Visicalc** D 32K	\$250.00	\$189.00
A number one best seller by Apple® now available for your Atari.		

Letter Perfect** ..... D 16K	149.95	119.95
LJK Utility		
Mail/Merge** ..... D 16K	29.95	22.50
Data Perfect** ..... D 32K	99.95	79.95
Text Wizard** ..... D 32K	99.95	79.95
Datasm-65** ..... D 48K	149.95	119.95
Edit 6502*** ..... R 24K	199.95	160.00
The Next Step*** ..... D 32K	39.95	31.95

### The following require joystick controllers

Character		
Generator*** ..... D 24K*	19.95	15.95
Lisp 2.0*** ..... D 48K	149.95	119.95

(C) Cassette Tape (D) Diskette (R) ROM Cartridge

\*Requires Atari Basic

\*\*\*No printer option

\*\*Printer optional

### We carry complete lines from the following companies:

AVALON HILL GAME COMPANY • COMPUTER MAGIC, LTD. • DATASOFT • EDU-WARE • GEBELLI • INNOVATIVE DESIGN • K-BYTE • LJK • ON-LINE SYSTEMS • QUALITY SOFTWARE • SPECTRUM COMPUTERS • STRATEGIC SIMULATIONS • SYNERGISTIC • UNITED SOFTWARE OF AMERICA • VERSA COMPUTING • VOYAGER SOFTWARE • ADVENTURE INTERNATIONAL • ARCADE PLUS • ARTSCI • AUTOMATED SIMULATIONS • ATARI

Catalog free with any order or send \$2 postage and handling to

## THE SOFTWARE CONNECTION

5133 Vista Del Oro Way  
Fair Oaks, CA 95628

MAIL ORDERS: For fast delivery, send certified check, money orders or Visa or MasterCard number and expiration date, for total purchase price plus 1% or \$2 minimum for postage and handling. Add \$5 for shipments outside the continental U.S.

California residents add 6% sales tax.



# COMPUTE!'s Listing Conventions

Many of the programs which are listed in **COMPUTE!** use special keys (cursor control keys, color keys, etc.) To make it easy to tell *exactly* what should be typed in when copying a program into the computer, we have established the following listing conventions.

## For The Atari

All the editing and cursor control characters are spelled out and surrounded by brackets in the program listings: {CLEAR} for "clear screen." Other characters, such as CTRL-T (the "ball" character) will be listed as the "normal" character, but it will be within brackets: {T}. A series of identical control characters will be indicated by a number within the brackets: {3DOWN} means type ESC CURSOR-DOWN three times; {12R} would mean type CTRL-R twelve times. Remember to press the ESC (escape) key before each cursor control key. If you should see {ESC} itself in a program listing, you would press ESC *twice*.

Two of the control characters, {=} and {-}, should be shifted. Any reverse field text will be enclosed within vertical lines. (In other words, any time you see a vertical line within a program listing in **COMPUTE!**, press the Atari logo key {⌘}.)

## Atari Conventions

```
{CLEAR}= SHIFT-← (Clear Screen)
{UP}= CTRL-minus (Cursor Up)
{DOWN}= CTRL-equals (Cursor Down)
{LEFT}= CTRL-plus (Cursor left)
{RIGHT}= CTRL-asterisk (Cursor right)
{BACK S}= BACK S (Back space)
{DELETE}= CTRL-DELETE (Delete character)

{DEL LINE}= SHIFT-DELETE (Delete Line)
{INSERT}= CTRL-INSERT (Insert character)

{INS LINE}= SHIFT-INSERT (Insert line)
{ESC}= ESC (ESCAPE key pressed twice)
{TAB}= TAB (Tab key)
{CLR TAB}= CTRL-TAB (Clear tab setting)
{SET TAB}= SHIFT-TAB (Set tab stop)
{BELL}= CTRL-2 (Ring buzzer)
```

## For PET/CBM/VIC

Generally, any PET/CBM/VIC program listings will contain bracketed words which spell out any special characters: {DOWN} would mean to press the cursor-down key; {3DOWN} would mean to press the cursor-down key three times.

To indicate that a key should be *shifted* (hold down the SHIFT key while pressing the other key), the key would be underlined in our listing. For example, S would mean to type the S key while holding the shift key. This would result in the "heart" graphics symbol appearing on your screen.

Sometimes in a program listing, especially within quoted text when a line runs over into the next line, it is difficult to tell where the first line ends. How many times should you type the SPACE bar? In our convention, when a line breaks in this way, the ~ symbol shows exactly where it broke. For example:

```
100 PRINT "TO START THE GAME ~
    YOU MAY HIT ANY OF THE KEYS
    ON YOUR KEYBOARD."
```

shows that the program's author intended for you to type two spaces after the word *GAME*.

## For The Apple

Programs listed as "Microsoft" are written for the PET/CBM,

Apple, OSI, etc. Although the programs are general in nature, you may need to make a few changes for them to run correctly on your Apple. Microsoft BASIC programs written for the PET/CBM sometimes contain special cursor control characters. The following table shows equivalent Apple words. Notice that these Apple commands are *outside* quotations (and even separate from a PRINT statement). PRINT "[RVS]YOU WON" becomes INVERSE: PRINT "YOU WON":NORMAL

```
[CLEAR] (Clear Screen) HOME
[DOWN] (Cursor down)
    Apple II + : Call -922
    POKE 37,PEEK(37)+(PEEK(37)<23)
[UP] (Cursor up)
    POKE 37,PEEK(37)-(PEEK(37)>0)
[LEFT] (Cursor left) PRINT CHR$(8);
[RIGHT] (Cursor right)
    PRINT CHR$(21)
```

[RVS] (Inverse video on. Turns off automatically after a carriage return. To be safe, turn off inverse video after the print statement with NORMAL unless the PRINT statement ends with a semicolon.)

INVERSE

[OFF] (Inverse video off) NORMAL

Shifted characters can represent either graphics characters or uppercase letters. If within text, just use the non-shifted character, otherwise substitute a space. Some "generalized" programs contain a POKE such as POKE 59468,14. Omit these from the program when typing it in. One final note: you will probably want to insert a question mark or colon within an INPUT prompt. PET/CBM and many other BASICs automatically print a question mark:

```
INPUT "WHAT IS YOUR NAME?";N$
    becomes
INPUT "WHAT IS YOUR NAME?";N$
```

## All Commodore Machines

Clear Screen {CLEAR}	Cursor Left {LEFT}
Home Cursor {HOME}	Insert Character {INST}
Cursor Up {UP}	Delete Character {DEL}
Cursor Down {DOWN}	Reverse Field On {RVS}
Cursor Right {RIGHT}	Reverse Field Off {OFF}

## VIC Conventions

Set Color To Black {BLK}	Function Two {F2}
Set Color To White {WHT}	Function Three {F3}
Set Color To Red {RED}	Function Four {F4}
Set Color To Cyan {CYN}	Function Five {F5}
Set Color To Purple {PUR}	Function Six {F6}
Set Color To Green {GRN}	Function Seven {F7}
Set Color To Blue {BLU}	Function Eight {F8}
Set Color To Yellow {YEL}	Any Non-implemented
Function One {F1}	Function {NIM}

## 8032/Fat 40 Conventions

Set Window Top {SET TOP}	Erase To Beginning {ERASE BEG}
Set Window Bottom {SET BOT}	Erase To End {ERASE END}
Scroll Up {SCR UP}	Toggle Tab {TGL TAB}
Scroll Down {SCR DOWN}	Tab {TAB}
Insert Line {INST LINE}	Escape Key {ESC}
Delete Line {DEL LINE}	



# commodore

Check Our New Lowest Prices!



## EQUIPMENT:

- VIC-20 (Includes RF Modulator)
- CBM 4016 CPU (40 Col. Screen, 16K RAM)
- CBM 4032 CPU (80 Col. Screen, 32K RAM)
- CBM 8032 CPU (80 Col. Screen, 32K RAM)
- CBM 8096 CPU (80 Col. Screen, 96K RAM)
- CBM Micro Mainframe (Super PET)
- CBM Single Disk Drive (170K per 5 1/4 Diskette)
- CBM 2031 Single Disk Drive (170K per 5 1/4 Diskette)
- CBM 4040 Dual Disk Drive (170K per 5 1/4 Diskette)
- CBM 4050 Dual Disk Drive (170K per 5 1/4 Diskette)
- VIC Tractor Feed Printer (80 Col., 150 CPS)
- CBM 4022 Tractor Feed Printer
- CBM 4022 Tractor Matrix Printer
- 8023P Dot Matrix Printer (7x7 Matrix)
- 8024-7 (7x9 Matrix) Printer
- Tally 8024-9 (7x9 Matrix) Printer
- 8300P Letter Quality Modern
- CBM 8010 Phone Deck (New Style)
- CBM C2N Cassette Deck (New Style)
- CBM CPU/IEEE Cable
- CBM IEEE/IEEE Cable

265  
790  
990  
1090  
1590  
1690  
470  
570  
990  
1340  
325  
625  
790  
1275  
1425  
1790  
225  
65  
40

## SOFTWARE:

- Wordcraft 80 Wordprocessor
- Wordpro 4+ Wordprocessor
- OZZ Data Base System
- Visicalc
- Tax Preparation System
- IRMA Information Retrieval
- Dow Jones Portfolio
- The Manager

# MART

3300 Buckeye Rd., N.E.  
Suite 641  
Atlanta, Ga. 30341

404-458-0729  
Call 9 AM-5 PM EST M-F

## PUBLICATIONS:

- CBM User Guide
- CBM Basic 4.0 Reference Manual
- CBM Disk Manual
- CBM Printer Manual
- MOS Hardware Manual
- MOS Programming Manual
- The PET Revealed
- Library of PET Subroutines
- Commodore Software Encyclopedia

43.95

43.95

## VIC SOFTWARE:

- VT 106A Recreation Six Pack  
Includes Car Chase, Blue Meanies, Space Math, Slither/Super Slither, Biorythm Capability
- VT 107A Home Utility Six Pack  
Includes Personal Finance I, Personal Finance II, VIC Typewriter, Expense Calendar, Loan & Mortgage Calculator, Home Inventory
- VIC 1901 VIC Super Alien
- VIC 1904 Super Slot
- VIC 1907 Super Lander
- VIC 1908 Draw Poker

29.95  
29.95  
29.95  
29.95

## "CALL ABOUT SUPPLIES"

COMING SOON:  
More VIC Peripherals and Software  
CBM 8250 Dual Disk Drive  
Data Acquisition and Control Devices

All items insured  
COD - UPS  
Prepaid Orders Shipped Free  
In stock items shipped within 48 hours  
MASTERCARD OR VISA ADD 3%  
GA RESIDENTS ADD 4% SALES TAX

## WORD PROCESSING SPECIAL

- CBM 8300 (Letter Quality, DIABLO 630, 40 CPS, Bi-directional)
- WORDCRAFT 80  
• Includes all cables

CBM 8032  
CBM 4040

\$4200

CALL ABOUT EQUIPMENT  
AND/OR SOFTWARE  
SUBSTITUTIONS

www.commodore.ca



# Advertisers Index

AB Computers	58,59,155	Krell Software Corp.	87
Aardvark Technical Services, Ltd.	125	Leading Edge Products, Inc.	IBC
Abacus Software	56	LemData Products	118
Academy Software	137	Level Limited	173
Advanced Computing Enterprises	85,131	Lyco Computer Marketing & Consultants	187
Adventure International	20,21	MED Systems Software	159
Alternate Reality Software	117	MIS	81
Amplify, Inc.	55	Macrotronics, Inc.	134
Artworx Software Company	51	Madison Computer	55
Atari, Inc.	6,7,55	Micro Business World Inc.	113
Automated Simulations Inc.	43	Micro Printer Marketing	109
BBI	178	Micro-Ed, Inc.	73
Batteries Included	161	Micro Technical Products, Inc.	173
John Bell Engineering, Inc.	57	Micrograms, Incorporated	89
The Bottom Line	64	Micromint, Inc.	95
R. J. Brachman Associates, Inc.	121	MicroSpec Ltd.	75
C-Mart	191	Microsoft Consumer Products	4
C E Software	82	Midwest Micro Associates	137
CFI	155	Miles Computing	100
CGRS Microtech	57	Mosaic Electronics	35,127
CMS Software Systems	2,3	Mountain Computer, Inc.	IFC
Cansoft Data Inc.	31	New England Electronics Company	44,45
Comm*Data Systems, Inc.	99	Nufekop	99
Commodore Computer Systems	BC	Olympic Sales Company	121
CompuServe	35	On Line Software	118
COMPUTE! Back Issues	39	On-Line Systems	19
COMPUTE! Books	25	Optimal Technology, Inc.	57
COMPUTE! Subscriptions	10	Optimized Data Systems	121
Computer Country	180	Optimized Systems Software, Inc.	68,129
Computer Mail Order	176,177	Oryx Software	71
ComputerMat Software	117,137	PSK Evaluation Service	50
Computertime, Inc.	100	Pacific Exchanges	27,50,175,178
Computer Service Center	77	Philadelphia Computer Discount	185
Comquest	41	Poquettes	131
Connecticut microComputer, Inc.	95	Pretzelland Software	70
Cosmic Computers Unlimited	85	Professional Software, Inc.	1,9
Creative Software	16	Program Design, Inc.	103
Data Equipment Supply Corp.	115	The Program Store	119
Data Pro Research Corporation	Business Reply Cards, 192	The Programmer's Institute	35
Data Pro Research Corporation	Business Reply Cards, 192	Quantum Data, Inc.	142
Data Transforms, Inc.	134	Qube International	142
Don't Ask Software	130	RAM/RBC Systems Inc.	62
Dream Print Homes, Inc.	104	RAR-Tech	62
Dunham Software & Consulting Co.	55	Retcom Systems Inc.	157
Dynacomp Inc.	36,37	Howard W. Sams Co. Inc.	33
ECX Computer Company	117	Santa Cruz Educational Software	61
ETC Corporation	27	Scientific Software	62
Eastern House Software	111	Skyles Electric Works	91,152
Eclectic Systems Corporation	179,181	The Software Connection	189
Execom Corporation	117	Software Galore	75
French Silk	56	Software Street	183
Gamma Software	82	Street Electronics Corporation	174
Gebelli Software, Inc.	15	Sunrise Software	93
H W Electronics	145	Swiftly Software, Inc.	144
Hayes Microcomputer Products, Inc.	13	Syncro, Inc.	29
High Country Microsystems	95	T.H.E.S.I.S.	93
Horizon Simulations Inc.	49	TIS Inc.	30
Huntington Computing	107	T'Aide Software Company	108
ICS Micro Wholesale	145	Tiny Tek, Inc.	85
IDSi	17	Toronto PET Users Group	75
Image Marketing Corporation	85	Unicom Inc.	188
InHome Software	71	United Microwave Industries, Inc.	53
Interlink Inc.	27	University Microfilms International	79
JAVJR Co.	175	VERVAN Software	118
Jini Micro-Systems, Inc.	11	Vixel, The Code Works	99
K-Byte	23	Wildfire Publishing	30
Kelly's Computing	93	Wunderware	82



# The 6502 Resource Magazine **COMPUTE!** PET-ATARI-APPLE OSI-KIM-SYM-AIM

For Fastest Service,  
 Call Our **Toll-Free**  
 US Order Line  
**800-345-8112**  
 In Pennsylvania call 800-662-2444

My Computer Is:

☐ PET ☐ Apple ☐ Atari  
☐ KIM ☐ SYM ☐ AIM  
☐ Don't yet have one...

☐ \$20.00 One Year US Subscription  
☐ \$36.00 Two Year US Subscription  
☐ \$54.00 Three Year US Subscription

☐ OSI  
☐ Other \_\_\_\_\_

(Readers outside of the US, please  
 see our foreign readers subscription  
 card or inquire for rates).

Name \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_

☐ Payment Enclosed ☐ VISA  
☐ MasterCard ☐ American Express

Account No. \_\_\_\_\_ Expires \_\_\_\_\_ / \_\_\_\_\_

5 6 7 8 9 10 11 12

## The Editor's Feedback:

Computer: ☐ PET ☐ ATARI ☐ APPLE ☐ VIC-20 ☐ OSI ☐ OTHER \_\_\_\_\_

Are you a **COMPUTE!** Subscriber? ☐ Yes ☐ No I would like to see:

<input type="checkbox"/> More <input type="checkbox"/> Fewer	Specific applications programs.	<input type="checkbox"/> More <input type="checkbox"/> Fewer	Games.
<input type="checkbox"/> More <input type="checkbox"/> Fewer	BASIC programs.	<input type="checkbox"/> More <input type="checkbox"/> Fewer	Reviews of game software.
<input type="checkbox"/> More <input type="checkbox"/> Fewer	Machine language programs.	<input type="checkbox"/> More <input type="checkbox"/> Fewer	Reviews of business software.
<input type="checkbox"/> More <input type="checkbox"/> Fewer	Tutorials.	<input type="checkbox"/> More <input type="checkbox"/> Fewer	Reviews of educational software.
<input type="checkbox"/> More <input type="checkbox"/> Fewer	Educational articles.	<input type="checkbox"/> More <input type="checkbox"/> Fewer	Reviews of hardware.
<input type="checkbox"/> More <input type="checkbox"/> Fewer	Detailed explanations of programs.	<input type="checkbox"/> More <input type="checkbox"/> Fewer	

What do you like best about **COMPUTE!**?

What do you like least?

5 6 7 8 9 10 11 12

## SELECTED EDP REPORTS

Characteristics, prices, user evaluations, comparisons

<input type="checkbox"/> All About Personal Computers	\$29
<input type="checkbox"/> User Ratings of Computer Systems	\$25
<input type="checkbox"/> European User Ratings of Computer Systems	\$25
<input type="checkbox"/> Word Processing Systems User Ratings	\$19
<input type="checkbox"/> All About 369 Small Business Computers	\$15
<input type="checkbox"/> All About 262 Minicomputers	\$15
<input type="checkbox"/> All About 190 Microcomputers	\$15
<input type="checkbox"/> All About 148 Microprocessors	\$15
<input type="checkbox"/> User Ratings of Proprietary Software Packages	\$25
<input type="checkbox"/> A Buyer's Guide to Database Management Systems	\$15
<input type="checkbox"/> All About CAD/CAM	\$25
<input type="checkbox"/> All About 260 Alphanumeric Display Terminals	\$15
<input type="checkbox"/> All About Data Communications Facilities	\$15
<input type="checkbox"/> All About Japanese Computers	\$29
<input type="checkbox"/> All About 112 Word Processing Software Packages	\$15
<input type="checkbox"/> All About 142 Teleprinter Terminals	\$15
<input type="checkbox"/> All About 400 Modems	\$15
<input type="checkbox"/> All About Winchester Disk Drives	\$15
<input type="checkbox"/> All About 150 Word Processors	\$19
<input type="checkbox"/> Directory of Over 1000 Suppliers	\$29

YES Please send me the EDP Reports I have checked.

Name \_\_\_\_\_

Title \_\_\_\_\_

Company \_\_\_\_\_

Phone \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_

State \_\_\_\_\_ Zip \_\_\_\_\_

☐ Check enclosed ☐ Bill me, add \$3.25 handling.

DRG-905A

**datapro** DATAPRO RESEARCH CORPORATION  
 1805 Underwood Blvd., Delran, NJ 08075

name	title
company	phone
address	
city	state zip

FREE with this report, Datapro's latest catalog of feature reports on EDP and office systems, selected from our monthly updated information services.

YES Please send Datapro's \$29 report **All About Personal Computers**

**datapro**

### 16 MOST POPULAR PERSONAL COMPUTERS

Datapro's latest report — **All About Personal Computers** — provides valuable information that will help you compare and evaluate 16 of the most popular personal computers including: Radio Shack's TRS-80 Model II, the IBM Personal Computer, the Xerox 800, the NEC PC 8000, and the Hewlett-Packard 85. Over 70 fact-filled pages in a convenient standardized format let you easily compare data on models, capabilities, configurations and applications, popular options, warranties, prices, retail availability and discounts. In addition, you'll find detailed characteristics of each major peripheral and software product the vendor provides for his system.

**All About Personal Computers** also contains extensive directories on nearly 200 software vendors, over 300 peripheral equipment vendors, 34 system vendors, and 19 periodicals dedicated to users of personal computers.

The report also defines personal computers, tracks their development, details their use, outlines their current and projected markets, and even takes a look at future trends.

If you are considering buying a personal computer — for home entertainment or work — you owe it to yourself to save time and money. Compare the 16 most popular in **All About Personal Computers**. Available for only \$29.

**All About Personal Computers** Only \$29  
 Newly updated for 1982 — the industry's first unbiased, objective report on the 16 most popular personal computers.

[www.commodore.ca](http://www.commodore.ca)





CMP0582

**BUSINESS REPLY CARD**

FIRST CLASS / PERMIT NO. 178 / DELRAN, NJ

POSTAGE WILL BE PAID BY ADDRESSEE

**datapro**

DATAPRO RESEARCH CORPORATION  
1805 Underwood Boulevard  
Delran, NJ 08075

NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES



CMP0582

**BUSINESS REPLY CARD**

FIRST CLASS / PERMIT NO. 178 / DELRAN, NJ

POSTAGE WILL BE PAID BY ADDRESSEE

**datapro**

DATAPRO RESEARCH CORPORATION  
1805 Underwood Boulevard  
Delran, NJ 08075

NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES



**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 236 BROOMALL, PA

POSTAGE WILL BE PAID BY ADDRESSEE

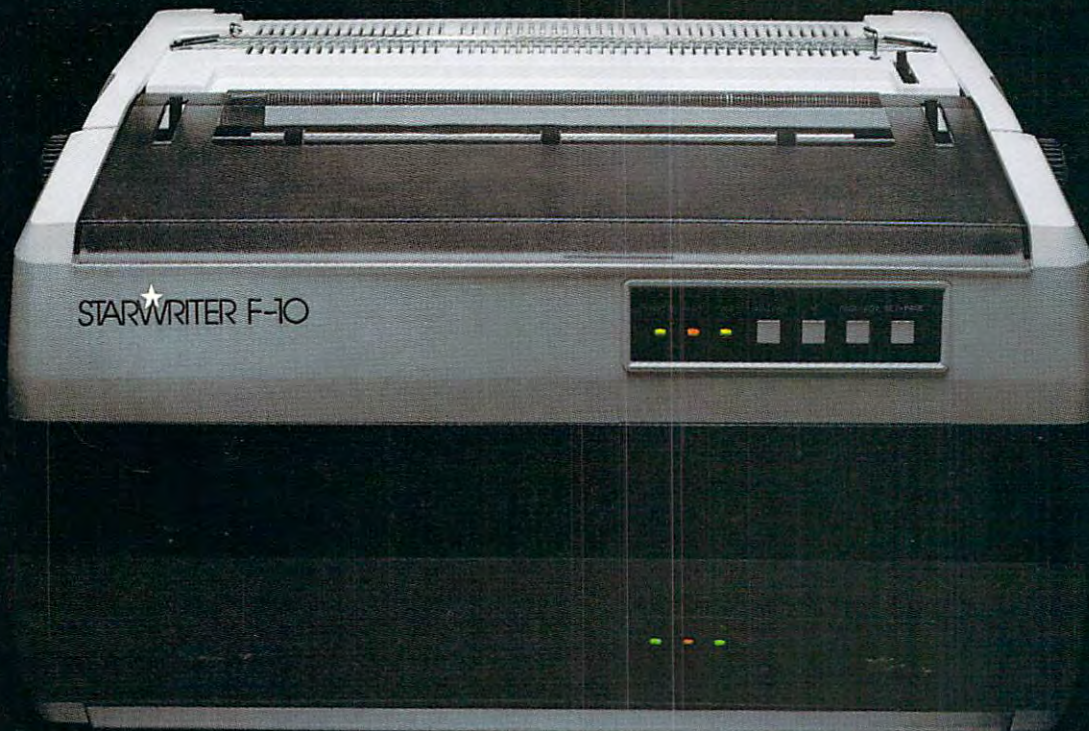
**COMPUTE! Magazine**  
515 Abbott Drive  
Broomall, PA 19008

Place  
Stamp  
Here

**COMPUTE! Magazine**  
Post Office Box 5406  
Greensboro, NC 27403



# SAVE 6 INCHES, 10 POUNDS, AND \$800.



On the new, slicked-up, trimmed-down Starwriter F-10.

It's C. Itoh's latest generation of letter-quality printers.

It cranks out flawless copy at 40 cps; and its full 15" carriage lets it double in brass for both letter processing and business applications. You can plug it into almost any micro on the market (serial or parallel) simply by plugging it in. And then make it keep on trucking with inexpensive, easily available Diablo com-

patible daisy wheels and ribbons.

In its serial mode, it can print just about anything (including boldface, underlines, subscripts and superscripts), and snap the carriage back to start the next line in less than a second. In its line mode, it prints in both directions, for even faster throughput.

(While making about as much noise as a cat walking on Kleenex.)

It's a nice, portable 30 pounds—about 10 pounds

lighter than the Starwriters before it. And it stands exactly as tall (or precisely as small) as a dollar bill.

Speaking of which:

Incredibly, the Starwriter F-10 sells for about the same preposterously low price as its predecessors. Which is to say, about \$800 less than a lot of other printers that don't even come close to measuring up. Or even better...

Measuring down.

*Distributed Exclusively by Leading Edge Products, Inc., 225 Turnpike Street, Canton, Massachusetts 02021. Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.*

**LEADING  
EDGE.**





# COMMODORE VIC-20®

## "THE WONDER COMPUTER OF THE 1980s. UNDER \$300."

—WILLIAM SHATNER

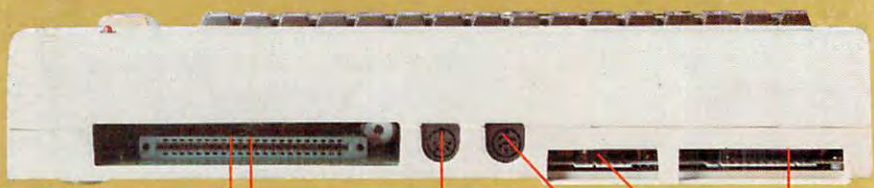
"The best computer value in the world today. The only computer you'll need for years to come."



JOYSTICK  
PADDLE  
LIGHTPEN

ON/OFF SWITCH

POWER  
PACK



PLUG-IN  
PROGRAMS

3K/8K/16K  
MEMORY  
EXPANSION

PLUG-IN  
PROGRAM

EXPANSION  
MODULE

RF  
MODULATOR

TELEVISION  
OR MONITOR

VIDEO CABLE

SINGLE  
DISK DRIVE

PRINTER

COMMODORE  
DATACASSETTE

MODEM FOR  
TELEPHONE AND  
TELECOMPUTING

### VIC-20® VS. OTHER HOME COMPUTERS

Product Features	Commodore VIC-20	Atari® 400™	TI® 99/4A	TRS-80® Color Computer
Price*	\$299.95	\$399.00	\$525.00	\$399.00
Maximum RAM Memory	32K	16K	48K	32K
Keyboard Style	Full-Size Typewriter Style	Flat Plastic Membrane	Full-Size Typewriter Style	Calculator Style
Number of Keys	66	57	48	53
Programmable Function Keys	4	0	0	0
Graphic Symbols On Keyboard	62	0	0	0
Displayable Characters	512	256	192	256
Microprocessor	6502	6502	TI990	6809
Accessible Machine Language	YES	YES	NO	YES
Upper/Lower Case Characters	YES	YES	NO	NO
Operates with all Peripherals (Disk, Printer and Modem)	YES	NO	YES	YES
Full Screen Editor	YES	YES	YES	NO
Microsoft Basic	Standard	N/A	N/A	\$ 99.00
Telephone Modem	\$109.95	\$399.95	\$450.00	\$154.95

\*Manufacturer's suggested retail price Jan. 1, 1982



Read the chart and see why COMPUTE! Magazine<sup>1</sup> calls the VIC-20 computer "an astounding machine for the price." Why BYTE<sup>2</sup> raves: "...the VIC-20 computer unit is unexcelled as a low-cost consumer computer." Why Popular Mechanics<sup>3</sup> says "...for the price of around \$300, it's the only game in town that is more than just a game." And why ON COMPUTING INC.<sup>4</sup> exclaims: "What is inside is an electronic marvel... if it sounds as if I'm in love with my new possession, I am."

The wonder computer of the 1980s. The VIC-20 from Commodore, world's leading manufacturer of a full range of desktop computers. See the VIC-20 at your local Commodore dealer and selected stores.

1 April '81 issue 2 May '81 issue 3 November '81 issue 4 Fall '81 issue

Commodore Computer Systems  
681 Moore Rd., King of Prussia, PA 19406  
Canadian Residents: Commodore Computer Systems  
3370 Pharmacy Ave., Agincourt, Ont., Canada, M1W 2K4

Please send me more information on the VIC-20.

Name \_\_\_\_\_  
Address \_\_\_\_\_  
City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_  
Phone \_\_\_\_\_

[www.commodore.ca](http://www.commodore.ca)

