PET/CBM & VIC OWNERS! Utilities & Games

"Probably the best-documented programs I've seen for PET/CBM."
Robert Baker
Microcomputing
September 1981

"The strongest points of this system are its unsurpassed documentation and its human engineering."
Ralph Bressler, The Paper
Nov/Dec. 1981

GAMES FOR VIC

Skier Thrill to downhill skiing, using your joystick to hit flags and avoid obstacles. Great graphics. 3 levels of difficulty. $17.95

Maze of Mikor Adventure-like game with stunning graphics challenges you to steal the Warlock's gold as you evade the demon. $17.95

Tank Wars Match your wits against the evasive enemy, maneuver around obstacles and avoid mines. $17.95

Victrek Graphics and sound add to the excitement as you scan galactic maps, maneuver through star bases, and battle klingons. Enhanced version included for 8K VIC. $17.95

Pinball Score points with flippers through bumpers and alleys. This game is the real thing. $15.95

Simon Four squares light and sound at random. Then you imitate the sequence. It gets tougher as you get better. $15.95

Fuel Pirates Protect your stock of atomic fuel from raiding pirates using your particle cannon. $15.95

Lazer Blitz Terrific graphics as you destroy enemy aircraft from your flying saucer. $17.95

Pak Bomber is dropping bombs that you must catch. Great challenge for eye-to-hand coordination. $15.95

NEW FOR 5K VIC 20!

Tank Trap You're challenged to protect your citizens. 4 exciting levels, each tougher than the one before. $17.95.

Concentration Test your recall skills, when you try to remember what you saw beneath the block, and match it. $15.95.

Dam Bomber You must break the dam while under cannon fire. $15.95.

VIC FORTH Full FigFORTH implementation with compiler, interpreter and complete editor. Runs on standard VIC 20 with 5K. $59.95 on cartridge.

HESMON Machine language monitor. Contains 50% more commands than Commodore's. $39.95 on cartridge.

Turtle Graphics Based on LOGO. Perfect for learning computer programming. Great for kids. Very versatile. $39.95 on cartridge.

HES Writer Word processing. $39.95 on cartridge.

SPECIAL ANNOUNCEMENT TO OUR CUSTOMERS AND DEALERS

HES has relocated to the San Francisco Bay Area and is now a division of USI International. We now have greater resources to provide you with excellent software on cartridge, cassette, or diskette in superior quality packaging.

Watch for more exciting products from HES.

UTILITIES FOR PET & VIC

6502 ASSEMBLER PACKAGE

HESBAL is a 1- or 2-pass Assembler using standard MOS mnemonics and operand formats, has pseudo-opcodes and over 25 error messages. HESEDIT is a full screen text editor for use with HESBAL or alone. Assembler package runs on PET or VIC with 1 cassette and minimum 8K, (specify PET or VIC). $23.95 on cassette, $26.95 on diskette.

HESCOM transfers data and programs bidirectionally between PETs, VICs, or a PET and VIC at 3 times the speed of the disk. Set up VIC as a terminal to PET and create games for 2 players. Or use VIC as a peripheral to PET for hi-res graphics and sound. Only $49.95 on cassette, $52.95 on diskette.

HESCOUNT monitors BASIC program's execution and accumulates data. Essential for debugging and optimization. Discover how many times your program looped, and when IF statements were true or false. Fast execution. Runs on PET or VIC. On cassette $23.95. On diskette $26.95.

HESCAT Complete hi-speed diskette cataloging system. Five programs let you sort names, print reports 3 ways, and locate file names in memory or on disk, and much more. Works with any PET/CBM, 16K and dual drives. $39.95.

HESLISTER takes complex BASIC programs and prints (to screen or printer) in an easily understood manner. Lets you analyze BASIC programs to alter or debug code. Works on any PET/CBM and 1 disk drive. $23.95.

HESPLOT Very fast hi-res graphics subroutines for VIC. Includes line drawing routines. With 8K VIC plot within field of 176 x 160. On cassette $17.95

All products available at your dealer or directly from HES.
Add $2 postage. Calif. res. add 6% sales tax.
We accept VISA and MasterCard. Dealer inquiries invited.

PET, CBM, and VIC are trademarks of Commodore.

HES Human Engineered Software
71 Park Lane • Brisbane, CA 94005
(415) 468-4110

Send today for your **FREE CATALOG** of VIC and PET/CBM Software

Name_____

Street_____

City_____

State_____ Zip_____

Mail to **Human Engineered Software**
71 Park Lane • Brisbane, CA 94005

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

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

The Really Dumb Terminal Program

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

```
10 OPEN1,2,3,CHR$(38)+CHR$(160)
20 GET A$: IF A$="" THEN 60
30 IFA$=CHR$(147) THEN 90: REM CLEAR/HOME ~
    QUITS
40 A=ASC(A$) AND 127 : IF A=20 THEN PRINT#
    1, CHR$(8);: GOTO60
50 IF A>31 OR A=13 THEN PRINT#1, CHR$(A);
60 GET#1,A$ : IFA$="" THEN 20
70 A=ASC(A$) AND 127 : IF A=8 THEN PRINT C
    HR$(20); : GOTO20
80 IF A>31 OR A=13 THEN PRINT CHR$(A);
90 CLOSE1 : END
```

It's fun. It's sophisticated. But it is a little complex, and experience will be needed before you feel completely at home with VIC's communications features. ©

Keyprint, a fast machine language utility to copy the exact contents of a screen to a Commodore printer, was first published in **COMPUTE!**, *November/December, 1980, #7. Here are fourteen versions of this popular program, so you're sure to find the right one for your system. Included are complete descriptions of how to type in and use Keyprint.*

The Keyprint Compendium

Charles Brannon
Editorial Assistant

Keyprint's usefulness made it very popular. Within weeks, **COMPUTE!** began to receive many conversions, updates, and improvements to Keyprint. We've gathered them all together in this issue, with descriptive notes.

The BASIC programs can be entered and SAVEd as usual. RUN will PUT Keyprint into memory. The hex dumps are entered with the Machine Language Monitor.

To get to the monitor, type SYS 1024. You'll see something like:

```
     PC   IRQ   SR AC XR YR SP
.;  B780 E455 2C 34 3A 9D FA
```

Now list the block of memory occupied by the version of Keyprint you want to enter by typing .M 033A,03CA (or whatever is given in the listing). Then use the cursor to move over to the first two-digit (hex) number and begin typing, replacing the numbers on the screen. Press RETURN at the end of each line.

Now enter the correct ".save" line to SAVE Keyprint to tape or disk. For example,

```
.S "KEYPRINT",08,033A,03CB     (FOR TAPE)
.S "0:KEYPRINT",08,033A,03CB     (FOR DISK)
```

Be sure you SAVE Keyprint before you activate it. That way, if you made a mistake and the program "crashes," you can LOAD the program and proofread it. To activate Keyprint, enter the appropriate SYS, for example, SYS 826 [*033A in hex*]. You can then press the backslash " \ " (or whatever key is used) to dump the screen at any time to your printer.

The programs are keyed to the descriptions below:

1. Keyprint for Upgrade ROMs.

2. Eric Brandon's conversion of Keyprint to the Original ROM PET. Same operating instructions as Keyprint.

Program 1.
.M 033A 03CB
SYS 826

```
033A 78 A9 03 85 91 A9 45 85
0342 90 58 60 A5 97 C9 45 D0
034A 03 20 51 03 4C 2E E6 A9
0352 80 85 20 A9 00 85 1F A9
035A 04 85 B0 85 D4 20 BA F0
0362 20 2D F1 A9 19 85 22 A9
036A 0D 85 21 20 D2 FF A9 11
0372 AE 4C E8 E0 0C D0 02 A9
037A 91 20 D2 FF A0 00 B1 1F
0382 29 7F AA B1 1F 45 21 10
038A 0B B1 1F 85 21 29 80 49
0392 92 20 D2 FF 8A C9 20 B0
039A 04 09 40 D0 0E C9 40 90
03A2 0A C9 60 B0 04 09 80 D0
03AA 02 49 C0 20 D2 FF C8 C0
03B2 28 90 CB A5 1F 69 27 85
03BA 1F 90 02 E6 20 C6 22 D0
03C2 A6 A9 0D 20 D2 FF 4C CC
03CA FF 72 21 61 3F 7F 76 57
```

3. David Swaim has made a BASIC loader for Original ROM Keyprint for those who don't have a machine language monitor.

4. Mark W. Petersmeyer's conversion of Keyprint to the 4.0 ROM PET/CBM. Same operating instructions as Keyprint.

5. Jean Pierre Blanger has converted Keyprint to the 4.0 ROMs. SYS 826 to activate. Use the ‹SPACE› bar to print the screen, or SYS 843 under program control.

6. David Curtis has changed Keyprint so that it will

Program 2.
.M 033A 03CB
SYS 826

```
033A 78 A9 03 8D 1A 02 A9 47
0342 85 90 90 58 60 AD 03 02
034A C9 45 D0 03 20 54 03 4C
0352 85 E6 A9 80 85 20 A9 00
035A 85 1F A9 04 8D 64 02 85
0362 F1 20 BA F0 20 32 F1 A9
036A 19 85 22 A9 0D 85 21 20
0372 D2 FF A9 11 AE 4C E8 E0
037A 0C D0 02 A9 91 20 D2 FF
0382 A0 00 B1 1F 29 7F AA B1
038A 1F 45 21 10 0B B1 1F 85
0392 21 29 80 49 92 20 D2 FF
039A 8A C9 20 B0 04 09 40 D0
03A2 0E C9 40 90 0A C9 60 B0
03AA 04 09 80 D0 02 49 C0 20
03B2 D2 FF C8 C0 28 90 CB A5
03BA 1F 69 27 85 1F 90 02 E6
03C2 20 C6 22 D0 A6 A9 0D 20
03CA D2 FF 4C CC FF 67 54 00
```

work with his AXIOM EX801 printer and has added some new commands. Less than key < prints the screen. Left arrow stops printing. Greater than symbol > deactivates Keyprint. It may also be useful to owners of printers other than Commodore's.

7. Melvin Field's version of Keyprint is for the 4.0 ROMs. Use SYS 634 to activate Keyprint. To access the screendump, use the backslash or SYS 657. He offers some alternatives to the use of the backslash. The direct mode statement FOR I = 1 TO

Program 3.

```
1 REM KEYPRINT LOADER PROGRAM
2 REM BY DAVID SWAIM
3 REM 2631 CALLAWAY RD
4 REM MARIETTA, GA  30060
5 REM THE FOLLOWING DATA IS THE DECIMAL
     EQUIVALENT OF
6 REM THE HEX LISTING BY ERIC BRANDON
7 REM COMPUTE! MARCH 1981 PAGE 92
10 DATA 120,169,3,141,26,2,169,71,141,25
    ,2,88,96,173,3,2,201,69
20 DATA 208,3,32,84,3,76,133,230,169,128
    ,133,32,169,0,133,31,169,4,141
30 DATA 100,2,133,241,32,186,240,32,50,2
    41,169,25,133,34,169,13,133,33
40 DATA 32,210,255,169,17,174,76,232,224
    ,12,208,2,169,145,32,210,255
50 DATA 160,0,177,31,41,127,170,177,31,6
    9,33,16,11,177,31,133,33,41,128
60 DATA 73,146,32,210,255,138,201,32,176
    ,4,9,64,208,14,201,64,144,10
70 DATA 201,96,176,4,9,128,208,2,73,192,
    32,210,255,200,192,40,144,203
80 DATA 165,31,105,39,133,31,144,2,230,3
    2,198,34,208,166,169,13,32,210
90 DATA 255,76,204,255,103,84,0,0
95 REM
96 REM POKE THE ML PROGRAM BEGINNING AT~
    LOCATION 826 ( 033A )
99 REM
100 FOR I=826 TO 978
110 READ X
120 POKE I,X
130 NEXT I
140 END
```

Program 4.
.M 027A 030B
SYS 634

```
027A 78 A9 02 85 91 A9 85 85
0282 90 58 60 A5 97 C9 45 D0
028A 03 20 91 02 4C 55 E4 A9
0292 80 85 20 A9 00 85 1F A9
029A 04 85 B0 85 D4 20 D5 F0
02A2 20 48 F1 A9 19 85 22 A9
02AA 0D 85 21 20 D2 FF A0 11
02B2 AE 4C E8 E0 0C D0 02 A9
02BA 91 20 D2 FF A0 00 B1 1F
02C2 29 7F AA B1 1F 45 21 10
02CA 0B B1 1F 85 21 29 80 49
02D2 92 20 66 F2 8A C9 20 B0
02DA 04 09 40 D0 0E C9 40 90
02E2 0A C9 60 B0 04 09 80 D0
02EA 02 49 C0 20 D2 FF C8 C0
02F2 28 90 CB A5 1F 69 27 85
02FA 1F 90 02 E6 20 C6 22 D0
0302 A6 A9 0D 20 D2 FF 4C CC
030A FF 00 00 00 00 00 00 00
```

Program 5.
.M 033A 03CB
SYS 826

```
033A 78 A9 03 85 91 A9 45 85
0342 90 58 60 A5 97 C9 06 D0
034A 03 20 51 03 4C 55 E4 A9
0352 80 85 20 A9 00 85 1F A9
035A 04 85 B0 85 D4 20 D5 F0
0362 20 43 F1 A9 19 85 22 A9
036A 0D 85 21 20 D2 FF A9 11
0372 AE 4C E8 E0 0C D0 02 A9
037A 91 20 D2 FF A0 00 B1 1F
0382 29 7F AA B1 1F 45 21 10
038A 0B B1 1F 85 21 29 80 49
0392 92 20 D2 FF 8A C9 20 B0
039A 04 09 40 D0 0E C9 40 90
03A2 0A C9 60 B0 04 09 80 D0
03AA 02 49 C0 20 D2 FF C8 C0
03B2 28 90 CB A5 1F 69 27 85
03BA 1F 90 02 E6 20 C6 22 D0
03C2 A6 A9 0D 20 D2 FF 4C CC
03CA FF 72 21 61 3F 00 00 00
```

Program 6.
.M 033A 03ED
SYS 826

```
033A 78 A9 03 85 91 A9 45 85
0342 90 58 60 A5 97 C9 05 F0
034A 11 C9 0C F0 03 4C 2E E6
0352 A9 E6 85 91 A9 2E 85 90
035A D0 F3 20 62 03 4C 2E E6
0362 A9 80 85 20 A9 00 85 1F
036A A9 04 85 B0 85 D4 20 BA
0372 F0 20 2D F1 A9 19 85 22
037A A9 0A 85 21 20 D2 FF A9
0382 00 AE 4C E8 E0 0C D0 02
038A A9 00 20 D2 FF A0 00 B1
0392 1F 29 7F C9 1F F0 4C AA
039A B1 1F 45 21 10 0B B1 1F
03A2 85 21 29 80 49 92 20 D2
03AA FF 8A C9 20 B0 04 09 40
03B2 D0 1B C9 40 90 17 C9 60
03BA B0 11 AD 4C E8 C9 0C F0
03C2 06 8A 09 20 4C CF 03 09
03CA 80 D0 02 49 C0 20 D2 FF
03D2 C8 C0 28 90 BA A5 1F 69
03DA 27 85 1F 90 02 E6 20 C6
03E2 22 D0 95 A9 0D 20 D2 FF
03EA 4C CC FF EA EA EA EA EA
```

1000:PRINT PEEK(151):NEXT will print the coordinate code of the key you want to use. If you POKE 648,x then the key that corresponds to the coordinate code x will dump the screen. He suggests using the REV/OFF key, since it won't print any-

Program 7.

```
.M 027A 030B
SYS 634

027A 78 A9 02 85 91 A9 85 85
0282 90 58 60 A5 97 C9 45 D0
028A 03 20 91 02 4C 55 E4 A9
0292 80 85 20 A9 00 85 1F A9
029A 04 85 B0 85 D4 20 D5 F0
02A2 20 66 F2 A9 19 85 22 A9
02AA 0A 85 21 20 43 F1 A9 11
02B2 AE 4C E8 E0 0C D0 02 A9
02BA 91 20 66 F2 A0 00 B1 1F
02C2 29 7F AA B1 1F 45 21 10
02CA 0B B1 1F 85 21 29 80 49
02D2 92 20 66 F2 8A C9 20 B0
02DA 04 09 40 D0 0E C9 40 90
02E2 0A C9 60 B0 04 09 80 D0
02EA 02 49 C0 20 66 F2 C8 C0
02F2 29 90 CB A5 1F 69 27 85
02FA 1F 90 02 E6 20 C6 22 D0
0302 A6 A9 0D 20 66 F2 4C CC
030A FF 20 20 20 20 20 20 20
```

Program 8.

```
.M 033A 03FF
SYS 826

033A 78 AC 1A 02 20 8D 1F C0
0342 E6 D0 03 20 98 1F 58 60
034A AD 03 02 C9 45 F0 07 C9
0352 0F D0 06 4C CF 03 20 5E
035A 03 4C 85 E6 A9 80 85 72
0362 A9 00 85 71 20 78 1F A9
036A 19 85 74 9A 0D 85 73 20
0372 30 F2 A9 11 AE 4C E8 E0
037A 0C D0 02 A9 91 20 30 F2
0382 A0 00 B1 71 29 7F AA B1
038A 71 45 73 10 0B B1 71 85
0392 73 29 80 49 92 20 30 F2
039A 8A C9 20 B0 04 09 40 D0
03A2 0E C9 40 90 0A C9 60 B0
03AA 04 09 80 D0 02 49 C0 20
03B2 30 F2 C8 C0 28 90 CB A5
03BA 71 69 27 85 71 90 02 E6
03C2 72 C6 74 D0 A6 A9 0D 20
03CA 30 F2 4C 7D F2 A2 08 20
03D2 DF 03 4C 85 E6 A2 10 20
03DA 78 1F 4C B0 C5 A0 08 8C
03E2 0D 02 BD EE 03 99 0E 02
03EA CA 88 D0 F6 00 93 53 59
03F2 53 39 38 33 0D 93 53 59
03FA 53 39 37 32 0D 00 00 00
```

8. J. Michael McCormick has enhanced Eric Brandon's version of Keyprint for the Original ROMs. The at-symbol "@" will automatically list a BASIC program onto the printer.

9. This is Jerry Levitt's conversion of Keyprint for

Program 9.

```
.M 033A 03E0
SYS 826

033A 78 A9 03 85 91 A9 45 85
0342 90 58 60 A5 97 C9 45 D0
034A 03 20 51 03 4C 55 E4 A9
0352 00 85 D9 A9 01 85 D2 A2
035A 04 86 D4 A4 FF 84 D3 20
0362 63 F5 A2 01 20 FE F7 A9
036A 80 85 20 A9 00 85 1F A9
0372 19 85 22 A9 0D 85 21 20
037A D2 FF A9 11 AE 4C E8 E0
0382 0C D0 02 A9 91 20 D2 FF
038A A0 00 B1 1F 29 7F AA B1
0392 1F 45 21 10 0B B1 1F 85
039A 21 29 80 49 92 20 D2 FF
03A2 8A C9 20 B0 04 09 40 D0
03AA 0E C9 40 90 0A C9 60 B0
03B2 04 09 80 D0 02 49 C0 20
03BA D2 FF C8 C0 28 90 CB A5
03C2 1F 69 27 85 1F 90 02 E6
03CA 20 C6 22 D0 A6 A9 0D 20
03D2 D2 FF 20 CC FF A9 01 4C
03DA E2 F2 20 20 D2 FF 88 D0
```

Program 10a.

```
10 DATA120,169,3,133,145,169,69,133
20 DATA144,88,96,165,151,201,69,208
30 DATA3,32,81,3,76,46,230,169
40 DATA128,133,32,169,0,133,31,169
50 DATA4,133,176,133,212,32,186,240
60 DATA32,45,241,169,25,133,34,169
70 DATA13,133,33,32,210,255,169,17
80 DATA174,76,232,224,12,208,2,169
90 DATA145,32,210,255,160,0,177,31
100 DATA41,127,170,177,31,69,33,16
110 DATA11,177,31,133,33,41,128,73
120 DATA146,32,210,255,138,201,32,176~
    ~
130 DATA4,9,64,208,14,201,64,144
140 DATA10,201,96,176,4,9,128,208
150 DATA2,73,192,32,210,255,200,192
160 DATA40,144,203,165,31,105,39,133
170 DATA31,144,2,230,32,198,34,208
180 DATA166,169,13,32,210,255,76,204
190 DATA255,114,33,97,63,127,118,87
200 FOR I=826 TO 997
210 READV:POKEI,V
220 NEXTI
225 PRINT"{CLEAR}{10 DOWN}"
230 PRINT"TYPE SYS 826 TO ACTIVATE"
240 PRINT"{REV}{03 DOWN}THEN \ PRINTS~
    ~S FROM SCREEN EXACTLY!!!!"
250 END
```

the 4.0 ROMs. He gives some comments on customizing it. Keyprint uses logical file #1. To change this, POKE the new logical file number into memory locations 854, 869, and 984. POKE any different device number into 858. To change the key that dumps the screen, use the same procedure as in version **6.**

Program 10b.

```
10 DATA120,169,2,133,145,169,133,133
20 DATA144,88,96,165,151,201,69,208
30 DATA3,32,145,2,76,46,230,169
40 DATA128,133,32,169,0,133,31,169
50 DATA4,133,176,133,212,32,186,240
60 DATA32,45,241,169,25,133,34,169
70 DATA13,133,33,32,210,255,169,17
80 DATA174,76,232,224,12,208,2,169
90 DATA145,32,210,255,160,0,177,31
100 DATA41,127,170,177,31,69,33,16
110 DATA11,177,31,133,33,41,128,73
120 DATA146,32,210,255,138,201,32,176~
   ~
130 DATA4,9,64,208,14,201,64,144
140 DATA10,201,96,176,4,9,128,208
150 DATA2,73,192,32,210,255,200,192
160 DATA40,144,203,165,31,105,39,133
170 DATA31,144,2,230,32,198,34,208
180 DATA166,169,13,32,210,255,76,204
190 DATA255,114,33,97,63,127,118,87
200 FOR I=634 TO 785
210 READV:POKEI,V
220 NEXTI
225 PRINT"{CLEAR}{10 DOWN}"
230 PRINT"TYPE SYS 634 TO ACTIVATE"
240 PRINT"{REV}{03 DOWN}THEN \ PRINTS~
   ~S FROM SCREEN EXACTLY!!!!"
250 END
```

Program 11a. (4.0C)
.M 033A 03CB
SYS 826

```
033a 78 a9 03 85 91 a9 45 85
0342 90 58 60 a5 97 c9 45 d0
034a 03 20 51 03 4c 55 e4 a9
0352 80 85 20 a9 00 85 1f a9
035a 04 85 b0 85 d4 20 d5 f0
0362 20 48 f1 a9 19 85 22 a9
036a 0d 85 21 20 d2 ff a9 11
0372 ae 4c e8 e0 0c d0 02 a9
037a 91 20 d2 ff a0 00 b1 1f
0382 29 7f aa b1 1f 45 21 10
038a 0b b1 1f 85 21 29 80 49
0392 92 20 d2 ff 8a c9 20 b0
039a 04 09 40 d0 0e c9 40 90
03a2 0a c9 60 b0 04 09 80 d0
03aa 02 49 c0 20 d2 ff c8 c0
03b2 28 90 cb a5 1f 69 27 85
03ba 1f 90 02 e6 20 c6 22 d0
03c2 a6 a9 0d 20 d2 ff 4c cc
03ca ff 00 00 00 00 00 00 00
```

10. Timothy Dailey has given us two BASIC loader programs for Keyprint. He has also moved Keyprint to the first cassette buffer. You use SYS 634 to activate the latter version, and SYS 657 to dump the screen independently of Keyprint.

11. Joseph Holmes has supplied 4.0 versions of Keyprint for tape systems (4.0C), disk (4.0D), or for use on an 80-column CBM (80D).

Program 11b. (4.0D)
.M 027A, 030B
SYS 634

```
027a 78 a9 02 85 91 a9 85 85
0282 90 58 60 a5 97 c9 45 d0
028a 03 20 91 02 4c 55 e4 a9
0292 80 85 20 a9 00 85 1f a9
029a 04 85 b0 85 d4 20 d5 f0
02a2 20 48 f1 a9 19 85 22 a9
02aa 0d 85 21 20 d2 ff a9 11
02b2 ae 4c e8 e0 0c d0 02 a9
02ba 91 20 d2 ff a0 00 b1 1f
02c2 29 7f aa b1 1f 45 21 10
02ca 0b b1 1f 85 21 29 80 49
02d2 92 20 d2 ff 8a c9 20 b0
02da 04 09 40 d0 0e c9 40 90
02e2 0a c9 60 b0 04 09 80 d0
02ea 02 49 c0 20 d2 ff c8 c0
02f2 28 90 cb a5 1f 69 27 85
02fa 1f 90 02 e6 20 c6 22 d0
0302 a6 a9 0d 20 d2 ff 4c cc
030a ff 00 00 00 00 00 00 00
```

Program 11c. (80D)
.M 027A, 030B
SYS 634

```
027a 78 a9 02 85 91 a9 85 85
0282 90 58 60 a5 97 c9 dc d0
028a 03 20 91 02 4c 55 e4 a9
0292 80 85 20 a9 00 85 1f a9
029a 04 85 b0 85 d4 20 d5 f0
02a2 20 48 f1 a9 19 85 22 a9
02aa 0d 85 21 20 d2 ff a9 11
02b2 ae 4c e8 e0 0c d0 02 a9
02ba 91 20 d2 ff a0 00 b1 1f
02c2 29 7f aa b1 1f 45 21 10
02ca 0b b1 1f 85 21 29 80 49
02d2 92 20 d2 ff 8a c9 20 b0
02da 04 09 40 d0 0e c9 40 90
02e2 0a c9 60 b0 04 09 80 d0
02ea 02 49 c0 20 d2 ff c8 c0
02f2 50 90 cb a5 1f 69 4f 85
02fa 1f 90 02 e6 20 c6 22 d0
0302 a6 a9 0d 20 d2 ff 4c cc
030a ff 00 00 00 00 00 00 00
```

For PET/CBM BASICs 4.0 or Upgrade (3.0), 40 or 80 column screens, and disk drive. This short routine shows an easy way to transfer screen images to disk and back to the screen, in BASIC.

Screen Saver

David Wine
Philadelphia

This two-part program Screen Saver will SAVE the screen on your PET or CBM to disk, and then LOAD it back. The screen on your computer is mapped onto a continuous chunk of memory. Lines 230-240 trick BASIC into thinking the current program in memory starts at the top left corner of the screen and ends at the bottom right corner. There are two pointers in zero page that BASIC uses to tell it where the program is – the start-of-BASIC text pointer, and the start-of-variables (end of BASIC text) pointer. Screen Saver stores these and then points them to the start and end of the screen. On line 250, a simple SAVE stores the screen on disk. After this, the BASIC pointers used are restored their previous values by PEEKing them from the locations in the second cassette buffer where they were stored.

If you were to try to use variables to store the original pointers, you would run into trouble when it came time to put them back. BASIC gets very confused about variables when these pointers are redirected. Just for fun, try putting a STOP at line 245 and looking at a few variables.

The second part of the program LOADs the "screen" file back onto the screen. BASIC knows where to put it, because the first two bytes of a program file are written with the load address of the program in the computer's memory. Usually, this is 1024, but for the screen it's 32768. The end of a program on disk is followed by three zero bytes, so BASIC knows when to stop LOADing.

If line 310 were a normal LOAD, the program would stop execution right after LOADing the screen. In addition, out-of-memory errors would haunt you until you typed NEW. Luckily, there is a convenient BASIC firmware routine which LOADs the current file without disturbing BASIC's delicate pointers.

I can think of a couple of uses for screen saver. One is as part of an on-line help system. When the user asks for help, the current screen is saved and the help messages displayed. When he or she is done viewing the help screen, the old screen can be restored. Another use is for easy documentation of screen formats. Maybe even frame-by-frame animation? I would be interested in hearing of any other ideas.

```
100 REM TO USE THIS ON 40-COLUMN SCREENS, C
    HANGE LINE 240 TO:
101 REM POKE 42,232  AND POKE 43,131
102 REM
103 REM TO ADAPT TO UPGRADE (3.0) BASIC, CH
    ANGE LINE 310'S SYS TO 62242
104 REM
105 REM
200 REM SAVE IT
210 POKE900,PEEK(40):POKE901,PEEK(41) : REM
     SAVE START BASIC TEXT
220 POKE902,PEEK(42):POKE903,PEEK(43) : REM
     SAVE START VARIABLES
230 POKE40,0:POKE41,128                : REM
     POINT TO START SCREEN
240 POKE42,208:POKE43,135              : REM
     POINT TO END SCREEN
250 SAVE"@0:SCREEN",8
260 POKE40,PEEK(900):POKE41,PEEK(901) : REM
     RESTORE POINTERS
270 POKE42,PEEK(902):POKE43,PEEK(903)
280 PRINTCHR$(147)
290 REM
300 REM LOAD IT
310 OPEN1,8,1,"0:SCREEN":SYS62294:CLOSE1
320 GETC$:IFC$=""THEN320
```

How to get 256 colors out of your Atari. Last month, this three-part series opened with a discussion of Atari Graphics. Part II examines techniques involving color indirection *and looks at the new GTIA chip in detail. If you have one of the older machines, your dealer should now have the new chip and can install it for you for about $60 (according to Atari). If your machine is still under warranty, the upgrade is free.*

Next month, this series concludes with several programs which put GTIA through its paces.

Part II:

Atari Video Graphics And The New GTIA

Craig Chamberlain
Birmingham, MI

Using Color Indirection

With color indirection, the number of different playfields is limited according to the number of bits per pixel, but the actual color/luminance of each playfield can be one of the 128 possibilities. The data bits are used as an index or offset into playfield color registers:

```
COLOR0  $02C4  708
    playfield zero color register
COLOR1  $02C5  709
    playfield one
COLOR2  $02C6  710
    playfield two (used in modes 0 and 8)
COLOR3  $02C7  711
    playfield three (used in color text modes)
COLOR4  $02C8  712
    background color register
```

These playfield color registers use seven bits to select the color and luminance, as follows:

```
D7,D6,D5,D4    color
D3,D2,D1       luminance
D0             not used
```

BITS	VALUE	COLOR
0000	0	gray (no color)
0001	1	light orange
0010	2	orange
0011	3	red orange
0100	4	pink
0101	5	purple
0110	6	purple blue
0111	7	blue
1000	8	blue
1001	9	light blue
1010	10	turquoise
1011	11	blue green
1100	12	green
1101	13	yellow green
1110	14	orange green
1111	15	light orange

Atari BASIC allows you to select a playfield color to draw in by using the COLOR statement. The color register that corresponds to that playfield can be changed by using SETCOLOR.

Color indirection is a tool that should not be overlooked. It is possible to draw a detailed figure on the screen with one playfield, and then change the color of the entire figure with just one command. For example, a printed message can flash in colors to attract attention. A "glowing" effect can be created by rapidly changing the luminance of a playfield while maintaining the same color. Or, the playfield colors can all be set to the same color/luminance as the background. Figures drawn will not appear until the playfield color registers are changed. By changing the registers one at a time, an animation effect can be created. Color indirection may still not solve the problem of having many colors on the screen at the same time, but it does afford possibilities that otherwise would be difficult to achieve.

In special instances, playfield color registers can be changed during the horizontal blank, in which case all 128 color variations can be shown in one frame. This requires the use of machine language and still does not solve the problem of many colors on one scan line. Fortunately, experience has shown that, for many applications, three playfield colors will be sufficient.

Multiple Colors

Nevertheless, there are times when many colors would be desirable. This is where the GTIA steps in. It should now be apparent that 16 colors will require four bits per pixel. This is very expensive in terms of memory, so either pixel size or display memory will have to increase. Because ANTIC has a limit on how much memory it can access during one horizontal scan line, we have a limit on how much memory can be devoted to a screen. Therefore, resolution will have to suffer.

Before we see what the memory limit is, we should mention the two modes which are exceptions to the above rules. Three things distinguish modes zero and eight from the normal modes. Each pixel is a half color clock wide; a side effect of this is artifacting. The background color now becomes the border, and the main part of the screen is filled with playfield two. Finally, since the whole screen is now playfield two, the bit no longer tells which playfield to use, but which *luminance* to use.

WELCOME TO ZORK!

Until you've entered the world of ZORK, you've never truly adventured underground. Both ZORK I and ZORK II are designed so that you'll experience their challenges in the most realistic sense. You can communicate in complete sentences rather than two-word commands, with the largest vocabulary and widest range of command options in the genre. Because ZORK's mysteries are the most intricate you'll ever encounter, it will take all your intellectual abilities to survive and emerge victorious from the underground. And because the challenges change with every move you make, each time you re-enter ZORK you'll face new intrigues.

INFOCOM
ZORK I

The Great Underground Empire confronts you with perils and predicaments ranging from the mysterious to the macabre, as you strive to discover the Twenty Treasures of ZORK and escape with them and your life!

Cat No. 4067 Atari,32K,disk

$39.95

INFOCOM
ZORK II

The Wizard of Frobozz takes you into new depths of the subterranean realm. There you'll meet the Wizard, who will attempt to confound your quest with his capricious powers.

Cat No. 4068 Atari,32K,disk

$39.95

ROKLAN
WIZARDS OF WOR

Descend into the Dungeons with your squadrons of Warriors to battle a host of monsters both visible and invisible—even the Wizard himself. Survive the changing patterns of mazes, fight in the arena, the Worlord Dungeon, and the Pit. Features simultaneous one or two player action.

Cat No. 4090 Atari,16K,disk

$39.95

ROKLAN
GORF

A unique sight and sound adventure that enlists players into the Interstellar Space Force for challenging voyages against the Gorfian Empire. Your mission is to repel attacks by Droids, Antigravity Bombs, Antiparticle Lasers, Kamikaze Ships, Gorfs, Fighters, Subquark Torpedoes, and destroy the Gorfian forces.

Cat No. 4089 Atari,16K,disk

$39.95

COMPUTE! BOOKS
Inside Atari DOS

Bill Wilkinson and his staff at Optimized Systems Software were the developers of Atari DOS 2.OS. In this book, they carefully describe the structure and workings of the DOS, with short chapters devoted to each module. In addition to these descriptive chapters, the book presents the complete and commented source code listings for Atari DOS 2.OS. An appendix explains the book for the non-advanced Atari owner. Cat No. 4030 **$19.95**

DISKETTE SALE

Take your choice between two top-quality brands of 5¼" soft-sector, single-side diskettes.

VERBATIM DATALIFE (525-01-18158)
Cat No. 1147
$30.00

DYSAN (104/1)
Cat No. 3966
$38.95

DISK PROTECTOR CASES

- Holds 50 diskettes
- Attractive yet durable
- Perfect for home or office

Cat No. 2956 (5¼")

$27.50

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 AUG. 31, 1982 Mention this ad and we pay shipping (UPS ground only).

HW ELECTRONICS

19511 BUSINESS CENTER DR.
DEPT. G8
NORTHRIDGE, CA 91324

WHEN IN SOUTHERN CALIFORNIA, VISIT OUR RETAIL STORES

19511 Business Center Dr.
Northridge, CA 91324

2301 Artesia Blvd.
Redondo Beach, CA 90277

444 S. Indian Ave.
Palm Springs, CA 92262

MODE	BIT	LUMINANCE REGISTER
0,8	1	playfield one
0,8	2	playfield two (no image)

The color part of playfield one is ignored; only the luminance data is used. If the luminance values of playfields one and two are the same, the writing disappears. Modes zero and eight use this special "half color clock, one playfield color, two brightness" arrangement. Both modes have 320 distinct points of light horizontally and have single scan line resolution. The only difference between mode zero and mode eight is that the first is a text mode and the second is a direct mapping mode. Mode zero uses a character set and thereby saves memory; about 1K is required for this mode. Mode eight doesn't use a character set, and requires approximately 8K. That is our display memory limit. The Atari 400/800 is not capable of doing DMA to much more memory than the memory represented by one television frame.

Since the "half color clock, one color, two brightness" mode is used by graphics modes zero and eight, all the GTIA really does is provide three variations on this mode. They all use the maximum memory arrangement used by mode eight, so each of the three new modes requires 8K. All of the new modes use four bit pixels, so the horizontal resolution goes from 320 (half color clock) to 80 (two color clock, as in modes four and five). Therefore, the resolution for all three new modes is 80 by 192, for a total of 15360 points. One side effect of changing only the horizontal resolution is that the pixels are no longer square.

The ANTIC instruction register mode number for the maximum memory mode (the number you will find in the display list) is $0F, or decimal 15. It is important to understand that this number indicates not only mode eight, but also nine, ten, and eleven as well. In fact, the display list for any one of these modes is identical to the display list for any of the others.

Selecting Modes With PRIOR

How then does ANTIC know which of the four is the desired mode? The answer is that ANTIC neither knows nor cares; no matter which mode is being used, ANTIC still has to do the same work of fetching memory. It's the GTIA that processes the video signal; somehow the chip must be told which of the four modes is wanted. The GTIA hardware register PRIOR does exactly that.

GPRIOR	$026F	623	shadow
PRIOR	$D01B	53275	hardware

The two most significant bits (bits six and seven) of this register are the GTIA special mode select bits. Here's how they are set.

MODE	BITS	HEX	DECIMAL
8	00	00	0
9	01	40	64
10	10	80	128
11	11	C0	192

For example, it is possible to switch from any one of the four modes to another simply by changing the values of the two select bits.

Other bits in GPRIOR serve different functions, so care must be taken not to alter them. These other bits allow multi-color players (blending on overlap), set all missiles to the color of playfield three to form a fifth player, and establish player/missile and playfield priorities. See the *Hardware Manual* for further information.

Now that we know how the three new modes are similar, let's find out how they are different.

Mode 11 is the one luminance, 16 color mode. The overall luminance is set by the background color, which, for this mode, defaults to a luminance of six, rather than the usual zero. It is now easy to draw rather finely detailed shapes in several colors without having to fool around with the display list and machine code interrupt routines. The thing I am especially excited about is going to make Apple owners envious. The Apple has a 16 color mode with resolution of 40 by 48, called the "lo res" mode. The Atari now has a 16 color mode, but the resolution is eight times greater than the Apple's.

Sixteen colors do present a problem, though, since the GTIA has only four playfield color registers. Therefore, mode 11 does not allow color indirection. The color on the screen is determined directly by the bit data stored in memory, according to the chart given earlier in the section on color indirection. The values in the four color/luminance registers are ignored. Some may consider this a disadvantage, but there is a benefit too. Just as the playfield color registers are not used, neither are the player/missile color registers used, so by using players it is possible to have 21 colors on the screen at the same time, without using display list interrupts or other tricks.

Producing 256 Colors

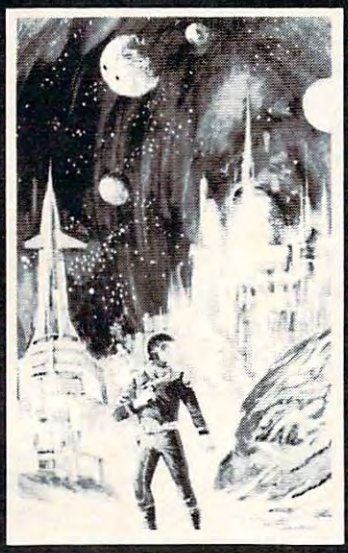
Mode nine is the one color, 16 luminance mode. This mode will be used to create some excellent three dimensional effects and digitized pictures. The 16 luminances, when stacked vertically by the scan line with each line having the next brightest luminance, blend so well that it is very difficult to see the division from one to the other. The main color is set by the background color. Weird things happen when you change the luminance of the background. Another nice fact is that having 16 main colors with 16 luminance variations means that the Atari is capable of producing 256 colors.

One advanced application for mode nine is

NEW FROM

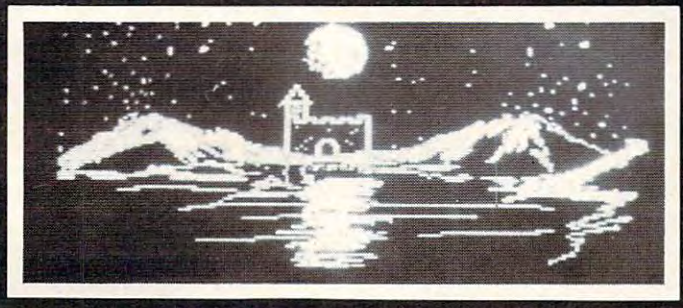
Synergistic Software

ACTION AND ADVENTURE



PROBE ONE
THE TRANSMITTER

A new high-res adventure game filled with arcade action, color, and sound effects. You must battle guard droids while searching room to room in a remote research colony. The original game design has various skill levels and new game layout each time you play. Probe One uses paddle or joystick as well as keyboard commands..... $34.95



WARLOCK'S REVENGE

Two full disks of challenging high-res adventure. You lead a select group of adventurers on a quest to gather treasure and destroy the evil warlock, Oldorf. As you explore caverns and castles you use your group's various skills to gather riches and fight off creatures. Warlock's Revenge uses keyboard commands and you may save the game for later play..... $34.95

Both games require 40K, 800/400, disk, and ATARI BASIC.

TOLL FREE ORDER LINE — 800-426-6505

We accept checks, VISA, MASTER CHARGE, and C.O.D.
$2.00 postage and handling charge, except on prepaid orders.

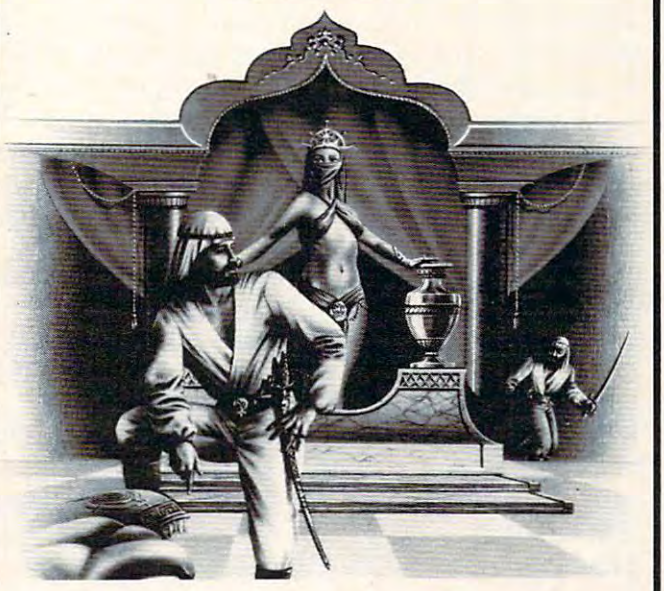
Synergistic Software
830 N. Riverside Dr., Suite 201, Renton, WA 98055
(206) 226-3216

ATARI 800/400 are trademarks of ATARI, INC.

NOW AVAILABLE FOR THE APPLE II

Ali Baba
and the forty thieves

By Stuart Smith



A fantasy role-playing adventure for Apple II and Atari Personal Computers.

Encounter sultans, thieves, fierce and friendly creatures as you guide your alter ego, Ali Baba, through the thieve's mountain den in an attempt to rescue the beautiful princess. Treasure, magic, and great danger await you! One or more human players can guide up to seventeen friendly characters through the many rooms, halls, and caves. Some characters wander around randomly, making each adventure a little different.

ALI BABA AND THE FORTY THIEVES is written in high resolution color graphics and includes music and sound effects. Adventures can be saved to disk and resumed at a later time.

Available for Apple II and Apple II Plus 48K or Atari 800 32K.

On diskette only — $32.95

FOR OUR COMPLETE LINE OF APPLE AND ATARI SOFTWARE PLEASE WRITE FOR OUR CATALOG

ASK FOR QUALITY SOFTWARE products at your favorite computer store. If necessary you may order directly from us. MasterCard and Visa cardholders may place orders by calling us at (213) 344-6599. Or mail your check or bankcard number to the address above. California residents add 6% sales tax. *Shipping Charges:* Within North America orders must include $1.50 for shipping and handling. Outside North America the charge for airmail shipping and handling is $5.00. Pay in U.S. currency.

QS QUALITY SOFTWARE

6660 Reseda Blvd., Suite 105, Reseda, CA 91335 (213) 344-6599

the display of digitized pictures. Digitization is a process by which a normal television picture, such as from a station or video recorder, can be analyzed and divided into different luminances. That information can be sent to the computer and stored on disk for later display. Mode nine, with 16 luminances and rather high resolution, is able to reproduce such pictures with impressive quality. Thus far we have seen only four digitized pictures. They were apparently made by some people at Atari, and two of the pictures were, uh, for mature viewers only. Standing from a short distance, however, it is very difficult to tell if any of these pictures is computer generated or not. I have never seen such quality on any other computer in the 400/800 price range without expensive additional equipment.

Mode ten is a cross between the other two modes; it allows eight colors plus the background, each with its own luminance as in the primary modes. Unlike the other two modes, this one allows color indirection, so it uses the playfield and player/missile registers for color/luminance information. This chart shows how data values correspond with playfield registers.

BITS	VALUE	REGISTER	PLAYFIELD
0000	0	704	PCOLR0
0001	1	705	PCOLR1
0010	2	706	PCOLR2
0011	3	707	PCOLR3
0100	4	708	COLOR0
0101	5	709	COLOR1
0110	6	710	COLOR2
0111	7	711	COLOR3
1000	8	712	COLOR4
1001	9	712	COLOR4
1010	10	712	COLOR4
1011	11	712	COLOR4
1100	12	708	COLOR0
1101	13	709	COLOR1
1110	14	710	COLOR2
1111	15	711	COLOR3

Only nine of the 16 possible data values correspond to different playfields. Data values greater than eight just repeat playfields. For some reason, the background color is no longer set by COLOR4, but instead by PCOLR0. The Atari BASIC statement SETCOLOR can't be used to change the player/missile color registers, so the equivalent POKE must be used. For any register, the data part of the POKE is the color choice number multiplied by 16, plus the luminance (refer to earlier chart).

The power of indirection is magnified when eight main drawing colors can be used. This mode is very useful for creating motion effects. With nine color/luminances and color indirection, mode ten may prove to be the most versatile of the three new modes.

Compatibility Between CTIA And GTIA

Remember that the GTIA only controls how the display is generated, so all programs written for the CTIA should run on a GTIA machine in the same way. There can be no such thing as incompatibility. We have, however, come across one discrepancy between the CTIA and GTIA. The video signal generated by the GTIA is shifted one half color clock, so colors produced by artifacting, such as in POOL 1.5 or Jawbreakers, will be different. That is just a minor visual difference; the important thing is that all software should be entirely compatible. Of course, you cannot expect a CTIA to generate these three new modes, but again the conflict is the display, not the program. In fact, I don't think it is even possible for the computer to tell whether it has a CTIA or GTIA in it.

Because of the half color clock shift, it is now possible for players and playfields to overlap perfectly, whereas with the CTIA they didn't.

There are some cases where software will not run on GTIA machines. This is due to the fact that some of the new computers with the GTIA also have a revised (no bugs) operating system in them. Atari has made very clear which memory locations and vectors are permanent and protected from any revisions. If a program does not run on a GTIA machine, it is the software's fault because illegal entry points were used.

One other conflict has appeared which really surprised me. We have discovered that a few programs written on CTIA machines carelessly set the GTIA special mode select bits of GPRIOR for no purpose. Since these two bits do nothing on the CTIA, there was no problem. But there was also no reason to involve them. When the same programs are run on GTIA, the accidental bit settings affect the display, even though modes nine, ten and eleven are not used. The function of those two bits has not been a secret. I figured out their function in July 1981, when I read the OS source listing before I bought my Atari 800. The *Hardware Manual* has described the three "new" modes in appendix H ever since the manual was released.

No Text Window

There is a difference between the normal modes and the three new modes – the three new ones do not allow split screen (text window at bottom) configurations. If you remember how modes eight and zero are related, you should understand why. The mode used in the text windows is mode zero, which follows the special "half color clock, one color, two luminances" arrangement. As stated above, having the mode select bits in GPRIOR set for a mode greater than eight causes mode zero to

act funny. A split screen would only be possible if a display list interrupt were inserted just before the text window area. The interrupt routine would have to reset to zero the mode select bits in the hardware register PRIOR, not the shadow register. The hardware register will then be reset to the value of GPRIOR during the vertical blank service routine.

The three new modes seem to handle player/missile to playfield collisions a little differently. In modes zero and eight, a playfield two collision is flagged when a player or missile hits a pixel whose luminance is controlled by COLOR1 rather than the COLOR2 for the main playfield. From what I have been able to tell thus far, there is no kind of playfield collision at all in modes nine and eleven. Mode ten collisions work only for playfield colors that correspond to the usual playfield registers (COLOR1 through COLOR3). Also, the fact that the background in this mode is set by PCOLR0 affects the priority of players and playfields in some cases. In priority, mode ten playfield colors PCOLR0 through PCOLR3 behave like players.

The GTIA still allows only eight luminances on the normal modes.

All new Atari computers are being shipped with the GTIA at no extra cost. The CTIA is no longer being produced. The new machines with the GTIA have little yellow or white stickers that have the letter "G" on them. Those of us who have older machines with the CTIA can replace it with a GTIA. The part number is C014805.

If you want to do it yourself, it will be a simple matter to replace the CTIA. The CTIA is on the CPU card that plugs into the motherboard inside the Atari case. It's not soldered in, so the replacement operation should take only 30 minutes if you have taken your computer apart before. Instructions are supplied with the chip. In the meantime, if you don't have the GTIA, don't fret. It will be a while before much software requiring the chip is available.

Do You Already Have The GTIA?

If you want to quickly see if your computer has a GTIA, try this: POKE 623,64 (while in the default mode, zero).

If you have the GTIA, the screen will go black. Otherwise, there will be no change and you'll know you've got the CTIA. If you have the GTIA and want to see 16 colors, try this.

```
10 GRAPHICS 11
20 FOR K=0 TO 79
30 COLOR K
40 PLOT K,0
50 DRAWTO K,191
60 NEXT K
70 GOTO 70
```

Telecommunications:

Choosing A Modem: Part II

Michael E. Day
Chief Engineer
Edge Technology
West Linn, OR

When considering the purchase of a modem, one important issue is compatibility – what other modems can you communicate with?

There are three major types of modems currently available to the general user. The 202 type, the 103, and a modem that has been enjoying a recent increase in popularity despite its high cost, the 212A.

The 202 modem has been around for some time. It has fallen out of favor recently due to the complexity of its operation. It requires some amount of computer control. Its main advantage is its speed of operation – up to 1200 bits per second (bps), which is equal to 120 characters per second. Because it has never received wide acceptance in the general user market, it also tends to cost far more than it should.

A Cassette Interface Is Practically A Modem

Of all the types of modems, the 202 has the potential for being the least expensive due to its minimal hardware required to convert the computer data to be sent over the phone lines. The greater degree of computer control over a 202 modem's operation also lowers its cost. It is actually possible to build a 202 for less than $25 in parts. This would translate to around $50 in high volume production, and would be even less if it were incorporated into a computer as a basic part of its design. In fact, many computers actually have a similar circuit in them already: the cassette interface.

Although the exact method used for placing data on a cassette varies between manufacturers, the basic conversion requirements are the same. In fact, with a little bit of work, it would be quite possible to convert many of the cassette interfaces to communicate over the phone line.

A disadvantage of the 202 modem (in addition to its requiring a high degree of computer control) is that it must be connected directly to the phone line. It is not feasible to acoustically connect the 202 type modem through the telephone handset. The distortion caused by the conversion to sound and back effectively limits the speed of communication. Because of the wideband transmission characteristics of the 202 type modem, this problem is made even worse.

Although it is *possible* to communicate acoustically with the 202 type modem, the distortion effects limit the communication speed to 330 bps or less. This makes it less efficient than the 103 type modem, which can also communicate at 300 baud acoustically, but does not have the extra computer control requirements.

The 103 modem, unlike the 202, requires little or no computer control. In its simplest form – as an acoustic coupler – no control at all is required. Placing the phone call and initiating the communications are all performed by the operator. The 103 is very attractive to the general user because little or no programming is needed.

The 103 could also be connected directly to the phone line, but, again, it has the advantage of requiring little or no control by the computer to operate it. The only real computer control that would be needed would be some software to generate a call automatically if that were wanted. Some modems do not allow automatic calling, while others provide for computer control, and other functions as well.

The Most Popular Model By Far

The 103 modem is the most popular modem available by quite a margin. This is a direct result of its simplicity and ease of use. The 103 has an upper speed limit of operation of 300 bps normally. Some direct connect types, however, can go up to 600 bps. The typical cost of the 103 is between $100 to $200, depending on quality and complexity. Some of the better direct connect 103's can cost $300 to $400 and offer many functions as well as increased reliability.

The 103 cannot communicate with the 202 type. The two modems do not use the same communications scheme. The 202 is generally a half duplex while the 103 is a full duplex modem. While the 202 can communicate in full duplex, it must have two phone lines to do so (one for each direction of communication). The 103 provides the full duplex communications over a single phone. However, it needs more circuitry to perform full duplex, and this is the reason for its increased cost over the 202.

A More Expensive Design

The third main modem design is the 212A. This modem is more expensive than the others, but it combines the functionality of the 103 with the speed capability of the 202. The 212A cannot be acoustically connected to the phone line through the telephone handset, but it does not require any

special computer control (unless that's desired).

The 212A cannot communicate with the 202. It does, however, have a mode of operation which allows it to communicate with the 103 design. In fact, it is two modems in one: the 103 communications method is entirely different from the 212A method. This ability to communicate with the 103 type modem as well as with another 212A is part of the reason for its popularity in spite of its high cost.

The 212A can cost from $500 to $1000. This higher cost is due primarily to the method by which it communicates. It is in actuality a computer and modem combination. This is necessary because the 212A not only converts the data sent to it into the audio signals which go over the phone line, but it also must change the way the data is transmitted. The 212A internally transforms asynchronous data that it receives into a synchronous data stream.

The 103 and 202 modems use a data conversion scheme called FSK (*Frequency Shift Keying*). The 212A, however, uses a different method called PSK (*Phase Shift Keying*). FSK does not require any special handling other than checking that the maximum speed of operation is not exceeded.

The PSK method, however, requires that the data to be sent be synchronized to the audio signal to be sent. By doing this, the 212A is able to make more effective use of the phone line and to allow for true full duplex communications at 1200 bps. The limitation here is that the communications must occur at exactly 1200 bps due to the conversion requirements.

When a 212A is operating in the 103 mode, it reverts to the FSK method to communicate with the 103 design. When placing a call with the 212A, the user must tell the modem which mode it is to use (103 or 212A). When the 212A is receiving a call, however, it will automatically determine which type of modem it is communicating with, switch to that mode (unless it is told otherwise), and tell the user which mode it has selected.

The 212A cannot be acoustically connected to the phone line for the same reason that the 202 type cannot – too much distortion is caused by the conversion telephone handset and by sound conversion. Such a connection would probably be possible if a condenser microphone were installed in the phone, but, since no one is manufacturing acoustically coupled 212A modems, there is not much point in trying this.

Racal Vadic's 3400

There is one manufacturer which has attempted to solve this problem, however. Racal Vadic builds a modem they call a 3400 series. This modem, while able to communicate acoustically over a telephone handset, can only do so to another 3400. The 3400 does this by using yet another communications scheme which is not compatible with any of the previously mentioned modems. The 3400 uses a PSK type of transmission (like the 212A), but it uses a specially designed structure which minimizes the distortion caused by the telephone handset.

The 3400 can also be directly connected to the phone line, and some models, able to communicate with the 103, also have a mode which allows them to communicate with 212A's. The 3400 is not in very widespread use at this time, which might be due to its incompatibility with the 212A.

When deciding which modem to buy, it might be best not to consider the 202 unless you have a specific need for it. It is not in general use and can only communicate with another 202. The 103 is the most common and least expensive, but the 212A, while it costs more, has greater functionality. For portability, the 103 acoustic coupler is probably the best choice since it can communicate with either another 103 or with a 212A. If the portability is unnecessary, and it is acceptable to have the modem directly connected to the phone line, then it becomes simply a matter of deciding how much you are willing to spend for functionality when deciding between a 103 or a 212A design. ©

For the TRS-80 Color Computer, 16K, with either Extended or Non-Extended BASIC, disk or tape – this program will show you the effects of home energy conservation.

Energy Monitor

Linton S. Chastain
Greensboro, NC

Since energy costs have been of major concern to many people in the past few years, here's a BASIC program that has helped me evaluate my energy costs and consumption. The program helps you to determine if those conservation changes that you may have made over the past year are meaningful.

The program keeps track of energy cost and consumption. The first thing you will probably notice is that when energy consumption has remained the same from year to year the cost of that energy has increased. This awareness is enough in itself to inspire conservation measures. Major changes to a home (storm windows, weather stripping, more insulation) can be validated with this program. Pick periods that have the same number of days with similar heating or cooling.

The program was originally written on a 16K Radio Shack Color Computer without Extended BASIC, and it used a cassette recorder for DATA storage. Energy Monitor now uses a disk drive. However, the program should work with few changes on any computer that has at least 8K of user memory available and uses Microsoft BASIC.

If you don't use a disk drive, I will point to what modifications you will have to make to use a cassette recorder. These modifications will be directed toward the Color Computer with at least 8K of user memory with or without Extended BASIC. If you have a different computer, please check your manual on how to store data on cassette.

Here are the changes for those who have cassette recorders:

For Cassette Users.

```
120 PRINT"6-READ OLD MASTER FILE":PRINT:"FR
    OM CASSETTE"
140 PRINT"8-WRITE NEW MASTER FILE":PRINT:"T
    O CASSETTE"
650 PRINT"PRESS THE RECORD AND":PRINT"PLAY ~
    KEY ON CASSETTE"
670 OPEN"O",#-1,T$:PRINT#-1,K:K=1:L=N
700 FORJ=KTOL:PRINT#-1,D$(J),A(J),B(J),C(J)
    ,D(J),E(J),F(J),G(J):PRINTJ:NEXTJ
710 CLOSE
720 PRINT:PRINT"PRESS THE CASSTTE":PRINT"ST
    OP KEY"
750 PRINT"PRESS THE PLAY KEY":PRINT"ON THE ~
    CASSETTE"
790 OPEN"I",#-1,T$:PRINT"READING FILE: "T$:
    INPUT#-1,N
820 FORJ=1TON:INPUT#-1,D$(J),A(J),B(J),C(J)
    ,D(J),E(J),F(J),G(J):PRINTJ:NEXTJ
840 CLOSE
850 PRINT:PRINT"PRESS THE CASSTTE":PRINT"ST
    OP KEY"
```

```
10 REM UTILITIES
20 REM BY STEVE CHASTAIN 1/31/81
30 CLEAR 200
40 MW=20:MR=20:N=0
50 DIMD$(MR),A(MR),B(MR),C(MR),D(MR),E(MR)
    ,F(MR),G(MR),W(MR),WW(MR),X(MR),XX
   (MR),Y(MR),YY(MR),Z(MR),U(MR)
60 R=0:S=0:W=0:WW=0:X=0:Y=0:YY=0:Z=0
65 CLS:PRINT"UTILITIES":PRINT:PRINT"COMMAN
    D LIST # 1"
70 PRINT"1-DISPLAY WATER COST AND UNITS"
80 PRINT"2-DISPLAY GAS COST AND UNITS"
90 PRINT"3-DISPLAY ELECTRIC COST AND"
95 PRINT"UNITS"
100 PRINT"4-DISPLAY TELEPHONE COST"
110 PRINT"5-DISPLAY UTILITIES COSTS AND"
115 PRINT"UNITS"
120 PRINT"6-READ OLD MASTER FILE FROM DISK"
130 PRINT"7-INPUT NEW DATA"
140 PRINT"8-WRITE NEW MASTER FILE TO DISK"
150 PRINT"9-TERMINATE PROGRAM":PRINT
160 INPUT"ENTER COMMAND BY NUMBER";R:IFR<1 ~
    OR R>9 THEN 60
170 ON R GOSUB 970,1170,1370,1570,520,740,1
    80,630,870:GOTO60
180 IFN=MR THEN510
190 PRINT:PRINT"ENTER THE FOLLOWING DATA AS
     REQUESTED"
200 PRINT"-DATE(1/31/81)"
210 PRINT"-WATER COST"
220 PRINT"-WATER UNITS"
230 PRINT"-GAS COST"
240 PRINT"-GAS UNITS"
250 PRINT"-ELECTRIC COST"
260 PRINT"-ELECTRIC UNITS"
270 PRINT"-TELEPHONE COST"
280 N=N+1:PRINT:INPUT"DATE";R$:R$=LEFT$(R$,
    8);D$(N)=R$
290 INPUT"WATER COST";R:A(N)=R;IFR<0 THEN30
    0
300 INPUT"WATER UNITS";R:B(N)=R;IFR<0 THEN3
    10
310 INPUT"GAS COST";R:C(N)=R;IFR<0 THEN320
320 INPUT"GAS UNITS";R:D(N)=R:IFR<0 THEN330
330 INPUT"ELECTRIC COST";R:E(N)=R;IFR<0 THE
    N340
340 INPUT"LECTRIC UNITS";R:F(N)=R;IFR<0 THE
    N350
350 INPUT"TELEPHONE COST";R:G(N)=R;IFR<0 TH
    EN 350
360 PRINT:PRINTTAB(1);"CHECK";TAB(7);"DATE:
    ";D$(N)
370 PRINTTAB(7);"WATER COST:";A(N)
380 PRINTTAB(7);"WATER UNITS:";B(N)
390 PRINTTAB(7);"GAS COST:";C(N)
400 PRINTTAB(7);"GAS UNITS:";D(N)
410 PRINTTAB(7);"ELECTRIC COST:";E(N)
420 PRINTTAB(7);"ELECTRIC UNITS:";F(N)
430 PRINTTAB(7);"TELEPHONE COST:";G(N)
```

```
440 PRINT:PRINTTAB(7)"-IS INPUT O.K.?-":PRI
    NT
450 INPUT"(Y=YES,N=NO,F=YES AND FINISHED)";
    R$:R$=LEFT$(R$,1)
460 IFR$="N" THEN N=N-1:PRINT:PRINT"REDO LA
    ST DATA":GOTO280
470 IFR$="F" THEN RETURN
480 IFR$<>"Y" THEN440
490 IFN=MR THEN510
500 GOTO280
510 PRINT:PRINT"*** NO MORE DATA ALLOWED***
    ":GOSUB940:RETURN
520 IFN<1 THEN PRINT:PRINT"*** NOT ENOUGH D
    ATA***":GOSUB940:RETURN
530 FORJ=1 TO N
540 W=W+W(J):WW=WW+WW(J):X=S+X(J):XX=XX+XX(
    J):Y=Y+Y(J):YY=YY+YY(J);Z=Z+Z(J)
550 U=(W+X+Y+Z)
560 NEXTJ:K=-1:L=0
570 K=K+2:L=L+2:IFL>N THEN L=N
580 CLS:FORJ=K TO L:PRINT@96,"GAS";TAB(14)"
    COSTS";TAB(26)"UNITS":PRINT@64,
    "WATER";TAB(14);W;TAB(26);WW:PRINT@96,"
    GAS";TAB(14);X;TAB(26);XX:PRINT@128,
    "ELECTRIC";TAB(14);Y;TAB(26);YY
585 PRINT]160,"TELEPHONE";TAB(14);Z
590 FORQ=0 TO 31:PRINTCHR$(45);:NEXT
595 PRINT]224,"TOTALS";TAB(13);U
600 NEXTJ:PRINT
610 IFL<=N THENPRINT"HIT ANY KEY FOR COMMAN
    D MODE":GOSUB950:RETURN
620 PRINT"HIT ANY KEY TO CONTINUE":GOSUB950
    :GOTO570
630 IFN<1 THEN PRINT:PRINT"*** NO DATA TO W
    RITE ***":GOSUB940:RETURN
640 R$="WRITING":PRINT
660 INPUT"NAME FOR FILE";T$:K=N:IFN>MW THEN
     K=MW
670 OPEN"O",#1,T$:WRITE#1,K:K=1:L=N
680 IFN>MW THENK=N-MW+1:PRINT"-ONLY LAST";M
    W"VALUES WILL BE WRITTEN"
690 PRINT"WRITING FILE:";T$:PRINT"    RECOR
    DS #";
700 FORJ=K TO L:WRITE#1,D$(J),A(J),B(J),C(J
    ),D(J),E(J),F(J),F(J),G(J):PRINTJ:
    NEXTJ
710 CLOSE#1
730 PRINT"PRESS THE KEYBOARD'S ENTER KEY.":
    GOSUB950:RETURN
740 R$="READING":PRINT
780 INPUT"NAME OF FILE ";T$
790 OPEN"I",#1,T$:PRINT"READING FILE: ";T$:
    INPUT#1,N
800 IFN>MR THEN PRINT"*** TOO MANY FILES ON
     DISK ***":END
810 PRINT"READING RECORDS # ";
820 FORJ=1 TO N:INPUT#1,D$(J),B(J),C(J),D(J
    ),E(J),F(J),G(J):PRINTJ:NEXTJ
830 PRINTN ;" DATA RECORDS READ"
840 CLOSE#1
860 PRINT"PRESS THE KEYBOARD'S ENTER KEY.":
    GOSUB950:RETURN
870 END
940 FORQ=1 TO 1000:NEXTQ:RETURN
950 B$="":R$=INKEY$-IFR$=B$ THEN950
960 RETURN
970 CLS:PRINT"WATER":PRINT:PRINT"COMMAND LI
    ST # 2"
980 PRINT"1-DISPLAY WATER"
990 PRINT"2-RETURN TO COMMAND LIST # 1"
1000 INPUT"ENTER COMMAND BY NUMBER";R:IFR<1 ~
    OR R>2 THEN970
1010 ON R GOSUB 1020,1110:GOTO970
1020 IFN<1 THEN PRINT:PRINT"*** NOT ENOUGH D
    ATA ***":GOSUB1140:RETURN
1030 FORJ=1 TO N
1040 R=A(J):S=B(J):W(J)=R:WW(J)=S
1050 W=W(J):WW=WW(J)
1060 NEXTJ:K=-3:L=0
1070 K=K+4:L=L+4:IFL>N THENL=N
1080 CLS:PRINT"DATE";TAB(14);"COST";TAB(26);
    "UNITS"
1085 FORJ=K TO L:PRINTD$(J);TAB(14);A(J);TAB
    (26);B(J);NEXTJ:PRINT
1090 IFL=N THEN PRINT"HIT ANY KEY FOR COMMAN
    D MODE ":GOSUB1150:RETURN
1100 PRINT"HIT ANY KEY TO CONTINUE":GOSUB115
    0:GOTO1070
1110 GOTO60
1120 R$=INKEY$:IFR$=B$ THEN 1120
1130 RETURN
1140 FORQ=1 TO 1000:NEXTQ:RETURN
1150 B$="":R$=INKEY$:IFR$=B$ THEN 1150
1160 RETURN
1170 CLS:PRINT"GAS":PRINT:PRINT"COMMAND LIST
     # 3"
1180 PRINT"1-DISPLAY GAS"
1190 PRINT"2-RETURN TO COMMAND LIST # 1"
1200 INPUT"ENTER COMMAND BY NUMBER";R:IFR<1 ~
    OR R>2 THEN 1170
1210 ON R GOSUB 1220,1310:GOTO1170
1220 IFN<1 THEN PRINT;PRINT"*** NOT ENOUGH D
    ATA ***":GOSUB1340:RETURN
1230 FORJ=1 TO N
1240 R=C(J):S=D(J):X(J)=R:XX(J)=S
1250 X=X(J):XX=XX(J)
1260 NEXT J:K=-3:L=0
1270 K=K+4:L=L+4:IFL>N THENL=N
1280 CLS:PRINT"DATE";TAB(14);"COST";TAB(26);
    "UNITS"
1285 FORJ=K TO L:0PRINTD$(J);TAB(14);C(J);TA
    B(26);D(J):NEXTJ:PRINT
1290 IFL=N THEN PRINT"HIT ANY KEY FOR COMMAN
    D MODE":GOSUB1350:RETURN
1300 PRINT"HIT ANY KEY TO CONTINUE":GOSUB135
    0:GOTO1270
1310 GOTO60
1320 R$=INKEY$:IFR$=B$ THEN 1320
1330 RETURN
1340 FORQ=1 TO 1000:NEXTQ:RETURN
1350 B$="":R$=INKEY$:IFR$=B$ THEN 1350
1360 RETURN
1370 CLS:PRINT"ELECTRIC":PRINT:PRINT"COMMAND
     LIST # 4"
1380 PRINT"1-DISPLAY ELELCTRIC"
1390 PRINT"2-RETURN TO COMMAND LIST # 1"
1400 INPUT"ENTER COMMAND BY NUMBER";R:IFR<1 ~
    OR R>2 THEN 1370
1410 ON R GOSUB 1420,1510:GOTO1370
1420 IFN<1 THEN PRINT:PRINT"*** NOT ENOUGH D
    ATA ***":GOSUB1540:RETURN
1430 FORJ=1 TO N
1440 R=E(J):S^F(J):Y(J)=R:YY(J)=S
1450 Y=Y(J):YY=YY(J)
1460 NEXTJ:K=-3:L=0
1470 K=K+4:L=L+4:IFL>N THEN L=N
1480 CLS:PRINT"DATE";TAB(14);"COST";TAB(26);
    "UNITS"
1485 FORJ=K TO L:PRINTD$(J);TAB(14);E(J);TAB
    (25);F(J):NEXTJ:PRINT
1490 IFL=N THEN PRINT"HIT ANY KEY FOR COMMAN
    D MODE":GOSUB1550:RETURN
1500 PRINT"HIT ANY KEY TO CONTINUE":GOSUB155
```

```
    0:GOTO1470
1510 GOTO60
1520 R$=INKEY$:IFR$=B$ THEN1520
1530 RETURN
1540 FORQ=1 TO 1000:NEXTQ:RETURN
1550 B$="":R$=INKEY$:IFR$=B$ THEN 1550
1560 RETURN
1570 CLS:PRINT"TELEPHONE":PRINT:PRINT"COMMAN
    D LIST # 5"
1580 PRINT"1-DISPLAY TELEPHONE"
1590 PRINT"2-RETURN TO COMMAND LIST # 1"
1600 INPUT"ENTER COMMAND BY NUMBER";R:IF R<1
     OR R>2 THEN1570
1610 ON R GOSUB 1620,1710:GOTO1570
1620 IFN<1 THEN PRINT:PRINT"*** NOT ENOUGH D
    ATA ***":GOSUB1740:RETURN
1630 FORJ=1 TO N
1640 R=G(J):Z(J)=R
1650 Z=Z(J)
1660 NEXTJ:K=-7:L=0
1670 K=K+8:L=L+8:IFL>N THEN L=N
1680 CLS:PRINT"DATE","COST"
1685 FORJ=K TO L:PRINTD$(J),G(J);NEXTJ:PRINT
1690 IFL=N THEN PRINT"HIT ANY KEY FOR COMMAN
    D MODE":GOSUB1750:RETURN
1700 PRINT"HIT ANY KEY TO CONTINUE":GOSUB175
    0:GOTO1670
1710 GOTO60
1720 R$=INKEY$:IFR$*B$ THEN1720
1730 RETURN
1740 FORQ=1 TO 1000:NEXTQ:RETURN
1750 B$="":R$=INKEY$:IFR$=B$ THEN1750
1760 RETURN                                  ©
```

Animate your Atari players – this set of programs creates the illusion of motion using only four drawings. And it's simple to add additional players which can each move independently.

Animation And P/M Graphics

Tom Sak and Sid Meier
Baltimore

You're already familiar with the Atari's ability to rapidly move a player from one location to another. But there are many times when you would like to do more than simply move a player; you'd like to give it lifelike motion, or animation. Spend a few minutes and learn how you can achieve these effects with far less effort than you might have imagined.

The art of bringing life to still pictures is much older than many of us realize. The production of books which contained moving pictures was well established before the invention of the motion picture camera and projector. The effect of moving pictures was typically accomplished by rapidly flipping the pages of a booklet containing simple character drawings, making them seem to spring to life.

Walt Disney and numerous other animators have produced this illusion of motion by drawing series of pictures in which each picture differs from the previous one only in a very small detail, a subtle displacement of each moving element. The pictures are then photographed for subsequent projection.

For example, an animator draws a man who appears to raise his arm away from his side, using a sequence of drawings. The first drawing would show the man facing you with both arms at his sides. The second picture differs only in that one arm is now slightly away from the man's side. The next picture shows the arm slightly further away, and so on through the sequence of drawings.

Animate With Only Four Drawings

As each picture in the series is viewed in rapid succession, by flipping through the stack of drawings, the figure appears to be raising his arm away from his side. A motion picture film consists of an analogous sequence of pictures which also provide the illusion of motion when they are projected and viewed in rapid succession.

As you can well imagine, a very large number of drawings is required to produce even a relatively short motion picture sequence. Since you're not about to adapt *Fantasia* for the small screen attached to your Atari, we will show you a way to use only four drawings, repeated in a cyclical pattern, to produce the illusion of motion. This is a very effective shortcut which makes it practical to adapt the animator's techniques to your BASIC program.

Now for some Atari animation. There is no question that our artistic creativity and graphic talents may never rival those of Walt Disney, but we will endeavor to adapt the basic animation technique which he popularized in order to move four "cowboys" from right to left across your television screen, totally out of step with each other.

For illustrative purposes we'll begin by moving only one cowboy. Program 1 accomplishes this objective by using the automatic player-missile graphic manipulation of the vertical blank interrupt routine which we discussed in **COMPUTE!**, February 1982, #21. Those of you who have entered the example program in that article will be pleased to know it already contains the animation features described here.

Program 2 adds complexity to the one cowboy program, illustrating the asynchronous movement of four players. Developing an understanding of the more complex program won't be too difficult once you've grasped the concepts in Program 1.

Reviewing Vertical Blank Interrupts

An elementary understanding of our vertical blank interrupt routine, VBLANK PM, is a prerequisite. Here we will review highlights of our previous article.

VBLANK PM is a machine language subroutine which occupies a portion of memory page six. It is initialized by a single BASIC USR function call which causes VBLANK PM to notify the operating system of both its presence and its desire to be automatically invoked during each vertical blank interrupt.

Prior to initialization, a 2K (2048) byte memory allocation must be made for the storage of players, and the players must be drawn. Following initialization, a POKE of the x-axis (horizontal) and y-axis (vertical) screen coordinates is all that is required to cause a player to be automatically moved during the next vertical blank period, or approximately every 1/60 of a second.

Not mentioned in the previous article is the fact that VBLANK PM has an animation feature just waiting to bring life to your players. All you need do is supply a few more drawings. The drawings and the current display image are contained in the 2K byte storage block.

Players Are Stored As Separate Images

Figure 1 depicts the memory allocated for the

storage of players (see line 1030 in Program 1; memory allocation is explained in our earlier article). The current displayed image of player zero resides at locations PMBASE+1024 through PMBASE+1279; player one's homestead is PMBASE+1280 through PMBASE+1535, and so on for the other two players.

To achieve the animation, you need more than one image of each player, so the lower 1K (1024) locations (PMBASE through PMBASE+1023) of the 2K byte storage block are used to hold the necessary set of drawings. Each player's drawings are stored in an area of memory beginning at a location which is 1K bytes below (lower memory address) the player's position in the upper 1K portion of the 2K byte storage block. A drawing is copied to the upper 1K portion by VBLANK PM when it is to be displayed. As a matter of fact, you won't draw anything at all in the upper 1K locations but will let VBLANK PM look after this chore for you.

For example, all of the player zero drawings reside at the 256 locations beginning at PMBASE. The currently displayed image of player zero resides at locations PMBASE+1024 through PMBASE+1279. The drawings for player zero are stored 1024 locations below this point, which is equal to PMBASE+1024 minus 1024, or simply PMBASE. The player one drawings begin at PMBASE+256, or (PMBASE+1280)-1024, and so on for players two and three at locations PMBASE+512 and PMBASE+768, respectively.

A note of caution: we mentioned in the previous article that you could use the lower 1K bytes for your own purposes without disturbing anything. This is true only when the VBLANK PM animation feature is not going to be used. We hope that you've not been led too far astray!

At the risk of stating the obvious, we'd like to mention that as soon as you've decided to use more than one drawing per player – which you must do in order to achieve the animation – you can no longer have a player which is 255 lines tall. This is true because there are only 256 locations in which to store all of the drawings necessary to animate a single player. The first position, location zero, of each storage bin is reserved for a reason discussed later.

Initialize The Vertical Blank Routine

Now let's turn our attention to Program 1. Line numbers ending in zero are unchanged since the February article; and, for those who previously keyed the lengthy DATA statements containing VBLANK PM, we've made no changes to the machine language subroutine.

Lines 105 through 205 are the main program which causes our ragtag cowboy to meander across the screen. The BASIC code required to load and initialize VBLANK PM is found on lines 1000 through 1110. The VBLANK PM machine language subroutine is represented as DATA in lines 2000 through 2100. Finally, lines 3005 through 3045 contain the four drawings, used to describe a single player.

Before reviewing the main program, we'll go over the initialization subroutine which performs three functions: load VBLANK PM, load the player's drawings, and initialize VBLANK PM.

Lines 1010 and 1020 cause VBLANK PM to be read from DATA statements and POKEd into memory page six. A more memory-efficient method of representing VBLANK PM is the use of a string variable instead of DATA statements. Using this alternative, you continue to POKE the VBLANK PM code into page six, but from the string variable instead of from DATA statements.

You would save memory because only a single byte of memory is required in the string variable assignment statement to represent a byte of machine language code. In the DATA statement, as many as three bytes may be required for the same thing. For certain other machine language code applications, you can directly execute from the string, eliminating the need to POKE the code into another memory location. If you're interested in more on this topic, look for the article "Creating and Using Program Storage Strings" in this issue.

How The Animation Works

Line 1030 acquires the 2K byte memory storage block and line 1040 assures that the upper 1K byte display portion is cleared. Lines 1045 through 1065 are responsible for reading and storing the player's drawings in the lower 1K byte portion of the storage block. The four drawings of a cowboy are illustrated in Figure 2; you see now why Disney Studios can rest easy!

Notice that in line 1045 the first location in which the first drawing is stored is established as one byte above PMBASE; you will learn why this is necessary in a minute. The FOR statement on line 1055 assures that four drawings (zero through three) are read and stored. Each drawing is 24 lines tall, so we begin the FOR loop on line 1065 with the base of the first drawing offset by 24 bytes for each previous drawing stored. Since each drawing consists of 24 bytes, the loop is completed by adding 23 to the starting point.

Line 1075 designates the player's color. Line 1080 establishes the locations to be POKEd to change the player's x-axis and y-axis screen coordinates (PLX and PLY) and to set the length (height)

The Hard & Soft of It:
ATARI from ASAP

If you're into Atari, get into ASAP. When it comes to Atari, ASAP has it all — computers, a full line of accessories, and one of the most complete lines of software on the market. Whether you use your Atari for work or play, make ASAP your source. Call today.

The **Atari® 800™** Computer features color graphics and English characters with truly high resolution, high quality sound, expandable memory and sleek modular appearance.

Atari® 400™ — 16K also available: **$349.00.**

OPTIONAL ACCESSORIES	PRICE
ATARI® 410™ Program Recorder	80.00
ATARI® 810™ Disk Drive	470.00
ATARI® 822™ Thermal Printer	299.00
ATARI® 820™ 40-column Dot Matrix Impact Printer	279.00
ATARI® 825™ 80-column Dot Matrix Impact Printer	645.00
ATARI® 830™ Acoustic Modem	159.00
ATARI® 850™ Interface Module	175.00
ATARI® Paddle (CX30-04) and Joystick (CX40-04) (Pair)	17.95
ASAP 16K RAM Module	45.00

COMPLETE SOFTWARE LIBRARY INCLUDES THESE POPULAR UNITS:

Atari®

Video Easel ROM	26.00
Music Composer ROM	44.00
Assembler/Editor ROM	45.00
Mortgage Loan Analysis Cassette	15.95
Stock Analysis Disk	19.95
Stock Charting	22.95
Bond Analysis Disk	22.95
Mailing List Cassette	19.95
Touch Typing (2 Cass)	19.95
Graph It (2 Cass)	17.95
Word Processor	119.00
Personal Finance	64.95
Microsoft Basic	75.00
Basketball ROM	27.00
Super Breakout ROM	33.00
Computer Chess ROM	32.00
3D Tic Tac Toe ROM	26.00
Star Raiders ROM	37.00
Kingdom Cassette	12.95
Blackjack Cassette	12.95
Biorhythm Cassette	12.95
Energy Czar Cassette	12.95
Telelink ROM	24.00
Space Invaders ROM	33.00
Scram	18.95
Asteroids	32.00
Missile Command	32.00
Caverns of Mars Disk	32.00
Centipede Disk	35.00
Pac-Man	35.00

Arcade Plus

Ghost Hunter Cassette	23.50
Ghost Hunter Disk	27.50

Datasoft

Atari Mailing List Disk	19.95
Atari Character Generator Disk	16.95
Text Wizard Disk	89.95
Micropainter Album 1 Disk	16.95
Micropainter Album 2 Disk	16.95
Le Stick Accessory	30.00

On-Line Systems

HI-RES Adv #0 — Mission: Asteroid Disk	19.95
HI-RES Adv #2 — Wiz & Princess Disk	25.00
Roadwork Disk	29.95
Jawbreaker Disk	25.00
Softporn Adventure (X-rated) Disk	25.00
The Next Step Disk	29.95

Personal Software

Visicalc Disk	169.00

FOR HOME ENTERTAINMENT SYSTEMS:

Activision

Dragster Cartridge	17.95
Boxing Cartridge	17.95
Checkers Cartridge	17.95
Fishing Derby Cartridge	17.95
Skiing Cartridge	17.95
Bridge Cartridge	22.00
Tennis Cartridge	17.95
Laser Blast Cartridge	17.95
Freeway Cartridge	17.95
Kaboom! Cartridge	17.95
Stampede Cartridge	17.95
Ice Hockey Cartridge	22.00

asap computer products, inc.

1198 E. Willow St., Signal Hill, CA 90806

ALL PRICES SUBJECT TO CHANGE WITHOUT NOTICE. CALL FOR BEST PRICE

ASAP offers a 15-day buyer protection policy: full money-back guarantee if not totally satisfied.

Ordering information: name, address, phone; ship by: UPS or Mail. Shipping charge: add $2.90 up to 1 lb. (UPS blue), U.S. Mail add $1.50 (U.S. only) ($25.00 minimum order).

Terms: We accept cash, check, money orders, Visa and Master Charge (U.S. funds only). Tax: 6% Calif. res., COD's and terms available on approval (School PO's Accepted).

Toll free outside California: **(800) 421-7701.** Inside California: **(213) 595-6431 (714) 891-2663**

Inside Canada call **ASAP Computer Products, LTD.**, 116 Viceroy Rd., D-12 Concord, Toronto, Ontario, Canada L4K 1A9 **(416) 738-0500 (800) 268-1996**

of the player (PLL).

The x-axis screen display position for players zero, one, two, and three are indicated by POKEs to PLX, PLX + 1, PLX + 2 and PLX + 3, respectively. The analogous situation is true for setting the player's y-axis coordinate (PLY, PLY + 1, ...) and the player's height (PLL, PLL + 1, ...), and for selecting the next drawing to be displayed (PDR, PDR + 1, ...).

PDR is defined on line 1085 and is used to select the next drawing to be used as the player's current display image. VBLANK PM is responsible for copying the drawing to the appropriate location in the upper 1K byte portion of the 2K byte storage block. A value in the range of one to 255 is POKEd into PDR to indicate the bottom-most line of the selected drawing. The most recent value POKEd into PLL indicates the number of bytes (the height of the player) to be copied.

VBLANK PM Must Announce Itself

A value of zero POKEd into PDR signals VBLANK PM to continue to display the current image. This is why we were careful to avoid location zero when loading the first drawing. VBLANK PM sets PDR to zero automatically after it copies a drawing to the upper 1K byte display area.

Location 1771, POKEd in line 1085, is a location in VBLANK PM which must contain the memory page number of the first page in which drawings are stored. Location 1788, referenced on line 1090, is also in VBLANK PM, and must contain the page number of the beginning of the upper 1K byte current display portion. (These parameters afford even greater flexibility to VBLANK PM, features which are beyond the scope of this discussion.)

The other POKEs on line 1090 are associated with the Atari's player-missile graphics mechanism which is described in numerous other articles including our February article.

VBLANK PM is initialized on line 1100. This is the only explicit BASIC function call to VBLANK PM which is required. As a result of this call, VBLANK PM will register its intention to become a part of the vertical blank interrupt process with the operating system.

Inside The Main Routine

Turning our attention to the main program, we start with line 105, which establishes the television screen background, or playfield. It is important that you always define a graphics mode (execute a graphics statement) before you initialize VBLANK PM; if you fail to follow this sage advice, you are likely to be plagued by a strange flashing vertical bar on your screen.

It doesn't matter which graphics mode is specified since Atari players are independent of the mode. Graphics mode one is chosen to provide a text window to serve as a walkway for our strolling cowboy. Line 125 sets the y-axis position of the cowboy so he appears to walk on top of the text window. The player's height is also established on line 125.

The animation is performed by lines 135 through 205. These lines should be relatively easy to comprehend once you have a mental picture of the way in which the drawings were stored during the initialization procedure. The variable DRAW, initialized as one on line 135, selects the next drawing to be used as the current display image.

Lines 145 and 165 control the right to left motion of the cowboy by using the index variable I as the x-axis coordinate of the player. The POKE to PDR on line 185 selects the next drawing to be displayed, and the calculation on line 195 results in the selection of the drawing to be used in the next cycle when the cowboy takes his next step.

The IF statement on line 195 assures that after the fourth drawing is used, the program will cycle and begin anew with the first drawing. The FOR loop on line 205 controls the speed with which the cowboy strolls across the screen. A maximum value of 30 results in a movement which you might describe as a brisk walk. The larger the maximum value of this delay loop, the slower the pace of the player.

The cowboy will continue to walk across the screen until you stop the program. Incidentally, the program does not gracefully turn off the Atari's player-missile graphics mechanism, so you are well advised to press SYSTEM RESET to remove the undesirable residue from the screen. (POKE 53277,0 turns off the player-missile gracefully.) Be patient when the program is started, since it takes more than ten seconds for the initialization procedure.

Four Heads Are Better Than One

And that's almost all there is to animation! Are you ready to tackle a little bit more challenging project? Program 2 represents enhancements to the program we've been reviewing. It uses all four players and, while it causes them to walk out of step with each other, it employs only the same four drawings.

Program 2 modifies seven lines and adds two more. The changed lines are: 125, 165, 185, 205, 1045, 1055 and 1075; lines 155 and 175 are new.

Line 1045 now includes a FOR statement to cause the drawings to be READ and POKEd in the storage area associated with the additional three players. Note also that the calculation of DRWBAS is revised to reflect the additional players. DRWBAS contains the address of the first byte of the drawing

storage area containing the first drawing for the current player. As the value of the variable, I, in the FOR loop is indexed from 0 to 3, DRWBAS will take the values 1, 257, 513 and 769. The first byte, location 0, of each storage area is skipped for the reason mentioned earlier.

A RESTORE statement is added to line 1055 which resets the DATA pointer to reread the same drawings for each player. The modification to line 1075 is simply the addition of player colors for the new players.

Looking at the main program, line 125 now establishes the y-axis and height for four players rather than one. Line 155 is added to cycle through the x-axis movement and picture selection for all players.

In line 165 we've added a calculation to the x-axis positioning POKE to maintain a separation between the cowboys which is equal to slightly more than the width of a single player as measured from the leftmost edge of one player to the leftmost edge of the following player.

Still Only Four Drawings

Line 175 is added to assure that a different drawing is used as the current display image for each player. The variable DRAW continues to determine the drawing to be selected for player zero. Study the statement, and you will discover that each player will be depicted by the drawing following that used for the previous player. That is, if player zero is pictured by the first drawing, then player one is illustrated by the second, player two by the third, and, finally, player three is displayed as the fourth drawing. A circular assignment is used so that the fourth drawing is followed by the first.

The delay loop is omitted from line 205 because the additional calculations needed for the added players consume sufficient time to maintain a reasonable pace for all four cowboys. You might want to experiment with a delay loop to further slow the action; better yet, consider using GET to accept a keystroke instead of employing a delaying FOR loop. The GET will allow you to step the players across the screen in order to study the animation technique.

Don't you agree that animation makes a world of difference in the use of player-missile graphics? I was fascinated when my more talented partner, Sid, gave me a half dozen lines of cryptic BASIC statements to turn into an animation tutorial. The first time I saw them execute I was mesmerized. Go ahead, type either program into your Atari; you'll be addicted too.

Figure 1.

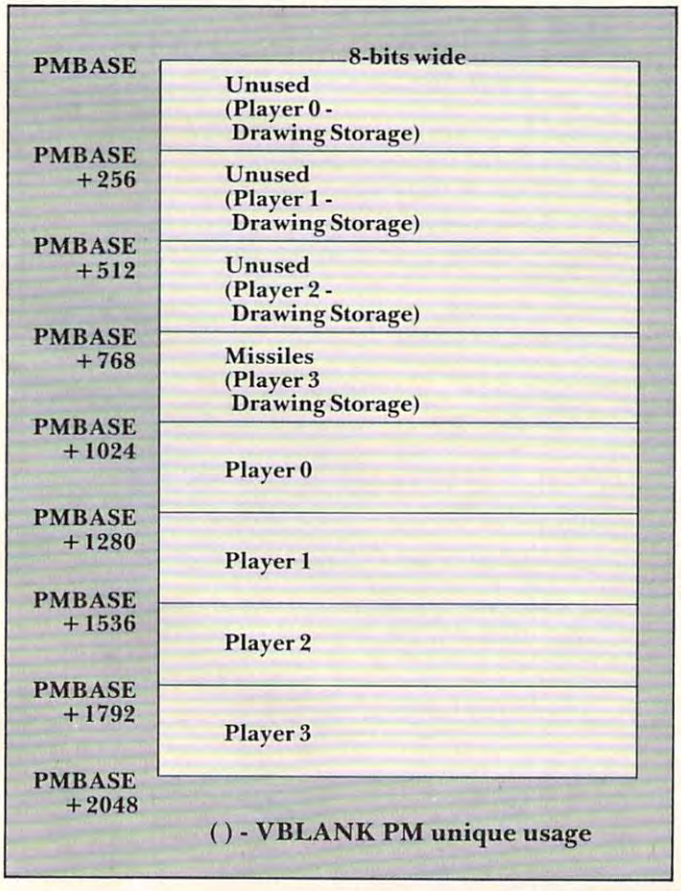
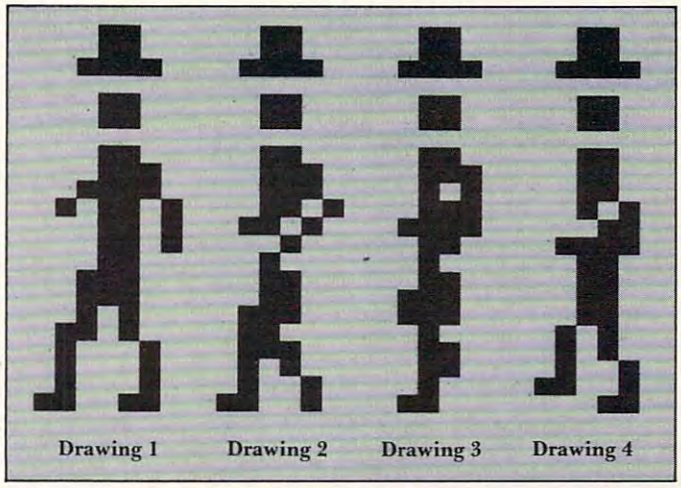


Figure 2.



Program 1.

```
5 REM .... P R O G R A M{4 SPACES}O N E ....
105 GRAPHICS 1:SETCOLOR 2,1,8:SETCOLOR 4,8,4
    :POSITION 5,3:? #6;"animation":POSITION
    3,5:? #6;"demonstration"
120 GOSUB 1000:REM initialize vb routine
125 POKE PLY,169:POKE PLL,24
135 DRAW=1
145 FOR I=212 TO 10 STEP -1:REM move rt to l
    ft horiz
165 POKE PLX,I:REM new position
185 POKE PDR,DRAW:REM new drawing
195 DRAW=DRAW+24:IF DRAW>73 THEN DRAW=1:REM
    select next drawing
```

```
205 FOR DELAY=1 TO 30:NEXT DELAY:NEXT I:GOTO
    145
1000 REM INITIALIZE VBLANK PM SUBR
1010 FOR I=1536 TO 1706:READ A:POKE I,A:NEXT
     I
1020 FOR I=1774 TO 1787:POKE I,0:NEXT I
1030 PM=PEEK(106)-16:PMBASE=256*PM
1040 FOR I=PMBASE+1023 TO PMBASE+2047:POKE I
     ,0:NEXT I
1045 DRWBAS=PMBASE+1
1055 FOR J=0 TO 3:REM four drawings
1065 FOR K=DRWBAS+J*24 TO DRWBAS+J*24+23:REA
     D X:POKE K,X:NEXT K:NEXT J
1075 POKE 704,12
1080 PLX=53248:PLY=1780:PLL=1784
1090 POKE 559,62:POKE 623,1:POKE 1788,PM+4:P
     OKE 53277,3:POKE 54279,PM
1095 PDR=1772:POKE 1771,PM
1100 X=USR(1696)
1110 RETURN
2000 REM vblank interupt routine
2010 DATA 162,3,189,244,6,240,89,56,221,240,
     6,240,83,141,254,6,106,141
2020 DATA 255,6,142,253,6,24,169,0,109,253,6
     ,24,109,252,6,133,204,133
2030 DATA 206,189,240,6,133,203,173,254,6,13
     3,205,189,248,6,170,232,46,255
2040 DATA 6,144,16,168,177,203,145,205,169,0
     ,145,203,136,202,208,244,76,87
2050 DATA 6,160,0,177,203,145,205,169,0,145,
     203,200,202,208,244,174,253,6
2060 DATA 173,254,6,157,240,6,189,236,6,240,
     48,133,203,24,138,141,253,6
2070 DATA 109,235,6,133,204,24,173,253,6,109
     ,252,6,133,206,189,240,6,133
2080 DATA 205,189,248,6,170,160,0,177,203,14
     5,205,200,202,208,248,174,253,6
2090 DATA 169,0,157,236,6,202,48,3,76,2,6,76
     ,98,228,0,0,104,169
2100 DATA 7,162,6,160,0,32,92,228,96
3005 REM drawings 0, 1, 2 and 3
3015 DATA 0,12,12,30,0,12,12,0,12,14,30,45,1
     3,13,12,28,28,20,52,34,34,34,102,0
3025 DATA 0,12,12,30,0,12,12,0,12,14,14,13,2
     6,4,8,12,12,28,24,28,20,18,50,0
3035 DATA 0,12,12,30,0,12,12,0,12,14,10,14,3
     0,12,8,12,28,28,8,12,12,8,24,0
3045 DATA 0,12,12,30,0,12,12,0,12,12,12,10,6
     ,30,12,12,12,12,20,20,18,50,6,0
```

Program 2.

This program uses the Vertical Blank Player/Missile routine, so add lines 2000-3045 of Program 1 when you type it in.

```
5 REM .... P R O G R A M{4 SPACES}T W O ....
105 GRAPHICS 1:SETCOLOR 2,1,8:SETCOLOR 4,8,4
    :POSITION 5,3:? #6;"animation":POSITION
    3,5:? #6;"demonstration"
120 GOSUB 1000:REM initialize vb routine
125 FOR J=0 TO 3:POKE PLY+J,169:POKE PLL+J,2
    4:NEXT J
135 DRAW=1
145 FOR I=212 TO 10 STEP -1:REM move rt to l
    ft horiz
155 FOR J=0 TO 3:REM four players
165 POKE PLX+J,I+J*10:REM new positioN, main
    tain separation
175 NXTDRW=DRAW+J*24:IF NXTDRW>73 THEN NXTDR
    W=NXTDRW-96:REM select different drawing
     for each player
185 POKE PDR+J,NXTDRW:NEXT J
195 DRAW=DRAW+24:IF DRAW>73 THEN DRAW=1:REM
    select next drawing
205 NEXT I:GOTO 145
1000 REM INITIALIZE VBLANK PM SUBR
1010 FOR I=1536 TO 1706:READ A:POKE I,A:NEXT
     I
1020 FOR I=1774 TO 1787:POKE I,0:NEXT I
1030 PM=PEEK(106)-16:PMBASE=256*PM
1040 FOR I=PMBASE+1023 TO PMBASE+2047:POKE I
     ,0:NEXT I
1045 FOR I=0 TO 3:DRWBAS=PMBASE+I*256+1:REM
     four players
1055 RESTORE 3015:FOR J=0 TO 3:REM four draw
     ings
1065 FOR K=DRWBAS+J*24 TO DRWBAS+J*24+23:REA
     D X:POKE K,X:NEXT K:NEXT J:NEXT I
1075 POKE 704,12:POKE 705,128:POKE 706,48:PO
     KE 707,192
1080 PLX=53248:PLY=1780:PLL=1784
1090 POKE 559,62:POKE 623,1:POKE 1788,PM+4:P
     OKE 53277,3:POKE 54279,PM
1095 PDR=1772:POKE 1771,PM
1100 X=USR(1696)
1110 RETURN
```

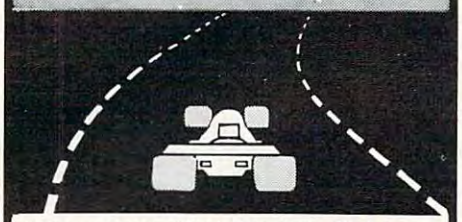
Royal Software

YOUR MARKETPLACE FOR:

* ATARI

WE DEAL EXCLUSIVELY IN PRODUCTS
FOR THE ATARI (THE BEST) COMPUTER

NIGHT RALLY



Test your driving skills to their limits. Climb into the driver's seat as you race against some of the best computer drivers in the world.

16K DISK $31.40
TAPE $26.90

Descend through mazes into the Dungeons with your squadron of warriors to battle with visible and invisible monsters.

Wizard of Wor

16K DISK ONLY $35.10

PRO-FOOTBALL

Play against the computer or another player. Over 25 offense and defense plays. The BEST football game yet.

16K
$26.90 TAPE
$31.40 DISK

HARDWARE

800 Computer 16K	$648
800 Computer 48K	$757
400 Computer 16K	$338
810 Disk Drive	$448
850 Interface	$178
410 Recorder	$ 78
MX80FT+ Printer	$636
16K Ram module	$ 69
32K Ram module	$109
Graphics Tablet	$278

Trade your 400 or older 800 for the new model 800 — Call for prices

SOFTWARE

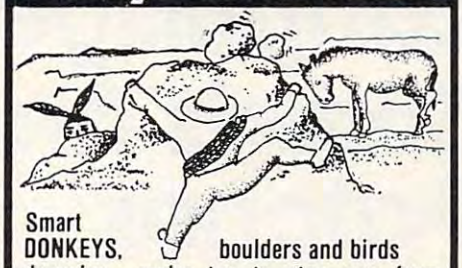
PAC MAN (cart.)	$39.50
Centipede (cart.)	$39.50
Microsoft Basic (D) 32K	$79.10
Ghost Hunter (T) 16K	$25.50
Ali Baba (D) 32K	$28.90
Protector (T) 32K	$22.45
Apple Panic (D) 48K	$26.90
Threshold (D) 40K	$33.90
GORF (D) 16K	$35.10
Crypts of Terror (D,T) 16K	$26.30
Text Wizard (D) 32K	$87.90
Dodge Racer (T) 16K	$20.20
Chicken (T) 16K	$20.20
De Re Atari Book	$17.90
Atari Games Book	$13.90
Personal Finance (D) 40K	$67.50
Zork I or II (D) 32K	$35.90
Deluxe Invaders (D) 16K	$31.40
Eastern Front (T,D)	$25.40
Mousekattack (D) 32K	$31.40
Galactic Chase (T) 16K	$22.40
Bug Attack (T,D) 32K	$26.90
Pacific Coast Hwy (T,D) 16K	$26.90
Alien Swarm (T) 16K	$26.90
Midnight Magic (D) 48K	$31.45
Intruder (T) 16K	$26.90
Nautilus (T,D)	$20.20

Call or Write for

FREE

ATARI PRODUCTS CATALOG
with hundreds of items

Canyon Climber



Smart DONKEYS, boulders and birds dropping rocks try to stop you from reaching the top. A real challenge. Arcade quality.

16K
DISK $26.90 TAPE $22.40

CROSSFIRE

The aliens have landed and it's your job to save the city. (If you can!) TOP RATE GAME, ARCADE QUALITY, HI-RES Graphics & Sound.

16K TAPE
32K DISK
$26.90

FROGGER

This is the genuine FROGGER game that you see in the arcades. Made by the same people that made Jawbreaker (One of the top ten sellers.)

16K DISK OR TAPE
$31.40

VISA · master charge

TO ORDER CALL TOLL FREE 800-452-8013

FOR INFORMATION
CALL 503-683-5361

HOW TO ORDER: Send check or money order or call our toll free number and use your Visa Card. Shipping on software is $2.00 per order anywhere in USA. Hardware shipping call for cost. Add 3% for VISA or MC. Equipment subject to price change and availability without notice.

Store Hours
8 am — 6 pm
Mon. — Sat.

Royal Software P.O. Box 10303
Eugene, Oregon 97440

Atari is Trademark of Atari Inc.

A Monthly Column

Machine Language: Shreds And Patches

Jim Butterfield
Toronto

When you write a program, you expect it to be perfect. Sometimes it misses the perfection you expected, and you have to fix it.

After serious debugging you isolate the fault – or one of them, at least. Occasionally, it's a single instruction that's wrong – that LDA (Load A) should have been LDY (Load Y). You can fix it by correcting the hexadecimal Op Code and immediately go for another run. Rarely, it's code that you don't need: instead of storing zero into location 96, do nothing. Again, you fix it by overwriting. Change the unwanted Store instruction into a do-nothing NOP (No Operation) – or, more accurately, a series of NOP's – and the program is ready to go.

The annoying problem is the most common – you've left something out of the code and need to shoehorn it in somehow. You need to find empty space within your program – and there's no space there.

Classic Correction

The classic answer to program repairs is to redo the program. If you have a symbolic assembler, this isn't hard. You add the missing instructions, call in the assembler, and a new program is generated. Unused space is closed up, new space is created as necessary, and all the branches, jumps, and calls are recalculated. It's ideal, but we don't always go that way.

Why not always reassemble? There are a number of valid reasons.

Sometimes the owner of a small computer doesn't have an assembler; perhaps his system isn't big enough to support one. He'll be assembling by hand, or by using tiny assemblers like the one in Supermon. A new assembly means a lot of work.

Even when an assembler is available, the user can perceive it as a lot of trouble to use during a debugging session. The test program must be thrown out and the assembler loaded; a new "object" program – that's a machine language program – must be created. The clincher is a paperwork problem: to reassemble and do the job right, the new program should be dated and version-numbered; and then a program listing should be generated. That's potentially a lot of paper and a lot of printing time. Yet it's needed, since the programmer will need to know where the code is located during testing.

What's the alternative? A simple procedure known as "patching" can add corrections to a program without the work of a full assembly.

Patches

The principle of a patch is this: to add new code, you must destroy some of the old code by overwriting it with a Jump instruction. The Jump will take you to a fresh part of memory (the patch area) in which the old code will be reconstructed and the new code added. Finally, the patch program will Jump back and allow normal program execution to resume.

Let's do this with a simple example. At address hex 027A, we have the following two instructions: LDY #0:LDA #$20 .. followed by more program. During testing we discover that we have forgotten an important step – say, printing a carriage return character (hex 0D). There obviously isn't room to insert the missing code into the program; how do we handle it?

First, we look around for a patch area. In this case, we might find that there is free memory starting at address $0300. We know that we will want to insert a JMP $0300 instruction – three bytes long – into our code. Since our LDY and LDA instructions are two bytes each in length, we're going to clobber both of them (which means that we'll need to rebuild them both).

OK, at location $0300 we can code: LDA #$0D:JSR $FFD2 to print the carriage return. Now we must rebuild the butchered code with LDY #0:LDA #$20 and finally return to the continuation point with JMP $027E. The last step is to place the JMP $0300 instruction at $027A and activate the patch.

All this must be done in machine language, so hand assembly is necessary. It's not really hard.

Our coding at $0300 works out to A9 0D 20 D2 FF A0 00 A9 20 4C 7E 02; and at 027A we place the code 20 00 03. Note that there's a "left-over" byte at address 027D, but that doesn't matter. Now we can take another shot at our program and see if all the problems are corrected.

One more thing: you need to make careful records of your patches. As patches are written, the memory they occupy must be marked off so that you won't try to use those locations again. The patch itself must be written out carefully – you may need it during debugging. If you find a bug in a patch, it's better not to try to "patch the patch." Just write a brand new corrected patch program somewhere else.

Wrapping It Up

Patches are usually temporary activities during a debugging session. Testing takes place; a bug is found; a patch is written; testing resumes; more bugs, more patches, etc. Eventually, when the program behaves satisfactorily, you'll want to clean up and reassemble. The patches have done their job; they've allowed you to whip the program into running shape. Now you'll want to clean up, document, and so forth. The patches look ungainly; you'll want them out of there.

Occasionally, however, patches are left in place permanently. If a program has been released and users have come to depend on certain "entry points," it would be unwise to reassemble, which would move things around and cause problems.

Patch Points

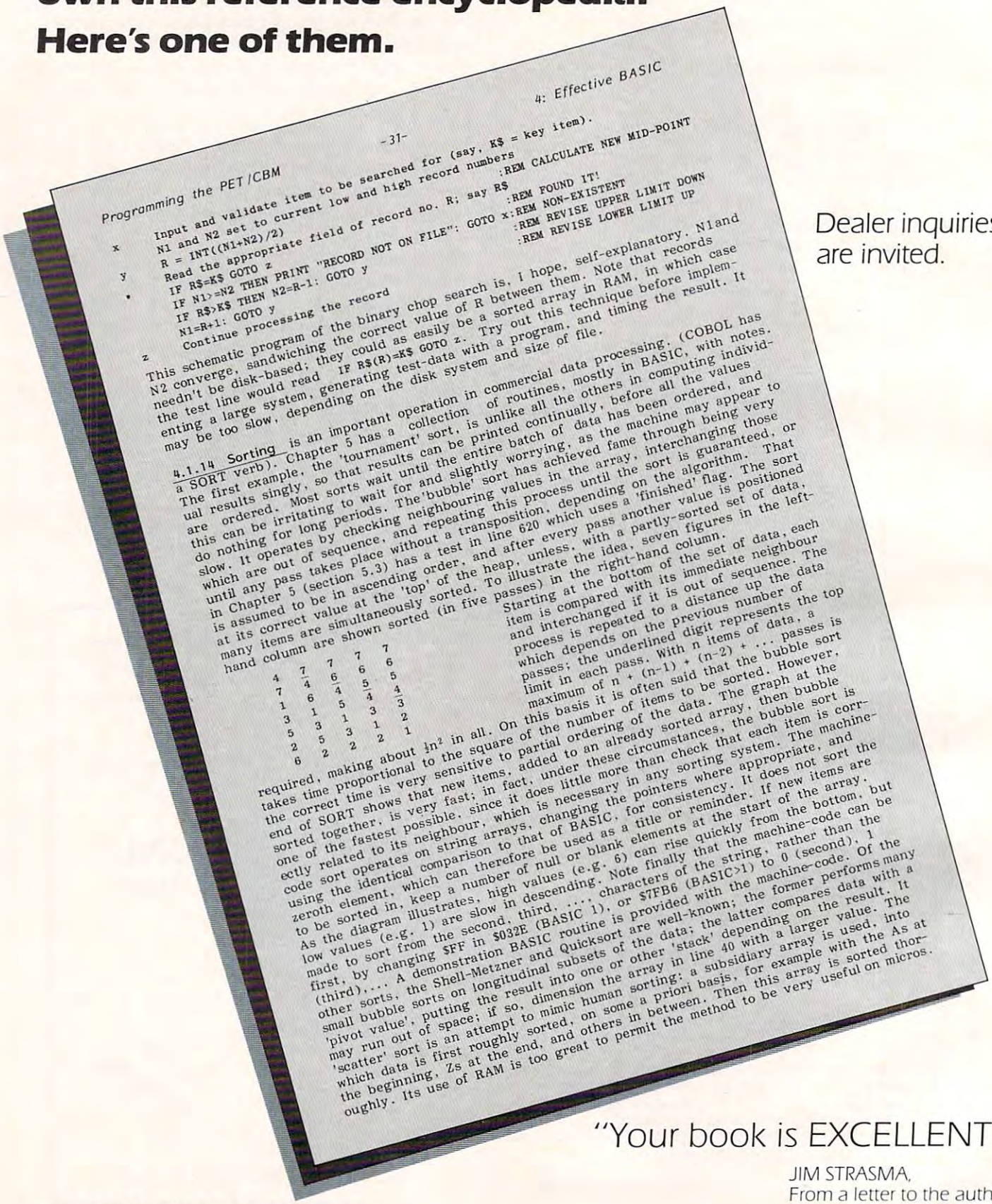
Certain places are easy to patch. If there's a JMP or JSR (or, for that matter, almost any three-byte instruction) at a convenient place in your code, you can quite easily slip in extra code with little dislocation.

Other code is quite difficult. The 6502 Branch instructions are relative, and will only reach 120-odd locations either way. If you tried to overwrite a Branch instruction, you might have troubles rebuilding it in your patch – it probably wouldn't reach.

Some programmers make provision for patches as they write code. Every once in a while, they throw in a group of three NOP instructions, which do nothing but provide space for hooking in a patch for correction or testing.

Most of us, however, forge ahead in the expectation that our coding will be perfect the first time around. Occasionally we're right. When we're wrong, we reach for a patch. ©

There are over 500 reasons to own this reference encyclopedia. Here's one of them.



Programming the PET/CBM -31- 4: Effective BASIC

```
x    Input and validate item to be searched for (say, K$ = key item).
     N1 and N2 set to current low and high record numbers
y    R = INT((N1+N2)/2)                          :REM CALCULATE NEW MID-POINT
     Read the appropriate field of record no. R; say R$
     IF R$=K$ GOTO z                             :REM FOUND IT!
     IF N1>=N2 THEN PRINT "RECORD NOT ON FILE": GOTO x:REM NON-EXISTENT
     IF R$>K$ THEN N2=R-1: GOTO y                :REM REVISE UPPER LIMIT DOWN
     N1=R+1: GOTO y                              :REM REVISE LOWER LIMIT UP
z    Continue processing the record
```

This schematic program of the binary chop search is, I hope, self-explanatory. N1and N2 converge, sandwiching the correct value of R between them. Note that records needn't be disk-based; they could as easily be a sorted array in RAM, in which case the test line would read IF R$(R)=K$ GOTO z. Try out this technique before implementing a large system, generating test-data with a program, and timing the result. It may be too slow, depending on the disk system and size of file.

4.1.14 Sorting is an important operation in commercial data processing. (COBOL has a SORT verb). Chapter 5 has a collection of routines, mostly in BASIC, with notes. The first example, the 'tournament' sort, is unlike all the others in computing individual results singly, so that results can be printed continually, before all the values are ordered. Most sorts wait until the entire batch of data has been ordered, and this can be irritating to wait for and slightly worrying, as the machine may appear to do nothing for long periods. The 'bubble' sort has achieved fame through being very slow. It operates by checking neighbouring values in the array, interchanging those which are out of sequence, and repeating this process until the sort is guaranteed, or until any pass takes place and no transposition, depending on the algorithm. That in Chapter 5 (section 5.3) has a test in line 620 which uses a 'finished' flag. The sort is assumed to be in ascending order, and after every pass another value is positioned at its correct value at the 'top' of the heap, unless, with a partly-sorted set of data, many items are simultaneously sorted. To illustrate the idea, seven figures in the left-hand column are shown sorted (in five passes) in the right-hand column.

```
4   7   7   7   7
7   4   6   6   6
1   6   4   5   5
3   1   5   4   4
5   3   1   3   3
2   5   3   1   2
6   2   2   2   1
```

Starting at the bottom of the set of data, each item is compared with its immediate neighbour and interchanged if it is out of sequence. The process is repeated to a distance up the data which depends on the previous number of passes; the underlined digit represents the top limit in each pass. With n items of data, a maximum of $n + (n-1) + (n-2) + \ldots$ passes is required, making about $\frac{1}{2}n^2$ in all. On this basis it is often said that the bubble sort takes time proportional to the square of the number of items to be sorted. However, the correct time is very sensitive to partial ordering of the data. The graph at the end of SORT shows that new items, added to an already sorted array, then bubble sorted together, is very fast; in fact, under these circumstances, the bubble sort is one of the fastest possible, since it does little more than check that each item is correctly related to its neighbour, which is necessary in any sorting system. The machine-code sort operates on string arrays, changing the pointers where appropriate, and using the identical comparison to that of BASIC, for consistency. It does not sort the zeroth element, which can therefore be used as a title or reminder. If new items are to be sorted in, keep a number of null or blank elements at the start of the array. As the diagram illustrates, high values (e.g. 6) can rise quickly from the bottom, but low values (e.g. 1) are slow in descending. Note finally that the machine-code can be made to sort from the second, third, ..., characters of the string, rather than the first, by changing $FF in $032E (BASIC 1), or $7FB6 (BASIC>1) to 0 (second), 1 (third),... A demonstration BASIC routine is provided with the machine-code. Of the other sorts, the Shell-Metzner and Quicksort are well-known; the former performs many small bubble sorts on longitudinal subsets of the data; the latter compares data with a 'pivot value', putting the result into one or other 'stack' depending on the result. It may run out of space; if so, dimension the array in line 40 with a larger value. The 'scatter' sort is an attempt to mimic human sorting: a subsidiary array is used, into which data is first roughly sorted, on some a priori basis, for example with the As at the beginning, Zs at the end, and others in between. Then this array is sorted thoroughly. Its use of RAM is too great to permit the method to be very useful on micros.

Dealer inquiries are invited.

"Your book is EXCELLENT!"

JIM STRASMA,
From a letter to the author.

A **COMPUTE! Books** Publication
In conjunction with Level Limited.

Programming The PET/CBM

by Raeto Collin West

The book described by Jim Butterfield as

"...unquestionably the most comprehensive and accurate reference I have seen to date..."

The Reference Encyclopedia for Commodore 2000, 3000, 4000, and 8000 series computers and peripherals.

Here's just a sample of reviewer and reader reaction:

From reviewers:

Educational Computing Review by Stephen Potts

"Of all the books I have read on the PET this book *Programming the PET/CBM* by Raeto West must rank as one of the most comprehensive and readable accounts on the PET that I have ever had the pleasure to see...

"If you wish to get more from your PET than arcade games and simple teaching programs then this book is a must for your bookshelf. It does not matter whether you run on BASIC 1, BASIC 2, or BASIC 4 since all routines are supplied with addresses and changes to make them run on any machines wherever possible...

"...this book, with its lucid explanations of the PET, its useful routines and programming hints, is an essential purchase."

IPUG Magazine Review (British PET User Group) by Ron Geere

"This publication represents over a year's intensive research ... and the resulting product is a valuable work of reference. A tremendous amount of useful information has been packed in this 500+ page work at which I was so over-awed that I did not know how to start this review at first...

"This book is a must for every CBM/PET user."

From readers:

"...a book the average to advanced user cannot afford *not* to possess..."

"My copy of your *'Programming the PET/CBM* has been in daily use for nearly a month and I am finding it totally addictive, suffering severe withdrawal symptoms whenever I try half-heartedly to move on to other reading matter. It is without doubt the best book on its subject available today..."

"I have recently acquired a copy of your book *Programming the PET/CBM* and must congratulate you on its concept and on packing in so much detail. It's so very much better than anything I have had up to now that it'll be my constant reference manual."

"I have received my copy of *Programming the PET/CBM* by Raeto West and I have recommended it to several of my students. This book is so valuable that I cannot now afford to be without it."

Published exclusively in North America by **COMPUTE! Books**. The book is an astonishing reference manual of useful information. Contents include this and much more:

1 Introduction and overview: Plan of the book, sources of information, features and chronology of CBM hardware.
2 BASIC and how it works: Storage of BASIC and its variables; tokens, pointers, syntax; optimising BASIC.
3 Program and system design: Capabilities of the equipment; charts, algorithms, space, timing.
4 Effective programming in BASIC: Seventeen examples, including subroutines, dates, DATA, INPUT, rounding.
5 Alphabetic reference to BASIC keywords: Full descriptions, with examples, of all keywords, with methods for adding additional commands not present in CBM BASIC, e.g. AUTO, DEL, OLD, POP, PRINT USING, SORT, VARPTR.
6 Disk drives: Descriptions of operation and workings of disk drives, with BASIC and machine-code examples; bugs.
7 Alphabetic reference to disk BASIC commands: BASIC 4 disk commands with examples and notes.
8 Other peripherals and hardware: Tape storage and handling; printers; modem; keyboard; EPROMs; reset switches.
9 Graphics and sound: Tables of CBM characters; CRT chip; animation, bar plots, 80 by 50 etc.; user-port sound.
10 The transition to machine-code: Introductory concepts; a BASIC monitor; use of MLM, Supermon, Extramon; easy examples.
11 More 6502 machine-code: 6502 hardware features; eighteen common problems in programming; debugging.
12 Alphabetic reference to 6502 opcodes: Examples, notes, and explanations on each opcode from ADC to TYA.
13 Using ROM routines: IRQ, NMI, RESET; the Kernel; examples – modifying LIST; ordinary and relocating loaders.
14 Effective 6502 programming: Assemblers; CHRGET and wedges; PIAs, VIA, IEEE; common mistakes.
15 Index to BASIC ROMs and RAM: Memory map of RAM and ROM, detailing and comparing BASICs 1, 2, and 4.
16 Mathematical programming: Precision; equations; statistics; simulation; finance; calculus; machine-code.
17 Programming in business and education: Examples, applications and pitfalls in business and education.
Appendices: 6502 reference charts; Supermon listings; ASCII; glossary

Plus many programs, diagrams and charts. Paperback, 504 pages. ISBN 0 942386 04 3. **$24.95.**

To Order

Programming The PET/CBM

Call

TOLL FREE 800-334-0868
In NC Call 919-275-9809

Or send coupon to

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

In England, order from Level Limited, P.O. Box 438, Hampstead, London, NW3 1BH. Price in England is £14.90, including P & P.

Please send _____ copy (copies) of **Programming The PET/CBM** at $24.95 each. (In the US and Canada, add $3.00 shipping and handling. Outside North America add $9.00 for air mail delivery, $3.00 for surface delivery).

All orders must be prepaid in US funds (money order, check, or charge).

☐ Payment Enclosed

Please charge my ☐ VISA ☐ MasterCard ☐ Am. Express

Account No. _____ Expires ____/____

Name _____

Address _____

City _____ State _____ Zip _____

Country _____

Allow 4-6 weeks for delivery. Foreign surface delivery allow 2-4 months.

Apple Manager:

An Alphanumeric Data Manager

Robert Jacques Beck
Minneapolis, MN

I began writing a data management program as part of a classroom assignment, but I finished it only because I had become obsessed with fitting the pieces of the puzzle together. I learned that the ideal data management system does everything under the sun and will never be invented. The data manager described in this article is designed primarily for string data, although numeric applications are possible.

It has two advantages. First, you get a listing. Second, it's written in BASIC and you get some explanation of how it works. If you find that it doesn't meet all your needs, and you don't want to modify it, you'll have some valuable knowledge if you go looking for another data manager. The program is written for the Apple – hence the name Apple Manager – but many ideas collected in it can be used elsewhere. The rest of this section is a somewhat theoretical discussion, so if you want to get into the particulars of the program just skip ahead.

Computers are great at keeping track of large masses of information. That's what data management is all about, so asking "Why do we need data managers?" is like asking "Why do we need computers?" But that is a kind of circular definition. Maybe we should ask, "What should a data manager program do?"

I like to think of data managers as two-way transportation systems between my diskettes and me or, more technically, between the storage device and the information source. To store data we must input it, but that's not enough to give us control over the contents of our data files: we might want to come back later and modify or delete something. Similarly, information retrieval is not just a matter of pulling the stuff out as fast as we can; we may be interested in one kind of information on Monday and another on Tuesday. You'll see flexibility come alive when you try the program.

From Aardvarks to Ziggurats

Since data is stored in files, a data manager is first and foremost a file manager. You can use the same data manager program to deal with files from Aardvarks to Ziggurats because each file will have the same general structure, even though individual components may vary. You may not think about it, but you will definitely take advantage of similarities in file structure when you write additions to a data manager or interface your files with other programs.

Just as books are made up of pages, files are made up of records. Records can be subdivided into fields, much like the sentences on a page. So we can define a record as a logical grouping of several individual data items. If each record in a file is identical (that is, if it has the same size and makeup), several records hypothetically placed adjacent to each other will look like a rectangle. A rectangular file structure is easy to program.

Another possibility is a hierarchical structure. Hierarchical files have records that are built from the same group – but not necessarily the same number – of components. Hierarchies occur naturally in many applications. Suppose you want to keep tabs on the books in your library and you want to cross-index them by one or more topics. One approach is a rectangular file with each record storing information about one book.

But there's a slight problem. You will need a field for author, another for title, and one additional field for each topic. Because you need to know how many fields to allot, you'll have to decide in advance on a reasonable maximum number of topics. On the other hand, a hierarchical file doesn't lock you into a fixed design. Imagine a hierarchy with author at the top. Titles are second in status, with each title being linked to its author's name. Topics are linked, in turn, to titles. In both cases, there is no set number of linkages.

Though this program is based on a rectangular organization, I just wanted to point out that there are alternative ways to set up data bases.

Module Structure

The more a program does, the more likely it is to grow to an unmanageable size. One way to cope with this is to break your program into chunks called modules. The main driver (lines 91-92) prints a menu and lets your choose one of the five modules: Files, Records, Reports, Select File, or Utilities. With the exception of Select File, all of the modules are multi-functional so the first thing you see when you enter them is another menu.

Menus allow you to move away from the main drive and into the tangled depths of the program, but how do you return? Whenever Apple manager requests input, if you type CONTROL O (for Out), plus a RETURN if necessary, you'll back up one level in the program hierarchy. This is a handy escape from any operation. It's the only way you

exit from functions that don't automatically return to menu. To switch from one module to another, type CONTROL O to back up to the main menu, then select the new module.

To conserve memory, I used a lot of subroutines and multiple statement lines. Program lines are numbered consecutively, instead of adding 10 to each line number as is usually done. There are so many GOSUBs and GOTOs that I saved about 300 bytes this way. Variable names are short and variables are used and reused whenever possible. Look at Table 1 to sort out some of the confusion.

Diskette Data Bases

Apple Manager assumes every diskette is an independent filing system. Each diskette has one title file, which is a sequential file containing the number of data files on the diskette and their names. When the program starts up it reads the names into an array (T$, line 89). Whenever a file name is input later on, it can be checked against this array to see if it is a valid file name for the diskette (lines 57-60). Data is kept in random access files. The data manager has to know how to relate to these files. Somebody has to tell it things like what size record to use, how many fields, what their names are, and so on. The easiest way to do it is to put the information into a file.

Rather than put it all in one file, I set up a separate file for the description of each data file. The description file is read into two arrays, one containing the names of the fields and the other containing the field lengths in characters (or bytes, since one character is stored per byte). You might want to look at the Atari Data Manager in **COMPUTE!** (November, 1981, #18) where the same concept is implemented somewhat differently. Fields are referred to as "items" by Apple Manager, so I'll use the two terms interchangeably. To summarize, each diskette has one title file, and for each data file there will be a description file.

Files

Most of this module is pretty easy to use once you get it running. The Catalog function is simple: it lists the title file array (lines 99-100). Describe File is similar in that it prints the description file arrays (lines 101-103). Create File is a bit more complicated – here's where new file structures are born. First type in the file name, then the number of items per record. All items are alphanumeric, in other words: strings. (Numbers are stored on diskette as strings anyway, one byte per number, because that's how Apple DOS formats diskette storage.)

After you finish entering a name and length for each item, you'll fall into the file editor. Record length is the sum of the field lengths plus the number of fields (because there is a return character after each field).

The file editor (lines 113-131) is basically a list editor. In BASIC, lists are virtually synonymous with arrays. It is the arrays holding the file description (L$ and L%) that get manipulated here. The edit menu uses abbreviations. (Replace the pound sign [#] with an integer.) I suggest you make a few simple typographical errors to see how the program responds. This is what the abbreviations mean:

S – saves/creates a data file and a description file.

R – review. Prints the file description.

A# – add # new items to the description (i.e., A3 = add three items).

D# – delete item # (i.e., D2 = delete item number 2).

I# – insert an item into position #.

N# – change the name of item number #.

L# – change the length of item number #.

Deleting and inserting items is done by shifting both description arrays; changing a name or length is done by entering a replacement for an element of one of the two arrays.

I once read somewhere that file maintenance consists of content changes and structural changes. The record editor described below takes care of content changes. *Evolvability*, the capacity to respond to changing needs, is accomplished through the file editor. If a check (line 114: is B>0?) shows that the file has data in it, you can still edit the file structure, although the program works a little harder. First, the original description is copied into some temporary arrays (line 114 again). When you're done monkeying around and you choose the Save option, the old and new descriptions are compared (lines 129-131). Next a scratch, or temporary, file is written to meet the new specifications (line 131).

Adding new items or changing an item name presents no problem. If an item is deleted it won't be rewritten; if an item is shortened, any instance of it that's too long gets truncated from the right. After the scratch file has been successfully completed, the old file is deleted and the scratch file is renamed. Apple Manager uses scratch files in a couple of other places, namely when sorting a file or deleting records. These routines also write a new, updated file before deleting the old file. You could run into a DISK FULL error if there weren't enough room. By the way, I've chosen the unlikely name of "A control D" for the scratch file's name, so it shouldn't interfere with any of your files.

The Other Choices

Perhaps you've asked yourself, "How does Apple Manager know when a file has data?" The method

is simple. Record zero, the first record in a random access file, stores the number of records. This number is updated whenever records are added or deleted (line 64). A newly created or emptied file (see below) is actually one record on the diskette with a 48 (the ASCII code for zero) in the first byte.

No data manager would be complete without the ability to get rid of unwanted files. Apple Manager deletes the data file and the corresponding description file and removes the file name from the title array (lines 132-134). The title array may change several times in one run. Rather than rewrite the file each time, a flag variable, F, is set. The title file on the diskette is updated (line 66) when you exit the Files module – if the flag is set (see lines 5 and 223). This is why the disk drive light may come on when you switch modules.

If you don't want to remove a file (just reuse it), then the Empty function at lines 135-136 is for you. This section deletes a data file but not its description file, and opens a new data file of the same name. The net effect is to empty a file of data while preserving the file structure. Copy creates a new data file by using an already existing description file.

A few words about limits and error handling are undoubtedly in order here. The dimension statement in line two defines the first three of these somewhat arbitrary limits, so they can be easily changed:

25	files per diskette.
50	fields per record.
1000	records per file.
115	bytes = maximum field length.
20	characters maximum in a file or item name.

The last two limits, as well as many errors, are avoided by checking input: line 32 (Is a number out of range?), line 35 (Has the return key been pressed without first typing something?) and line 36 (Is a string too long?), are examples. But what if you try to make the program do something illegal, such as read data that doesn't exist? ONERR is meant for just such cases, though it does have the drawback of stopping the Apple's excellent error messages.

Here's how I compromised. If there's an error in line 89 – e.g., if there's no title file because it's the first time you're using a diskette – you jump to line 90 where the error flag for the rest of the program is set. From here on in, unless your error is one of the DOS errors dealt with by lines 220 and 221, the POKE 216,0 at line 222 cancels ONERR. RESUME causes the error to recur so you get an Apple message. Control C (program interrupt; error code = 255) is handled differently; it still works, but without the expected BREAK message.

Using A Directory

Let's assume we've got an imaginary file defined and that data has been entered into it. Let's also assume we want to extract information about an author named Kilroy. One way to do it is to search the entire file until we find the Kilroy record. But disk access is slow and we are impatient, so let's use a directory to locate the record instead.

When a file is selected, Apple Manager opens the description file and reads it into arrays (line 67, called as a subroutine from line 94). The data file is also opened and the first field of every record is read into the array D$ (line 94). It's faster to search this directory array than to search the file. There is a one-to-one correspondence between array elements and records (Figure 6): if *Kilroy* is the seventh array element then record seven is the record we want.

What we have done is to define the first field – in this case *author* – as the record identifier. Record IDs are used for rapid access in Delete, Print, and Change (described below). A subroutine beginning at line 43 requests the ID and searches the directory. You don't have to enter the complete ID. For instance, repeatedly typing "K" will locate, in sequence, all authors whose name begins with K.

Assuming 48K of memory, there are about 17,000 free bytes for the directory. (The exact amount depends on how much is used up by the title array, the description array, and other string variables.) If the first item is a long one, it may not be possible to have 1000 records in the file without disabling the directory. The same memory problem may arise when you sort, since the D$ array holds the item being sorted.

You now have an outline of how the program works. We could step slowly and leisurely through the code, but I don't want to send the editors into apoplexy. The rest of this writeup is a guide to using the program. A good way to start is by creating a file or two. Then go to the Records module, enter some fictitious data, sort it, and edit it. Next try a report. Then go back to Files and change the file structure. Now generate another report to see how stored data has been affected.

Apple Manager makes a good, if rudimentary, stab at most data management functions. One omission is computed variables. This is not a short program, so I'll make copies for anyone who sends $3, a diskette, and a stamped, self-addressed mailer to: Robert Beck, 2101 21 Ave. S., #W15, MPLS, MN 55404. To those who are typing it in, I wish a steady hand and a steadier eye. Happy data

managing.

Records Module

ENTER (lines 147-150) – Initializes each item to an asterisk (*) – so missing data is not a problem – then goes to the record editor (lines 8-27), which has five options:

1) Retype – type in a new value.
2) Control O – exit to menu.
3) Control B – back up one field in the file.
4) Control F – forward to next record.
5) RETURN – the return key must be pressed after each of the above options. Pressing the return key alone does not affect the item displayed; it moves you to the next item in the record.

Record number appears in parentheses to the left of the item name. Use Control B to backspace through a file; use RETURN to move forward through a file.

DELETE (lines 139-146) – marks records you choose to delete by placing a Control E in the first byte. When you exit this option via Control O, Apple Manager rewrites the file without marked records.

CHANGE (lines 151-158) – allows a "window" in each record to be set by selecting a starting and an ending item number. Once a starting record in the file is chosen, the record editor, which works as previously described, is called.

PRINT (lines 159-160) – prints one record at a time. This is the fastest way to retrieve a record (Figure 7).

SORT (lines 161-165) – reads the sort key (item to sort by) into the array D$, then sorts, in either ascending or descending order, by the bubble method. It's really a tag sort because the record numbers (in S%) are also sorted. Once record numbers are properly ordered, the file is rewritten. The sort is alphanumeric, so "17" is placed before "7" and after "07".

Report Module

1. Retrieve (lines 169-171, 190-198) – formats the report in tabular form if the printout from one record fits on one line (Figure 8), otherwise prints one item per line. A variable number of fields can be retrieved, and in any order.

2. With Sums – same report as Retrieve, with the addition (pun intended) that a sum for each item is printed.

3. Frequencies (lines 199-204) – counts the number of times each value of an item occurs. First, sort the file by that item.

4. Case Selection (lines 172-188) – a technique that lets you retrieve information by its characteristics – you can pick out a subset of the file. All you have to do is input selection criteria in the form of minimum and maximum item values. The values are stored in arrays (M$ and N$) and may be ORed together in groups of five or less. Up to five of these groups can be ANDed together:

(a1 OR b1 OR c1 OR d1 OR e1) AND ... AND (a5 OR b5 OR c5 OR d5 OR e5).

Each a1, b1, c1, etc. is of the form MIN(A)<=A=<MAX(A).

Got that? Well, here's how to work it:

1) Select an item by typing its number.

2) Select a range for that item by typing a minimum and a maximum. Pressing the return key without typing anything sets a minimum to the null string or a maximum to CHR$(95).

3) Terminate a series of ORs and go on to the next AND by typing a slash (/).

4) Terminate the whole thing at any point, including the very beginning, by typing a period.

Note: Each record is checked against the criteria stored in arrays (lines 76-79) and ignored if it doesn't meet them (i.e., if G=5). For an alternative, and very interesting, method of introducing changeable functions into your program, see "Algebra String – a Self-altering Program" in **COMPUTE!** (September, 1981, #16).

Utilities Module (lines 212-218)

Upload – Each record's fields are laid down sequentially within the record. There will be null bytes at the end of the record if any item is shorter than its defined byte length. Upload removes all null bytes from a file by adding blanks where needed. The name "Upload" comes from the fact that some mainframe computers interpret null bytes as end-of-record marks; to send diskette files to them null bytes must be removed.

Download – Does the opposite of Upload.

Drive Select – Use to switch from one disk drive to another. Each diskette remains an independent data base.

```
1  REM    APPLE MANAGER    **  ROBERT JAC
     QUES BECK  **
2 D$ =  CHR$ (4): FOR I = 768 TO 777: RE
     AD J: POKE I,J: NEXT : DIM L$(50),
     F$(4),T$(25),L%(50),D$(1000),R$(50)
     ,S%(1000),S(50),M$(25),N$(25): FOR
     I = 0 TO 4: READ F$(I): NEXT : GOTO
     88
3  VTAB 23: PRINT "PRESS RETURN TO CONT
     INUE";: CALL  - 958
4  POKE  - 16368,0: INPUT "->";Z$:H =
     RIGHT$ (Z$,1) =  CHR$ (15)
5  IF E = 7 AND F = 1 AND H THEN  GOSUB
     66
6  IF H THEN  POP : CALL 768: ON E GOTO
```

```
     96,104,166,212,137,152,91,115,144,
     156,172
7 Z = VAL (Z$): RETURN
8  FOR Y = 0 TO K
9  VTAB 19: HTAB 1: PRINT " (R"I")" SPC
     ( 2)Y". "L$(Y)":": PRINT : PRINT R
     $(Y)
10  HTAB 1: VTAB 21: INPUT "";Z$: IF Z$
     = "" THEN 15
11  IF Z$ =  CHR$ (2) THEN 23
12  IF Z$ =  CHR$ (6) OR Z$ =  CHR$ (15
     ) THEN X =  ASC (Z$): GOTO 18
13 L = L%(Y): VTAB 17: GOSUB 36: IF D T
    HEN 10
14 R$(Y) = Z$: VTAB 18: CALL  - 868
15  GOSUB 27: NEXT Y
16 Q = I: GOSUB 48:D$(I) = R$(1): IF X
    = 15 THEN  GOSUB 64: CALL 768: GOTO
    137
17  RETURN
18  IF V = 6 THEN 16
19 H = Y = 1 AND I = B + 1 AND R$(1) =
    "*": IF H AND X = 6 THEN 10
20  IF H AND X = 15 THEN  GOSUB 64: CALL
    768: GOTO 137
21  IF X = 15 THEN B = B + 1
22  GOSUB 27: GOTO 16
23  IF I = 1 AND Y < 2 THEN  VTAB 21: PR
    INT R$(Y): GOTO 10
24  IF Y > 1 THEN  GOSUB 27:Y = Y - 1:
    GOTO 9
25 Q = I: GOSUB 48:D$(I) = R$(1):I = I -
  - 1: IF V = 5 THEN  GOSUB 27:Y = K:
    GOSUB 47: GOTO 9
26  VTAB 6: CALL  - 958: PRINT "* ID= "D
    $(I): POKE 34, PEEK (37) + 1:Y = K:
     GOSUB 47: GOTO 9
27  HTAB 1: VTAB 21: PRINT R$(Y): FOR E
     = 1 TO  LEN (R$(Y)) / 40 + 4: CALL
      - 912: NEXT : RETURN
28  PRINT "** WHICH?";: POKE  - 16368,0:
     GET Z$: IF Z$ < >  CHR$ (15) THEN
     Z =  VAL (Z$): RETURN
29 H = 1: GOTO 5
30 C =A
31  GOSUB 4
32  IF Z < 1 OR Z > C OR  INT (Z) < > Z
     OR ( VAL ( RIGHT$ (Z$,1)) =0 AND
     RIGHT$ (Z$,1) < > "0") THEN PRINT
      "TYPE AN INTEGER FROM 1 TO "C;:W
      = 1: GOTO 31
33  RETURN
34  GOSUB 4
35 D = 0: IF Z$ = "" THEN  PRINT : PRINT
     "TYPE SOMETHING BEFORE PRESSING
     RETURN!":D = 1: RETURN
36 D = 0: IF  LEN (Z$) > L THEN  PRINT :
     PRINT "THAT IS TOO LONG, TRY AG
     AIN":D =1
37  RETURN
38 I =  VAL ( MID$ (Z$,2)): IF (I > A)
     THEN PRINT "NUMBER OUT OF RANGE-
      START OVER!": POP : GOTO 116
39 I =  VAL ( MID$ (Z$,2)): IF I = 0 OR
     ( VAL ( RIGHT$ (Z$,1)) = 0 AND R
     IGHT$ (Z$,1) < > "0") THEN PRINT
     .  "TYPO, TRY AGAIN!": POP : GOTO
     116
40  RETURN
41  PRINT : PRINT "NAME OF ITEM "I"?";:
     GOSUB 34:L$(I) = Z$: IF D = 1 TH
     EN 41
42  PRINT "ITEM LENGTH?";:C = 115: GOSUB
     31:L%(I) = Z:R = + Z + 1: RETURN
43  PRINT :L = 115: PRINT "RECORD ID?";:
```

Table : Partial list of variables.

Name	Description
A$	File name.
D$	Control D (DOS commands).
D$(1000)	Directory and sort array.
F$(4)	Option titles (Records).
L$(50)	Item (field) names.
L%(50)	Item lengths.
M$(25)	Minima of criteria (Report).
N$(25)	Maxima of criteria (Report).
R$	Current value of item being counted (frequencies).
R$(50)	Fields of current record.
S(50)	Item sums (Report).
S%(1000)	Sort array (holds record numbers).
S%(1-50)	Item numbers being reported.
S%(200-250)	Criteria item numbers (Report).
S%(400-450)	Spaces allotted to an item in report printout.
Z$	Input.
A	Number of fields per record.
B	Number of records.
C	Maximum permissible value. Frequency count of an item (frequencies).
D	Input error flag. Number of lines on screen.
E	Module flag.
F	Title file update flag.
G	Option flag (Files). G<>5 if record meets criteria (Report). Total sum (Report).
H	Control 0 flag.
I,J,K,Q	Temporary indices.
K	Ending item number (edit window).
L	String or item length.
N	Number of files.
O	Starting item number (edit window). Number of criteria (Report).
P	P=1 if file name is in title file. Number of dashes to print.
Q	Record number. Larger of item length and length of item name (Line 74).
R	Record length.
S	Option selected (Report).
V	Drive number. Number of printer columns (Report).
W	Slot number.
X	Number of spaces needed for tabular report. First record to edit (Records).
Y	Number of items being reported. Item number being counted (frequencies). Error code.
Z	Numeric value of input.

```
      GOSUB 34:U = I:J = LEN (Z$): IF
      D = 1 THEN 43
44  IF U = B THEN U = 0
45  FOR I'= U + 1 TO B: IF  LEFT$ (D$(I)
      ,J) < > Z$ THEN  NEXT : IF U > 0
      THEN U = 0: GOTO 45
46  IF I > B THEN  PRINT : PRINT "ID NOT
      FOUND, TRY ANOTHER!": GOTO 43
47  PRINT D$"READ"A$",R"I: FOR J = 1 TO
      A: INPUT R$(J): NEXT : PRINT D$:
      RETURN
48  PRINT D$"WRITE"A$",R"Q: FOR J = 1 TO
      A: PRINT R$(J): NEXT : PRINT D$:
      RETURN
49  PRINT D$"WRITEA"D$",R"Q: FOR J = 1 T
      O  A: PRINT R$(J): NEXT : PRINT
      D$: RETURN
50  IF W = 0 AND D / 15 < >  INT (D / 15
      ) THEN  PRINT : PRINT : GOSUB 3:
      VTAB 23: HTAB 1: CALL  - 868
51  RETURN
52  PRINT
53 D = D +1: IF W = 0 AND D / 15 =  INT
      (D / 15) THEN  PRINT : PRINT : P
      RINT : GOSUB 3: VTAB 21: HTAB 1:
      CALL  - 958
54  RETURN
55  PRINT L$(Q)": "R$(Q): FOR J = 1 TO
      INT (( LEN (L$D(Q) + R$(Q))) / 4
      0) + 1: GOSUB 53: NEXT :RETURN
56 M$ = D$(J):D$(J) = D$(J - 1):D$(J - 1
      ) = M$:K = S%(J):S%(J) = S%(J -
      1):S%(J - 1) = K:K = 1: RETURN
57  PRINT : PRINT "FILE NAME?";:L = 20:
      GOSUB 34: IF D = 1 THEN 57
58 A$ = Z$: FOR I =1 TO N: IF Z$ = T$(I)
      THEN P = 1: RETURN
59  NEXT :P = 0: IF G < 2 THEN  PRINT :
      PRINT Z$" ISN'T IN THE TITLE": P
      RINT "FILE, TRY AGAIN.": GOSUB 3
60  RETURN
61  GOSUB 63: IF B >  0 THEN  PRINT : P
      RINT "FILE "Z$" HAS DATA -": IF
      P < 3 THEN  POP : GOSUB 3: GOTO
      96
62  RETURN
63  GOSUB 65: PRINT D$"READ"A$",R0": IN
      PUT B : PRINT D$: RETURN
64  GOSUB 65: PRINT D$"WRITE"A$",R0": P
      RINT B: PRINT D$: RETURN
65  PRINT : PRINT D$"OPEN"A$",L"R: RETU
      RN
66  PRINT : PRINT D$"OPEN TITLE FILE":
      PRINT D$"WRITE TITLE FILE": PRI
      NT N: FOR I = 1 TO N: PRINT T$(
      I): NEXT : PRINT D$"CLOSETITLE
      FILE":F =0: RETURN
67  PRINT D$"OPENDES "A$: PRINT D$"READ
      DES "A$: INPUT A,R: FOR I = 1 T
      O A: INPUT L$(I),L%(I): NEXT :
      PRINT D$"CLOSE": RETURN
68  HOME : PRINT "FILE NAME: "A$: PRINT
      : PRINT A" ITEMS PER RECORD": P
      RINT : PRINT "RECORD LENGTH: "R
69  PRINT : PRINT "#" SPC( 6)"ITEM" SPC
      ( 20)"LENGTH":P = 40: GOSUB 82:
      PRINT : RETURN
70  GOSUB 68: FOR I = 1 TOA: PRINT I"."
      SPC( 6 -  LEN ( STR$ (I)))L$(I)
      SPC( 26 -  LEN (L$(I)))L%(I): P
      RINT : IF (I - 6) / 8 =  INT ((
      I - 6) / 8) AND (I < > A) AND W
      = 0 THEN  GOSUB 3: HOME : GOSUB
      69
71  NEXT : PRINT D$"PR#0": RETURN
72  FOR W = U TO O * 5:M$(W) =  CHRS$ (
      95):S%(200 + W) = 0: NEXT : RET
      URN
73  VTAB 23 - C: CALL  - 958: PRINT : R
      ETURN
74 J = S%(I):Q = L%(J): IF  LEN (L$(J))
      > L%(J) THEN Q =  LEN (L$(J))
75 X = Q + X + 2:S%(400 + I) = Q: RETURN
76 G = 0: GOSUB 47: IF O = 0 THEN  RETU
      RN
77  FOR K = 1 TO O:G = 0: FOR J = 1 TO
      5:U = (K - 1) * 5 + J:Q = S%(20
      0 + U): IF R$(Q) < M$(U) OR R$(
      Q) > N$(U) THEN G = G + 1
78  NEXT : IF G = 5 THEN  RETURN
79  NEXT : RETURN
80  PRINT : PRINT "SEND PRINTOUT TO SLO
      T NUMBER?":P = 12: GOSUB 82: PR
      INT "DEFAULT = TV": GOSUB 82: V
      TAB  PEEK (37) - 3: HTAB 30 : G
      OSUB 4:W = Z: CALL  - 958: RETU
      RN
81  PRINT D$"PR#"W: RETURN
82  FOR J = 1 TO P: PRINT "-";: NEXT :
      PRINT : RETURN
83  HOME : VTAB 2: PRINT  TAB( 13)F$(Z
      - 1)" RECORDS"
84  VTAB 4: PRINT "FILE NAME: "A$: POKE
      34,5: VTAB 7: RETURN
85  IF B = 0 THEN  RETURN
86  FOR J = 1 TO V: IF L$(I) < > D$(J)
      THEN  NEXT : RETURN
87 D$(J) = Z$: RETURN
88 V = 1: ONERR  GOTO 90
89  PRINT D$"OPENTITLE FILE,D"V: PRINT
      D$"READTITLE FILE": INPUT N: IF
      N > 0 THEN  FOR I = 1 TO N: INP
      UT T$(I): NEXT
90  PRINT D$"CLOSE": ONERR  GOTO 219
91  TEXT : HOME : VTAB 2: PRINT  SPC( 1
      0)   APPLE MANAGER   ": VTAB 7:
      HTAB 3: PRINT "1  FILES" SPC( 1
      2)"4  REPORTS": PRINT : HTAB 3:
      PRINT "2  SELECT FILE" SPC( 6)"
      5 UTILITIES": PRINT : HTAB 3: P
      RINT "3  RECORDS" SPC( 10)"6 QU
      IT"
92 E = 7: VTAB 20: GOSUB 28: ON Z GOTO
      96,93,137,166,212,223: PRINT :
      PRINT "YOU CAN'T CHOOSE THAT! T
      RY AGAIN": GOTO 92

93 O = 1:E = 7: HOME : VTAB 4:G = 0: PR
      INT  TAB( 11)"-- SELECT A FILE
      --": VTAB 6: GOSUB 57: IF P = 0
      THEN 93
94  GOSUB 67: GOSUB 63: IF B > 0 THEN F
      OR I = 1 TO B: PRINT D$"READ"A$
      ",R"I: INPUT D$(I): NEXT : PRIN
```

```
        T D$
95  GOTO 91
96 O = 2:L = 20:G = 0:P = 0: HOME : VTA
        B 2:E = 7: PRINT  SPC( 13)" <<
        FILES  >>": VTAB 6: PRINT : PRI
        NT "1  CATALOG" SPC( 10)"5 EDIT
        DESCRIPTION": PRINT : PRINT "2
        DESCRIBE FILE" SPC( 4)"6  EMPTY
        FILE": PRINT : PRINT "3  CREATE
        FILE" SPC( 6)"7  COPY DESCRIPTI
        ON"
97  PRINT : PRINT "4  DELETE FILE" SPC(
        6)"8  QUIT"
98  VTAB 22: GOSUB 28:E = 1: ON Z GOTO
        99,101,104,132,113,135,106,223:
        FLASH : PRINT : PRINT "TYPE A N
        UMBER FROM 1 TO 8": NORMAL : GO
        TO 98
99  HOME : PRINT  TAB( 9)"CATALOG OF DA
        TA FILES": PRINT :D = 0:W = 0:
        IF N = 0 THEN  GOSUB 3: GOTO 96
100  FOR I = 1 TO N: PRINT T$(I): GOSUB
        53: NEXT : GOSUB 50: GOTO 96
101  HOME : PRINT  TAB( 9)"** DESCRIBE
        FILE **": PRINT : GOSUB 57: IF
        P = 0 THEN 101
102  GOSUB 67: GOSUB 63: GOSUB 80: GOSU
        B 81: GOSUB 70: IF W = 0 THEN
        GOSUB 3
103  GOTO 96
104 E = 1: HOME : PRINT  TAB( 7)"** CRE
        ATE A NEW FILE **": PRINT :G =
        2: GOSUB 57: IF P = 1 THEN  GOS
        UB 67: GOSUB 61
105  PRINT "NUMBER OF ITEMS PER RECORD?
        ";:C = 50: GOSUB 31:A = Z:R = 0
        : FOR I = 1 TO A: GOSUB 41: NEX
        T :B = 0: GOTO 115
106  HOME : PRINT  TAB( 18)"COPY": PRIN
        T : PRINT "OLD";: GOSUB 57: IF
        P = 0 THEN 106
107  GOSUB 67: PRINT :G = 7:B = 0: PRIN
        T "NEW";: GOSUB 57: IF P = 1 TH
        EN  GOSUB 61
108  IF B > 0 THEN 129
109  VTAB 24: FLASH : PRINT "CREATING F
        ILE "A$: NORMAL
110  PRINT D$"OPENDES "A$: PRINT D$"DEL
        ETEDES "A$: PRINT D$"OPENDES "A
        $: PRINT D$"WRITEDES "A$: PRINT

        A: PRINT R: FOR I = 1 TO A : PR
        INT L$(I): PRINT L%(I): NEXT :
        PRINT D$"CLOSE"
111  FOR I = 1 TO N: IF A$ < > T$(I) TH
        EN  NEXT :N = N + 1:T$(N) = A$
        :F = 1
112  GOSUB 64: GOTO 96
113 G = 1: HOME : PRINT  TAB( 10)"## ED
        IT DESCRIPTION ##": GOSUB 57: I
        F P = 0 THEN 113
114  GOSUB 67:P = 3: GOSUB 61: PRINT :
        IF B > 0 THEN FOR I = 1 TO A:D$
        (I) = L$(I):S%(I) = L%(I): NEXT
115 R$ = "SRADINL":P = 39: GOSUB 82: PR
        INT  TAB( 3)"SAVE, REVIEW, ADD,
        DELETE, INSERT,": PRINT  TAB( 3
        )"CHANGE NAME, OR CHANGE LENGTH
        ....."
116 E=G: PRINT "(S, R, A#, D#, I#, N#,
        OR L#)";: GOSUB 4:E = 8:I = 1
117  IF  LEFT$ (Z$,1) =  MID$ (R$,I,1)
        OR I > 7 THEN 119
118 I = I + 1: GOTO 117
119  ON I GOTO 108,120,121,123,126,127,
        128: FLASH : VTAB 23: PRINT "HE
        Y!";: NORMAL : GOTO 116
120  GOSUB 70: GOTO 116
121  GOSUB 39: PRINT : PRINT "ADD "I" I
        TEMS:":Y = A + I: IF Y > 50 THE
        N  PRINT "YOU CAN'T HAVE MORE T
        HAN 50 ITEMS!": GOTO 116
122  FOR I = A + 1 TO Y: GOSUB 41:A = I
        : NEXT : GOTO 116
123  GOSUB 38: PRINT : PRINT : PRINT "D
        ELETE: "L$(I) TAB( 28)"LENGTH:
        "L%(I):R =R - L%(I) - 1: IF (I
        = A) THEN 125
124  FOR J = I TO A - 1:L$(J) = L$(J +
        1):L%(J) = L%(J + 1): NEXT
125 A = A - 1: GOTO 116
126  GOSUB 38:M$ = L$(I):Y = L%(I): GOS
        UB 41:A = A + 1: FOR J = A TO I
        + 2 STEP  -1:L$(J) = L$(J - 1):
        L%(J) = L%(J - 1): NEXT : L$(I
        + 1) = M$:L%(I + 1) = Y: GOTO 1
        16
127  GOSUB 38: PRINT "OLD NAME: "L$(I):
        PRINT : PRINT "NEW NAME?";: GOS
        UB 34: GOSUB 85:L$(I) = Z$: GOT
        O 116
128  GOSUB 38: PRINT L$(I)"  >>>  LENGT
        H IS "L%(I): PRINT : PRINT "NEW
        LENGTH?";:C = 115: GOSUB 31:R =
        R - L%(I) + Z:L%(I) = Z: GOTO 1
        16
129  HOME : FLASH : PRINT "REWRITING":
        NORMAL : FOR I = 1 TO A:S%(100
        + I) = 0: FOR J = 1 TO V: IF D$
        (J) = L$(I) THEN S%(100 + I) =
        J: GOTO 131
130  NEXT J
131  NEXT I:D$(0) = "*": PRINT D$"OPENA
        "D$",L"R: FOR Q = 1 TO B: PRINT
        D$"READ"A$",R"Q: FOR J = 1 TO V
        : INPUT D$(J): NEXT : FOR J = 1
        TO A:R$(J) =  LEFT$ (D$(S%(100
        + J)),L%(J)): NEXT : GOSUB 49:
        NEXT : PRINT D$"DELETE"A$: PRIN
        T D$"RENAMEA"D$","A$: GOTO 110
132  HOME : PRINT : PRINT  TAB( 9);: FL
        ASH : PRINT "###";: NORMAL : PR
        INT "  DELETE FILE  ";: FLASH :
        PRINT "###": NORMAL : VTAB 8: G
        OSUB 57: IF P = 0 THEN 132
133  IF I < > N THEN  FOR J = I TO N -
        1:T$(J) = T$(J + 1): NEXT
134  PRINT D$"OPEN"A$: PRINT D$"DELETE"
        A$: PRINT D$"OPENDES "A$: PRINT
        D$"DELETEDES "A$:F = 1:N = N -
        1: GOTO 96
135  HOME : PRINT : PRINT  TAB( 10)"EMP
        TY A FILE": GOSUB 57: IF P = 0
```

```
      THEN 135
136  PRINT D$"OPEN"A$: PRINT D$"DELETE"
      A$: GOSUB 67:B = 0: GOSUB 64: G
      OTO 96
137  PRINT : PRINT D$"PR#0": TEXT : HOM
      E : VTAB 2:E = 7: PRINT  SPC( 1
      1)"<<<  RECORDS  >>>": VTAB 8:
      PRINT "1 ENTER RECORDS" SPC( 6)
      "4 PRINT RECORDS": PRINT : PRIN
      T "2 DELETE RECORDS" SPC( 5)"5
      SORT": PRINT : PRINT "3 CHANGE
      RECORDS" SPC( 5)"6  QUIT"
138  VTAB 22: GOSUB 28: PRINT :E =5: ON
      Z GOTO 147,139,151,159,161,223:
      FLASH : PRINT : "TYPE A NUMBER
      FROM 1 TO 6!": NORMAL : GOTO 138
139 E = 9: GOSUB 83:I = 0
140  GOSUB 43: PRINT : PRINT "DELETED:
      ";: FOR J = 1 TO A: PRINT R$(J)
      " ";: NEXT : PRINT : PRINT "IS
      THIS WHAT YOU WANT DELETED?"
141  PRINT "(Y OR N)";: GOSUB 4: IF Z$
      = "Y" THEN  PRINT D$"WRITE"A$",
      R"I: PRINT  CHR$ (5): PRINT D$:
      GOTO 140
142  IF Z$ = "N" THEN 140
143  FLASH : PRINT "HEY!";: NORMAL : GO
      TO 141
144  HOME : HTAB 12: FLASH : PRINT "REW
      RITING": NORMAL : PRINT D$"OPEN
      A"D$",L"R:K = B:B = 0: FOR I =
      1 TO K: GOSUB 47: IF  LEFT$ (R$
      (1),1) < >  CHR$ (5) THEN B = B
      + 1:Q = B: GOSUB 49:D$(B) = R$(
      1)
145  NEXT
146  PRINT D$"DELETE"A$: PRINT D$"RENAM
      EA"D$","A$: GOSUB 64: GOTO 137
147  GOSUB 83
148 V = 5:0 = 1:K = A: FOR I = 1 TO A:R
      $(I) = "*": NEXT : I = B + 1
149  GOSUB 8: IF I < B + 1 THEN I = I +
      1: GOSUB 47: GOTO 149
150 B = B + 1: GOTO 148
151  GOSUB 83
152 E = 5: PRINT : PRINT "STARTING ITEM
      NUMBER? (DEFAULT= 2)";: GOSUB 4
      :E = 6: IF Z$ = "" THEN O = 2:
      GOTO 154
153 C = A: GOSUB 32:O = Z
154  PRINT - PRINT "ENDING ITEM NUMBER?
      ": PRINT "(DEFAULT= LAST ONE)";
      : GOSUB 4: IF Z$ = "" THEN K =
      A: GOTO 156
155 K = Z:C = A: GOSUB 32: IF K <
      O THEN PRINT : PRINT "THE LAST
      ITEM NUMBER MUST BE AT LEAST "O
      : GOTO 154
156 E = 6: PRINT : PRINT "STARTING ID?
      (DEFAULT= FIRST ONE)";: GOSUB 4
      :E = 10: IF Z$ = "" THEN X = 1:
      GOTO 158
157 J = LEN (Z$): FOR X = 1 TO B: IF  L
      EFT$ (D$(X),J) < > Z$ THEN  NEX
      T : PRINT : PRINT "ID NOT FOUND
      , TRY ANOTHER!": GOTO 156
158 V = 6: FOR I = X TO B: GOSUB 47: VT
      AB 6: CALL  - 958: PRINT "*ID=
      "D$(I): POKE 34, PEEK (37) + 1:
      GOSUB 8: NEXT I: GOSUB 64: GOTO
      137
159  GOSUB 83: GOSUB 80:I = 0
160  PRINT D$"PR#0": GOSUB 43: GOSUB 81
      : PRINT : PRINT :Z =0:D =0: FOR
      Q = 1 TO A: GOSUB 55: NEXT : GO
      TO 160
161  GOSUB 83: PRINT "NUMBER OF ITEM TO
      SORT BY?";: GOSUB 30: PRINT : P
      RINT "I WILL SORT BY "L$(Z): PR
      INT :V = Z - 1: PRINT : PRINT "
      ASCENDING (1)": PRINT "OR DESCE
      NDING (2) ORDER?";:C = 2: GOSUB
      31
162  HOME : FLASH : PRINT "SORTING": NO
      RMAL : FOR I = 1 TO B:S%(I) = I
      : PRINT D$"READ"A$",R"I: PRINT
      D$"POSITION"A$",R"V: PRINT D$"R
      EAD"A$: INPUT D$(I): NEXT : FOR
      I = B TO 2 STEP  - 1:K = 0: FOR
      J = 2 TO I: IF Z = 1 AND D$(J -
      1) > D$(J) THEN  GOSUB 56
163  IF Z = 2 AND D$(J) > D$(J - 1) THE
      N  GOSUB 56
164  NEXT : IF K THEN  NEXT
165  PRINT D$"OPENA"D$",L"R: FOR Q = 1
      TO B:I = S%(Q): GOSUB 47: GOSUB
      49:D$(Q) = R$(1): NEXT : GOTO 1
      46
166  PRINT : PRINT D$"PR#0": TEXT :E =
      7: HOME : HTAB 15: PRINT "+ REP
      ORTS +": GOSUB 84: PRINT : HTAB
      9: PRINT "1  RETRIEVE": PRINT :
      HTAB 9: PRINT "2  WITH SUMS": P
      RINT : HTAB 9: PRINT "3  FREQUE
      NCIES": PRINT : HTAB 9: PRINT "
      4 QUIT"
167  VTAB 21: GOSUB 28:C = 4: PRINT : G
      OSUB 32:S = Z: IF Z = 4 THEN 22
      3
168  HOME :E = 3: IF Z = 3 THEN 199
169 X = 0: PRINT "REPORT TITLE?";: GOSU
      B 4:M$ =Z$: PRINT "HOW MANY ITE
      MS TO PRINT?": PRINT "(DEFAULT
      = ALL)";: GOSUB 4: IF Z$ < > ""
      THEN 171
170 Y = A: FOR I = 1 TO Y:S%(I) = I:S(I
      ) = 0: GOSUB 74: NEXT : GOTO 172
171 C = A: GOSUB 32:Y = Z: PRINT "TYPE I
      N ITEM NUMBERS ONE AT A TIME.":
      PRINT : FOR I = 1 TO Y: CALL  -
      868: PRINT  SPC( 16)"<-";: HTAB
      13:F =  PEEK (37): GOSUB 30: VT
      AB F + 1: HTAB 1: CALL  - 868:
      HTAB 7: PRINT L$(Z):S(I) = 0:S%
      (I) = Z: GOSUB 74: NEXT
172 P = 40: GOSUB 82: HTAB 13: PRINT "S
      ELECT CASES":E = 3: PRINT : PRI
      NT "TYPE AN ITEM NUMBER,THEN IT
      S MINIMUM ANDMAXIMUM VALUES (IN
      THAT ORDER!). PRESS  RETURN AFT
      ER EACH  ONE.": PRINT : PRINT "
      TYPE A SLASH (/) TO SKIP THE 'O
```

```
       R'."
173  PRINT "TO IGNORE A MINIMUM OR A MA
       XIMUM, MERELYPRESS RETURN.": PR
       INT : PRINT "TO FINISH, TYPE A
       PERIOD (.), THEN PRESS  RETURN.
       ":E = 3:O = 0: PRINT
174 O = O + 1: PRINT : FOR K = 1 TO 5:U
       = (O - 1) * 5 + K: PRINT O"." S
       PC( 3)"ITEM NUMBER?";: GOSUB 4:
       E = 11: IF Z$ = "." THEN C = 2:
       ON K GOTO 186: GOSUB 73: GOTO 1
       88
175  IF Z$ = "/" AND K > 1 THEN C = 3:
       GOSUB 73: GOSUB 72: GOTO 184
176 C = A:W = 0: GOSUB 32:S%(200 + U) =
       Z:L = L%(Z): VTAB 23 - W: HTAB
       5: CALL  - 958: PRINT "** "L$(Z)
177  PRINT  TAB( 7)"MINIMUM?";: GOSUB 4
       : GOSUB 36: IF D = 1 THEN  PRIN
       T : GOTO 177
178 M$ (U) = Z$: IF Z$ = "." THEN C = 3
       : ON K GOTO 186: GOSUB 73: GOTO
       188
179  PRINT  TAB( 7)"MAXIMUM?";: GOSUB 4
       : GOSUB 36: IF D = 1 THEN PRINT
       : GOTO 179
180 N$(U) = Z$: IF Z$ = "" OR Z$ = "."
       THEN N$(U) =  CHR$ (95)
181  IF Z$ = "." THEN U = U + 1: ON K G
       OTO 188,188,188,188,189
182  IF K < 5 THEN PRINT : PRINT TAB( 8
       )"- OR -": PRINT
183  NEXT
184  IF O < 5 THEN PRINT : PRINT TAB( 8
       )"+ AND +": GOTO 174
185  IF O = 5 THEN 189
186 O = O - 1: IF O = 0 THEN C = 1
187  GOSUB 73: GOTO 189
188  GOSUB 72
189  IF S = 3 THEN 200
190  GOSUB 80:V = 40: IF W > 0 THEN PRI
       NT "HOW MANY COLUMNS PER LINE?"
       ;: GOSUB 4:V =Z
191  GOSUB 81: PRINT : PRINT : PRINT :
       PRINT  SPC( 10)M$: PRINT: PRINT
       :PRINT :D = 0:I = 0: IF X < V T
       HEN P = X: FOR J = 1 TO Y: PRIN
       T L$(S%(J)) SPC( 2 + S%(400 + J
       ) -  LEN (L$(S%(J))));: NEXT :
       PRINT : GOSUB 82
192 I = I + 1: GOSUB 76: IF G = 5 THEN
       195
193  IF X < V THEN  FOR K = 1 TO Y:Q =
       S%(K): PRINT R$(Q); SPC( 2 + S%
       (400 + K) -  LEN (R$(Q)));:S(K)
       = S(K) + VAL (R$(Q)): NEXT : GOS
       UB 52: GOTO 195
194  GOSUB 52: GOSUB 52: GOSUB 52: FOR
       K = 1 TO Y:Q = S%(K): GOSUB 55:
       S(K) = S(K) + VAL (R$(Q)): NEXT
195  IF I < B THEN 192
196  IF X < V AND S = 2 THEN  GOSUB 82:
       GOSUB 53: GOSUB 52: FOR K = 1 T
       O Y:G = INT (S(K) * 100 + .5) /
       100: PRINT G; SPC( S%(40  0 + K
       ) + 2 - LEN ( STR$ (G)));: NEXT
       : GOSUB 52: GOTO 205
197  IF S = 2 THEN GOSUB 52: GOSUB 52:
       GOSUB 52: PRINT SPC( 10)"SUMS:"
       : GOSUB 53:GOSUB 52: GOSUB 52:
       FOR K = 1 TO Y:Q = S%(K) : PRIN
       T L$(Q)": " INT (S(K) * 100 + .
       5) / 100: GOSUB 53: NEXT : GOTO
       205
198  GOTO 205
199  PRINT : PRINT "ITEM TO COUNT?";: G
       OSUB 30:Y = Z: PRINT : PRINT A$
       " MUST BE SORTED": PRINT "BY ";
       : FLASH : PRINT L$(Y): NORMAL :
       GOTO 172
200  GOSUB 80: GOSUB 81:R$ = "":D = 0:C
       = 0:S%(I) = Y:X = 0: GOSUB 74:X
       = X + 5:P = X + 6: IF W = 0 THE
       N P = 40
201  PRINT : PRINT : PRINT L$(Y) SPC( X
       - 3 - LEN (L$(Y)))"FREQUENCY":
       GOSUB 82: PRINT: FOR I = 1 TO B
       : GOSUB 76: IF G = 5 THEN 204
202  IF R$(Y) < > R$ AND I < > 1 THEN P
       RINT R$ SPC( X -  LEN (R$))C:R$
       = R$(Y): GOSUB 53:C = 1: GOTO
       204
203 C = C + 1: IF I = 1 THEN R$ = R$(Y)
204  NEXT : PRINT R$ SPC( X - LEN (R$))
       C: GOSUB 53
205  GOSUB 50: IF O < 1 THEN 166
206  PRINT : PRINT : PRINT : FOR Q = 1
       TO 30: PRINT "*";: NEXT : PRINT
       : PRINT "THESE CRITERIA WERE US
       ED:": PRINT : PRINT :I = 0
207 I = I + 1: FOR K = 1 TO 5:U = (I -
       1) * 5 + K:Q = S%(200 + U): PRI
       NT L$(Q): PRINT SPC( 5)"MIN:"M$
       (U): PRINT SPC( 5)"MAX:"N$(U):D
       = 14: GOSUB 52: IF S%(201 + U)
       = 0 THEN 210
208  IF K < 5 THEN  PRINT : PRINT  SPC(
       8)"- OR -": PRINT
209  NEXT K
210  IF I < O THEN PRINT : PRINT  "----
       - AND": PRINT : GOTO 207
211  GOTO 166
212  HOME :E = 7: PRINT  TAB( 15)"UTILI
       TIES": VTAB 7: HTAB 9: PRINT "1
       UPLOAD": PRINT : HTAB 9: PRINT
       "2 DOWNLOAD": PRINT : HTAB 9: P
       RINT "3 DRIVE SELECT": PRINT :
       HTAB 9: PRINT "4 QUIT"
213  VTAB 18: GOSUB 28:C = 4: PRINT : G
       OSUB 32: ON Z GOTO 214,216,218,
       223
214  GOSUB 84: GOSUB 65: FOR I = 1 TO B
       : GOSUB 47: FOR J = 1 TO A: IF
       LEN (R$(J)) < L%(J) THEN FOR K
       = 1 TO L%(J) - LEN (R$(J)):R$(J
       ) = R$(J) + " ": NEXT
215  NEXT :Q = I: GOSUB 48: NEXT : GOSU
       B 65: GOTO 91
216  GOSUB 84: GOSUB 65: FOR I = 1 TO B
       : GOSUB 47: FOR J = 1 TO A: FOR
       K = L%(J) TO 1 STEP  - 1: IF MI
       D$ (R$(J),K,1) < > " " THEN R$(
```

```
        J) = LEFT$ (R$(J),K):K=1
217   NEXT : NEXT :Q = I: GOSUB 48: NEXT
        : GOSUB 65: GOTO 91
218   HOME : VTAB 6:E = 4: PRINT TAB( 5)
        "DISK DRIVE FOR DATA FILES?";:C
        = 2: GOSUB 31:V = Z:N = 0: GOTO
        89
219 Y =  PEEK (222): IF Y = 255 THEN 223
220   IF Y = 5 THEN HOME : VTAB 10: PRIN
        T A$" IS EMPTY!": GOSUB 3: CALL
        768: GOTO 91
221   TEXT : IF Y = 6 OR Y = 11 OR Y = 2
        OR Y = 3 THEN CALL 768: GOTO 93
222   POKE 216,0: RESUME
223   IF F = 1 THEN GOSUB 66
224   TEXT : CALL  - 868: HTAB 17: FLASH
        : PRINT "SO LONG": NORMAL
225   DATA 104,168,104,162,223,154,72,15
        2,72,96,ENTER, DELETE, CHANGE,
        PRINT, SORT                    ©
```


If you have an Original or an Upgrade BASIC PET, you'll find this repeating-keys program frequently useful, especially when you need to make corrections to a large program.

PET Auto Repeat

Art Hunkins
School of Music
University of North Carolina at Greensboro
Greensboro, NC

These programs were adapted from several sources. SYS889 enables them both. The same command also *disables* the repeat function, which, incidentally, works for *all* keys. Auto repeat must be disabled for cassette functions to operate.

In the Original ROM version, POKE914,(30) specifies the .5 second delay time (hold time) before the character begins to repeat. POKE932, (4) specifies the repeat rate. Either value can be changed. For Upgrade ROMs, the locations are, respectively, POKE927,(30) and POKE945,(4).

Both programs store in the second cassette buffer. Whenever the need for extensive program editing arises, an auto repeat function is a real timesaver.

Program 1.

```
100 REM FOR ORIGINAL ROMS
110 DATA120,56,169,233,237,26,2,141,26
120 DATA2,88,96,173,3,2,201,255
130 DATA208,12,169,0,141,119,3,169
140 DATA30,141,120,3,208,30,238,119
150 DATA3,173,120,3,205,119,3,176,19
160 DATA169,4,141,120,3,169,0,141,3,2
170 DATA141,119,3,169,2,141,37,2,24
180 DATA76,133,230
190 FORI=889TO952:READJ:POKEI,J:NEXT
```

Program 2.

```
100 REM FOR UPGRADE ROMS
110 DATA120,56,169,233,229,145,133,145
120 DATA165,144,201,46,208,6,169,147
130 DATA133,144,208,4,169,46,133,144
140 DATA88,96,165,151,201,255
150 DATA208,12,169,0,141,119,3,169
160 DATA30,141,120,3,208,28,238,119
170 DATA3,173,120,3,205,119,3,176,17
180 DATA169,4,141,120,3,169,0,133,151
190 DATA141,119,3,169,2,133,168,24
200 DATA76,46,230
210 FORI=889TO963:READJ:POKEI,J:NEXT ©
```

A grab bag of tricks and VIC techniques to make programming easier and safer.

VIC Curiosities

Doug Ferguson
Elida, OH

Here's a potpourri of odd things I discovered by accident on the keyboard of the VIC-20. I hope you find something useful.

Cold Start By SYS 64802

You, too, may hate turning your VIC off and on to clear all the funny POKEs you've made or to get a clean start after strange happenings. Save your power switch by typing SYS 64802. Of course, you'll still have to power down the usual way in the event of a crash or lock-up.

One-handed RUN

It is already generally known that some operations may be initiated using one hand. You can stop a program by merely hitting the RUN/STOP key. And you might know the quick way to LOAD: just hold down the left shift key with your thumb and touch the RUN/STOP key. But how does one initiate a RUN with one hand?

Easy. Type quotation marks followed by left-shifted RUN/STOP. I usually hold down the left shift key with my thumb and touch the "2" key and then the RUN/STOP key in sequence with my middle finger of the same hand. Try it! You'll see a quick flash of an error and then a RUN will begin. A lot easier than typing RUN and hitting RETURN all of the time. Incidentally, if you want a mysterious-looking two-handed RUN, try SYS 50830.

LIST Killer

Would you like to prevent nosy people from reading your program? Add a line to POKE 755,200. Unfortunately, it only works if they RUN it before they try to LIST it. By the way, don't let this trick prevent *you* from listing your own stuff. POKE 755,199 to restore the LIST function.

SAVE Killer

This is the LIST killer's big brother. (Like its sibling, it only works if the program is first RUN.) It also kills RUN/STOP and the RESTORE key in order to prevent easy reversal. To make it work, add this line to your program: POKE 802,0: POKE 803,0: POKE 818,165. This should discourage the casual thief. Remember, you cannot use your RUN/STOP key. To undo this little trick POKE 802,243: POKE 803,243: POKE 818,133. ©

A Light Pen For Under $10

William Hale, Albuquerque, NM

The light pen capabilities of the VIC-20 can be put to use for less than $10 (or for less than $2 if you have a connector to mate with game port connector). All that is needed is a ballpoint pen case or fine tip marker pen case, a photodiode or phototransistor, a resistor, three to four feet of shielded cable, and a 9-pin female connector.

The phototransistor can be a Radio Shack Catalog No. 276-138 ($.89) or a Sylvania ECG-3038 ($1.25). The ECG-3038 is the smaller of the two and fits closer to the fine tip of a marker pen. A 1/8-watt resistor, 1K for the ECG-3038 and 100K for the RS 276-138, is connected between the collector of the phototransistor and +5V located on Pin 7 on the game port connector. The collector of the phototransistor is also connected to the light pen input on Pin 6 of the game port connector and its emitter to ground, located on Pin 8.

The phototransistor is mounted in the tip of a pen case as near to the opening as possible and is tied to the connector via a shielded cable. Solder the collector to the center conductor and the emitter to the shield. A pushbutton can be added between the emitter and the shield at the pen to prevent false triggering. The other end of the shielded cable will, of course, be connected to the game port mating connector where the load resistor should be located between Pins 6 and 7. A one shot multivibrator could be added between the phototransistor output and Pin 6 in case the CRT of the user's monitor or TV screen is weak. In my case, I found it was not necessary with either of the light pens I constructed.

The principle of operation is quite simple. With the pen touching the screen, the scanning beam of the CRT causes the phototransistor to produce a negative pulse. This pulse is used by the VIC-20 to latch into memory locations 36870 and 36871 the horizontal and vertical position of the scan line via numbers ranging from 0 to 128. I was able to obtain repeatable readings out of the vertical location (36871), but the horizontal (36870) was not as reliable. In my case, I feel this was due to a slight 60H$_z$ ripple in my CRT sweep circuitry causing the scan line pulse to change periodic rate.

By PEEKing into these locations, a user could recognize the scan line position and branch a program accordingly. Uses could be a menu, listings, multiple choice answers, and for chase or move-the-target type games. ©

This hybrid (a mixture of BASIC and machine language) utility program will quickly locate a target within a large Atari "superstring."

Substring Search Utility

Edward C. Smith
Harrisburg, PA

The Atari can handle very long strings of data. However, searching for a substring within a long string is slow when the search is executed entirely in BASIC. A combination of BASIC and machine language results in greater speed. Both methods are presented below.

BASIC A+ [*sold by Optimized Systems Software*] employs an instruction called "FIND," which searches a long string to find a substring at a specified starting location. If you don't have BASIC A+, you can achieve similar results by utilizing a subroutine that combines BASIC and machine code.

Program 1 incorporates two subroutines for comparison purposes. A long string of data (5592 bytes) is created to permit searching for any of 100 records. If you RUN the program, the prompt "ENTER SUBSTRING" will appear. Respond by entering "RECORD #100." After the next prompt "ENTER START LOCATION," respond with "1." The next prompt "ENTER SELECTION 700 or 800." If you respond with 800, you have selected the BASIC subroutine to perform the search operation. If you start your stopwatch immediately after entering 800, it should take approximately 98 seconds until the answer "SEARCH RESULT = 5536" appears. This means that RECORD #100 starts at the 5536th byte of the long string (Y$). Confirmation of the result is indicated by the next line printed – "FOUND STRING = RECORD #100."

You can prove this to yourself by typing (in the direct mode): ?Y$(5536,5546). Now repeat the same procedure, except after the prompt "ENTER SELECTION 700 or 800," respond with "700." You should receive the same answer in less than one second. You may wish to try other substrings. All answers are referenced to the beginning of the main string.

Program Operation

Line 10 defines the size of all strings used. Y$ is the main string. X$ is the substring and should not exceed 255 bytes. DAT$ and R$ are used in construction of the main string.

Lines 20 and 25 create a long string (Y$) to simulate 100 records, numbered RECORD #1 to RECORD #100 (5592 bytes).

Line 40 loads the 92 machine code bytes.

Lines 50 to 65 are the input prompts. A starting location less than one is assumed to be one.

Line 70 directs execution of chosen subroutine.

Lines 80 to 85 are possible search results. If no substring is found, the search result is zero.

Lines 700 to 770 are the subroutine combining BASIC and machine language. Inputs required for the machine language portion defined by the USR function are: (1) Main string Y$, (2) Substring X$, (3) Length of Y$, (4) Length of X$-1, and (5) Starting location.

Lines 800 to 870 are the subroutine written entirely in BASIC. Inputs required for this routine are identical to those used for subroutine 700.

Lines 900 to 920 check for abnormal entries.

Lines 20000 to 21050 load the machine code bytes into page six of memory.

Program 2 is a liberally remarked listing of the assembled machine code.

Program 1.

```
5 REM SUBSTRING SEARCH PROGRAM
6 REM BY EDWARD C. SMITH
7 REM APRIL 6,1982
10 DIM Y$(6000),X$(255),DAT$(80),R$(9)
12 REM Y$ IS THE MAIN STRING - X$ IS THE SUB
   STRING - DAT$ AND R$ ARE USED TO DEVELOP
   THE MAIN STRING
15 REM LINES 20 TO 25 CREATE A LARGE STRING.
     PRINT Y$ TO SEE THIS STRING.
20 R$="RECORD #":DAT$=" NAME....ADDRESS....C
   ITY....STATE....PHONE...."
25 FOR I=1 TO 100:Y$(LEN(Y$)+1)=R$:Y$(LEN(Y$
   )+1)=STR$(I):Y$(LEN(Y$)+1)=DAT$:NEXT I
40 GOSUB 20000:REM LOAD MACHINE CODE BYTES
45 ? :REM SUBSTRING X$ MUST BE LESS THAN 255
    BYTES.
47 REM IF YOU ENTER 'RECORD #100',START LOCA
   TION 1, AND GO BASIC SUBROUTINE 800 SEARC
   H TIME WILL BE 98 SECONDS.
50 ? :? "ENTER SUBSTRING";:INPUT X$
52 ?
55 ? "ENTER SEARCH START LOCATION";:INPUT A1
57 ?
60 ? "700 BASIC PLUS MACHINE CODE"
62 ? "800 BASIC CODE ONLY"
63 ?
65 ? "ENTER SELECTION,700 or 800";:INPUT SR
67 ?
70 GOSUB SR:REM THE VALUE "A" RETURNED IS TH
   E # OF BYTES TO THE RIGHT OF (A1-1)
80 ? CHR$(253);"SEARCH RESULT=";A1+A-1
83 IF A=0 THEN ? "STRING NOT FOUND":GOTO 50
85 ? "FOUND STRING=";Y$(A1+A-1,A1+A+LX-2)
90 GOTO 50
700 REM STRING SEARCH USING BOTH BASIC AND M
    ACHINE CODE TOGETHER
705 LY=LEN(Y$):LX=LEN(X$):POKE 207,LX-1
```

```
710 GOSUB 900:IF A=0 THEN 770
720 B=LY-LX-A1+3
730 A=USR(1664,ADR(Y$(A1)),ADR(X$),B)
740 IF A=0 THEN A1=1
770 RETURN
800 REM STRING SEARCH USING BASIC ONLY
805 LY=LEN(Y$):LX=LEN(X$)
810 GOSUB 900:IF A=0 THEN 870
820 FOR I=1 TO LY-LX-A1+2
830 IF Y$(A1+I-1,A1+I+LX-2)=X$ THEN 850
840 NEXT I
845 A=0:A1=1:GOTO 870
850 A=I
870 RETURN
900 REM CORRECT START LOCATION A1 IF ENTERED
     VALUE IS OUT OF RANGE
905 A=1:IF A1<1 THEN A1=1
910 IF A1>LY-LX+1 OR LX>LY THEN A=0:A1=1
920 RETURN
20000 REM LOAD 92 MACHINE CODE BYTES
20005 FOR I=1664 TO 1755:READ A:POKE I,A:NEX
      T I:RETURN
20008 DATA 104,104,133,204,104,133
20010 DATA 203,104,133,206,104,133
20020 DATA 205,104,141,222,6,104
20030 DATA 141,221,6,169,1,133
20040 DATA 212,169,0,133,213,160
20050 DATA 255,200,177,203,209,205
20060 DATA 240,40,24,165,203,105
20070 DATA 1,133,203,165,204,105
20080 DATA 0,133,204,24,165,212
20090 DATA 105,1,133,212,165,213
21000 DATA 105,0,133,213,205,222
21010 DATA 6,208,216,165,212,205
21020 DATA 221,6,208,209,240,7
21030 DATA 152,197,207,208,204,240
21040 DATA 6,169,0,133,212,133
21050 DATA 213,96
```

Program 2.

```
10 ; SUBSTRING SEARCH PROGRAM
20 ;   BY EDWARD C. SMITH
30 ;      APRIL 6,1982
40 ; CALLED FROM BASIC BY
50 ; A=USR(1664,ADR(Y$(A1)),
    ADR(X$),B)
60 ; WHERE
70 ;  Y$ IS THE MAIN STRING
     LOCATED AT ADDRESS YD
80 ;
90 ;  X$ IS THE SUBSTRING
     LOCATED AT ADDRESS XD
0100 ;
0110 ;LX=LENGTH OF X$-1
0120 ;
0130 ;A1 IS THE START OF
      SEARCH MEASURED FROM
      LEFT END OF Y$
0140 ;
0150 ;B=LY-LX-A1+3
0160 ;A2 IS THE RESULT
      0 IF NOT FOUND
0170 ;
0180 ;NOTE:LX-1 MUST BE
      POKED AT 207 ($CF)
0190  *=$680
0200 YD=$CB
0210 XD=$CD
0220 LX=$CF
0230 A2=$D4
0240  PLA
0250  PLA ;GET ADDRESS OF MAIN STRING
0260  STA YD+1
0270  PLA
0280  STA YD
0290  PLA ;GET ADDRESS OF SUBSTRING
0300  STA XD+1
0310  PLA
0320  STA XD
0330  PLA ;GET ADDR OF MAX POSSIBLE COMPARES
0340  STA B+1
0350  PLA
0360  STA B
0370 ;INITIALIZE TO 1ST      BYTE OF MAIN STR
    ING
0380  LDA #$01
0390  STA A2
0400  LDA #$00
0410  STA A2+1
0420 START LDY #$FF
0430 NEXT INY
0440 CMP1 LDA (YD),Y ;COMPARE YTH BYTE OF MA
    IN STRING VS SUBSTRING
0450  CMP (XD),Y
0460  BEQ CMP2
0470  CLC
0480 ; MOVE TO NEXT BYTE     IN MAINSTRING
0490  LDA YD
0500  ADC #$01
0510  STA YD
0520  LDA YD+1
0530  ADC #$00
0540  STA YD+1
0550  CLC
0560 ;UPDATE RESULT LOCATOR
0570  LDA A2
0580  ADC #$01
0590  STA A2
0600  LDA A2+1
0610  ADC #$00
0620  STA A2+1
0630  CMP B+1 ;IS MAX ALLOWABLE COMPARES REA
    CHED?
0640  BNE START
0650  LDA A2 ;                    YES ON HIGH BY
    TE
0660  CMP B ;IS MAX ALLOWABLE COMPARES REACH
    ED?
0670  BNE START
0680  BEQ NOMATCH ;                    YES ON L
    OW BYTE
0690 CMP2 TYA ;HAVE ALL BYTES OF SUBSTRING B
    EEN LOOKED AT?
0700  CMP LX
0710  BNE NEXT ;NO
0720  BEQ RETN ;YES
0730 NOMATCH LDA #$00
0740  STA A2
0750  STA A2+1
0760 RETN RTS
0770 B=RETN+$02
```

A Monthly Column

All computer users can benefit from this month's column – many of Bill's observations and hints are not specific to the Atari. If you're thinking of translating a BASIC game program into machine language to achieve greater speed, you'll find some valuable information below. For example, there's a discussion of the "ball/boarder" problem which can be the most difficult puzzle to solve when programming certain kinds of games.

Insight Atari

Bill Wilkinson
Optimized Systems Software
Cupertino, CA

This month we return to the world of program writing. As I noted in my last column, there has been a growing demand for me to explain how to write graphics programs in assembly language. So I will begin a two or three-part series this month on converting BASIC programs to assembly language. Although the programs will be specifically written for the Atari computers, it won't take too much imagination to convert them to Apple and Commodore machines.

The Bouncing BASIC Ball

Since we are going to try to build up this program in stages, we will start this month with the simplest possible form. Program 1 is an Atari BASIC program which bounces a "ball" around inside the rectangular screen. There is no scoring, no paddles, no sound, no players, no missiles, no intelligence.

In fact, perhaps the only thing which needs explaining is the frequent occurrence of the subexpression: INT(n * RND (0)). With Apple Integer BASIC, one could obtain the equivalent function by coding RND(n); and I have often wished that Atari had let us include that capability in the original specifications for Atari BASIC (oh, well, maybe in the next version of BASIC A + ?). Anyway, the idea is to produce an integer random number in the range of 0 to n-1, inclusive.

So now let's examine the program as a whole. (First, a comment: I have used the convention that X means "horizontal" and Y means "vertical." This

Program 1. Simple Bouncing Ball Program

```
100 GRAPHICS 3
200 XMOVE=INT(5*RND(0))-2
300 YMOVE=INT(5*RND(0))-2
400 IF XMOVE+YMOVE=0 THEN 200
500 X=INT(40*RND(0))
600 Y=INT(20*RND(0))
700 XNEW=X:YNEW=Y
900 POKE 19,0:POKE 20,0:REM RESET TIMER
1000 REM LOOP STARTS HERE
1100 COLOR 0:PLOT X,Y
1200 COLOR 2:PLOT XNEW,YNEW
1300 X=XNEW:Y=YNEW
1400 XNEW=X+XMOVE:YNEW=Y+YMOVE
1500 IF XNEW<=0 OR XNEW>=39 THEN XMOVE=-XMOV
E
1600 IF XNEW<0 OR XNEW>39 THEN XNEW=X
1700 IF YNEW<=0 OR YNEW>=19 THEN YMOVE=-YMOV
E
1800 IF YNEW<0 OR YNEW>19 THEN YNEW=Y
1900 IF PEEK(19)=0 THEN 1000
2000 RUN
```

Program 2. Bouncing Ball Initialization

```
0000            20          .PAGE 'initialization'
                30 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                40 ;
                50 ; A SIMPLE BOUNCING BALL PROGRAM
                60 ;
                70 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                80 ;
0000            90          *=    $3000
                0100 LINE100
                0101 ;>>>
                            GRAPHICS 3

3000 A903       0110        LDA   #3
3002 20E830     0120        JSR   GRAPHICS
                0197 ;
                0198 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                0199 ;
                0200 LINE200
                0201 ;>>>
                            XMOVE=INT(5*RND(0))-2

3005 A904       0210        LDA   #4
3007 205731     0220        JSR   RND        ; GET RANDOM NUMBER FROM 0 TO 4
300A 38         0230        SEC
300B E902       0240        SBC   #2         ; NOW IS RANDOM FROM -2 TO +2
300D 8DE230     0250        STA   XMOVE      ; AS IN BASIC PROGRAM
                0297 ;
                0298 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                0299 ;
                0300 LINE300
                0301 ;>>>
                            YMOVE=INT(5*RND(0))-2

3010 A904       0310        LDA   #4
3012 205731     0320        JSR   RND        ; GET RANDOM NUMBER FROM 0 TO 4
3015 38         0330        SEC
3016 E902       0340        SBC   #2         ; NOW IS RANDOM FROM -2 TO +2
3018 8DE330     0350        STA   YMOVE      ; AS IN BASIC PROGRAM
                0397 ;
                0398 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                0399 ;
                0400 LINE400
                0401 ;>>>
```

can have some strange implications. See below.) We start by establishing the least detailed graphics mode (which is, incidentally, roughly equivalent to Apple's LO-RES mode). Then we set both of the variables XMOVE and YMOVE to a random number in the range -2 to +2, inclusive. (Do you see how? 'INT(5*RND(0))' gives a number from zero to four, inclusive, and we then subtract two from it.) But we don't allow both values to be zero (line 400). (In a real "Pong" type game, you wouldn't want the X-motion to ever be zero. Here, allowing XMOVE to be zero is instructive.)

We then give the ball a starting position with X in the range of 0 to 39 and with Y from 0 to 19. Note that both the current position (X and Y) and the to-be-made-current position (XNEW and YNEW) are set equal. This is simply to get things started evenly. Line 900 resets the system timer. (You will have to do something differently here if you are using an Apple.)

The main loop is almost as simple. First, we erase (COLOR 0) the old "ball" (note that we are erasing nothing if this is the first time through the loop). Then we PLOT the new ball with a convenient, visible color (COLOR 2). We update our current ball position (line 1300) and also our to-be-made-current position (line 1400).

It Gets A Bit Difficult

Here is where it begins to get tricky. If the ball will be *at or beyond* the edge(s) of the screen, we must reverse its movement, as appropriate (lines 1500 and 1700). But suppose that the movement has *already* carried it beyond the screen bounds; we must then bring it back in-bounds (lines 1600 and 1800). Finally, for this simple demo,

```
                         IF XMOVE+YMOVE=0 THEN 200

301B ADE230 0410         LDA  XMOVE
301E 18     0420         CLC
301F 6DE330 0430         ADC  YMOVE      ; XMOVE + YMOVE
3022 F0E1   0440         BEQ  LINE200    ; IF = 0  THEN 200
            0497 ;
            0498 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0499 ;
            0500 LINE500
            0501 ;>>>
                         X=INT(40*RND(0))

3024 A927   0510         LDA  #39
3026 205731 0520         JSR  RND        ; GET RANDOM NUMBER FROM 0 TO 39
3029 8DDE30 0530         STA  X          ; AND KEEP IT
            0597 ;
            0598 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0599 ;
            0600 LINE600
            0601 ;>>>
                         Y=INT(20*RND(0))

302C A913   0610         LDA  #19
302E 205731 0620         JSR  RND        ; GET RANDOM NUMBER FROM 0 TO 19
3031 8DDF30 0630         STA  Y          ; AND KEEP IT
            0697 ;
            0698 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0699 ;
            0700 LINE700
            0701 ;>>>
                         XNEW=X : YNEW=Y

3034 ADDE30 0710         LDA  X
3037 8DE030 0720         STA  XNEW       ; XNEW = X
303A ADDF30 0730         LDA  Y
303D 8DE130 0740         STA  YNEW       ; YNEW = Y
            0897 ;
            0898 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0899 ;
            0900 LINE900
            0901 ;>>>
                         POKE 19,0:POKE 20,0

3040 A900   0910         LDA  #0
3042 8513   0920         STA  19
3044 8514   0930         STA  20         ; DON'T NEED TO DO LDA #0 TWICE
            0997 ;
            0998 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            0999 ;
            1000 LINE1000
            1001 ;>>>
                          REM LOOP STARTS HERE

            1097 ;
            1098 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1099 ;
            1100 LINE1100
            1101 ;>>>
                          COLOR 0 : PLOT X,Y

3046 A900   1110         LDA  #0
3048 201531 1120         JSR  COLOR
304B AEDE30 1130         LDX  X
304E ACDF30 1140         LDY  Y          ; LOAD VALUES FOR SUBROUTINE CALL
3051 202031 1150         JSR  PLOT
            1197 ;
            1198 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1199 ;
            1200 LINE1200
            1201 ;>>>
                          COLOR 2 : PLOT XNEW,YNEW

3054 A902   1210         LDA  #2
3056 201531 1220         JSR  COLOR
3059 A900   1230         LDA  #0         ; (NEEDED FOR PLOT)
305B AEE030 1240         LDX  XNEW
305E ACE130 1250         LDY  YNEW
3061 202031 1260         JSR  PLOT
            1297 ;
            1298 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1299 ;
            1300 LINE1300
```

we simply do this loop until the clock ticks (4.26 seconds, roughly) and then start all over.

Even ignoring the limited goals of this program, there are a few significant flaws: (1) There is no visible border around the screen to tell you when and where the ball will "hit." (2) There are no sound effects. (3) The ball isn't round (or even remotely so). (4) Sometimes, the ball rebounds without hitting the wall. I am going to leave (1) and (2) for next time, and (3) can't really be changed without using player-missile graphics. But flaw (4) is an interesting one, and worth some discussion.

The problem lies in the basic algorithm I chose for moving the ball: the X and Y movements can range from -2 to +2 units, independently, and I move the ball each time in both X and Y according to the current movement factors (XMOVE and YMOVE). Let's take an example: suppose that the XMOVEment is zero and the YMOVEment is -2. And further suppose that the ball is currently at Y position +1 (one square from the edge of the screen). If I allow the ball to move to the new Y position determined by Y and YMOVE (YNEW= Y+YMOVE in line 1400), then it will be off the screen (YNEW will be -1). What to do?

One solution might be to pretend we have absorbent walls (IF YNEW<0 THEN YNEW=0). This will work, but will give strange flight paths for the ball. The solution I chose was to imagine that the ball hit the wall smack in the middle the two times I chose to make it visible. (Imagine: the ball is displayed at Y position +1. One-half of a time-tick later, it hits the wall and rebounds. Another one-half of a time-tick later, it has rebounded back out

```
            1301 ;>>>
                             X=XNEW : Y=YNEW

3064 ADE030 1310          LDA  XNEW
3067 8DDE30 1320          STA  X
306A ADE130 1330          LDA  YNEW
306D 8DDF30 1340          STA  Y
            1397 ;
            1398 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1399 ;
            1400 LINE1400
            1401 ;>>>
                             XNEW=X+XMOVE : YNEW=Y+YMOVE

3070 ADDE30 1410          LDA  X
3073 18     1420          CLC
3074 6DE230 1430          ADC  XMOVE
3077 8DE030 1440          STA  XNEW
307A ADDF30 1450          LDA  Y
307D 18     1460          CLC
307E 6DE330 1470          ADC  YMOVE
3081 8DE130 1480          STA  YNEW
            1497 ;
            1498 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1499 ;
            1500 LINE1500
            1501 ;>>>
                             IF XNEW<=0 OR XNEW>=39 THEN XMOVE=-XMOVE

3084 ADE030 1510          LDA  XNEW
3087 3006   1515          BMI  THEN1500   ;XNEW < 0
3089 F004   1520          BEQ  THEN1500   ;XNEW = 0
308B C927   1525          CMP  #39
308D 9009   1530          BCC  LINE1600   ;XNEW NOT >= 39
            1550 THEN1500
308F A900   1555          LDA  #0
3091 38     1560          SEC
3092 EDE230 1565          SBC  XMOVE      ;GET 0 - XMOVE
3095 8DE230 1570          STA  XMOVE      ; TO XMOVE
            1597 ;
            1598 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1599 ;
            1600 LINE1600
            1601 ;>>>
                             IF XNEW<0 OR XNEW>39 THEN XNEW=X

3098 ADE030 1610          LDA  XNEW
309B 3006   1620          BMI  THEN1600   ;XNEW < 0
309D C927   1630          CMP  #39
309F F008   1640          BEQ  LINE1700   ;XNEW = 39
30A1 9006   1650          BCC  LINE1700
            1660 THEN1600
30A3 ADDE30 1670          LDA  X
30A6 8DE030 1680          STA  XNEW       ; XNEW = X
            1697 ;
            1698 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1699 ;
            1700 LINE1700
            1701 ;>>>
                             IF YNEW<=0 OR YNEW>=19 THEN YMOVE=-YMOVE

30A9 ADE130 1710          LDA  YNEW
30AC 3006   1715          BMI  THEN1700   ;YNEW < 0
30AE F004   1720          BEQ  THEN1700   ;YNEW = 0
30B0 C913   1725          CMP  #19
30B2 9009   1730          BCC  LINE1800   ;YNEW NOT >= 19
            1750 THEN1700
30B4 A900   1755          LDA  #0
30B6 38     1760          SEC
30B7 EDE330 1765          SBC  YMOVE      ;GET 0 - YMOVE
30BA 8DE330 1770          STA  YMOVE      ; TO YMOVE
            1797 ;
            1798 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1799 ;
            1800 LINE1800
            1801 ;>>>
                             IF YNEW<0 OR YNEW>39 THEN YNEW=Y

30BD ADE130 1810          LDA  YNEW
30C0 3006   1820          BMI  THEN1800   ;YNEW < 0
30C2 C913   1830          CMP  #19
30C4 F008   1840          BEQ  LINE1900   ;YNEW = 39
30C6 9006   1850          BCC  LINE1900
```

to Y position + 1. We thus display it again at position + 1, since we are displaying only at integral time-ticks.) This choice is reflected in the programming in lines 1600 and 1800.

Of course, all "motion" via a computer is no more true motion than is a motion picture or a television picture. In truth, you are simply seeing a series of still pictures flashed in front of your eyes so quickly that your brain perceives the result as motion. Thus, there is nothing inherently wrong with my solution. Except that, from BASIC, the time between pictures is so long that even my lazy brain can sometimes clearly see that the ball didn't touch the wall. (Notice that if XMOVE is zero, so that we have only vertical ball movement, the effect is even easier to see.)

Can we do better? From BASIC, probably not. From assembly language, probably yes. If we choose a different algorithm, a different graphics mode, or make the pictures change faster, maybe we can give better illusions of motion. But that will wait for next time. This month, we will simply recode our BASIC routine in assembly language.

Having A Ball With Assembly Language

First note that the BASIC line numbers have been preserved, with line 100 in the assembly code having the label LINE100 and being followed, on line 101, with a remark containing the BASIC source for that line. (If you want to make your listings neat and readable, you might try the trick I used here: I placed a control-J [an ASCII line-feed character] both before and after the BASIC source. It can make your listing much more readable.)

Also note the inclusion of my graphics subroutines from the February issue of **COMPUTE!**

```
            1860 THEN1800
30C8 ADDF30 1870        LDA  Y
30CB 8DE130 1880        STA  YNEW       ; YNEW = Y
            1897 ;
            1898 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1899 ;
            1900 LINE1900
            1901 ;>>>
                      IF PEEK(19)=0 THEN 1000

30CE A513   1910        LDA  19
30D0 D009   1920        BNE  LINE2000
            1930 ;====  LEAVE THESE 6 LINES OUT FIRST TIME ====
30D2 A514   1940        LDA  20         ; LSB OF CLOCK
            1950 CLOCKWAIT
30D4 C514   1960        CMP  20         ; CHANGED YET?
30D6 F0FC   1970        BEQ  CLOCKWAIT ; NO...WAIT
            1980 ;====  BUT ALWAYS KEEP LINE 1990 ====
30D8 4C4630 1990        JMP  LINE1000
            1997 ;
            1998 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            1999 ;
            2000 LINE2000
            2001 ;>>>
                      RUN

            2010 ;
            2020 ; to truly simulate RUN, we should store
            2030 ; zero to all variables.  For this program
            2040 ; that is not necessary, since all variables
            2050 ; are reset in the beginning of the program
            2060 ; anyway.  In general, though, be careful
            2070 ; to check such things.
            2080 ;
30DB 4C0030 2090        JMP  LINE100
            3000 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            3010 ;
            3020 ; !!!   VARIABLES AND SUBROUTINES !!!
            3030 ;
            3040 ; There are no direct BASIC equivalents
            3050 ; for the following lines ... the BASIC
            3060 ; interpreter handles all this for you
            3070 ; automatically.
            3080 ;
            3090 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
            3100 ;
            3110 ; first -- the variable declarations
            3120 ;
30DE 00     3130 X      .BYTE 0
30DF 00     3140 Y      .BYTE 0
30E0 00     3150 XNEW   .BYTE 0
30E1 00     3160 YNEW   .BYTE 0
30E2 00     3170 XMOVE  .BYTE 0
30E3 00     3180 YMOVE  .BYTE 0
            3190 ;
            3200 ; second -- the subroutines
            3210 ;
            3220 ; these are mostly the same subroutines that
            3230 ; we presented in the February, 1982, issue
            3240 ; though they have been modified slightly.
            3250 ; Note that SETCOLOR and LOCATE have been
            3260 ; dropped (not used in this program)
            3270 ; and a new function, RND, has been added.
            3280 ;
            3290 ;
30E4        3300        .INCLUDE #D:GRAPHICS.ASM
```

Equates, etc., for graphics subroutines

```
30E4        9000        .PAGE 'Equates, etc., for graphics subroutines'
            9005 ;
            9010 ; CIO EQUATES
            9015 ;
E456        9020 CIO    =     $E456      ; Call OS thru here
0342        9025 ICCOM  =     $342       ; COMmand to CIO in IoCb
0344        9030 ICBADR =     $344       ; Buffer or filename ADdRess
0348        9035 ICBLEN =     $348       ; Buffer LENgth
034A        9040 ICAUX1 =     $34A       ; AUXilliary byte # 1
034B        9045 ICAUX2 =     $34B       ; AUXilliary byte # 2
            9050 ;
0003        9055 COPN   =     3          ; Command OPeN
000C        9060 CCLOSE =     12         ; Command CLOSE
```

(Issue #21). I have added a RaNDom function, to make the mainline code easier and more compatible with the BASIC original. Even if you choose not to type in the mainline assembly language this month, you should type in and preserve these routines. Or simply add RND to the listing you typed in from February (you *did* type all that in, of course). We will use these same routines in the later articles in this series, but the listing will *not* be repeated.

As much as possible, the assembly language is self-explanatory, especially when coupled with the BASIC source. For example, what could be clearer than the translation of "GRAPHICS 3" into "LDA #3" and "JSR GRAPHICS"? If you don't understand *why* this works, you really need to get a good introductory book and read up on 6502 assembly language. For those of you into such things, you might note that when we convert from BASIC to assembly language, we tend to convert expressions by using reverse Polish notation. Thus, for example, line 300's assembly language equivalent might be expressed in "pidgin-HP" (that is, in a parody of the keyboard language used by HP reverse Polish calculators) as something like this:

4 RND 2 – ENTER xmove STORE

And those of you into FORTH will presumably also see the obvious corollaries.

The assembly language coding here is not the best nor the most efficient. For example, lines 410 through 430 could be replaced by a simple "ORA XMOVE" (because the A-register already contains YMOVE and because we don't really need the sum to find out if the two values are both zero). Rather, the idea here was to do as straightforward a translation as possible, allowing more of

```
000B          9065 CPBINR =    11          ; Command Put BINary Record
0011          9070 CDRAW  =    17          ; Command DRAWto
              9075 ;
0004          9080 OPIN   =    4           ; OPen for INput
0008          9085 OPOUT  =    8           ; OPen for OUTput
              9090 ;
              9095 ;
              9100 ; EQUATES used by the S: driver and
              9105 ;   the VBLANK routines
              9110 ;
0055          9115 HORIZONTAL = $55
0054          9120 VERTICAL =  $54
02FB          9125 DRAWCOLOR = $2FB
02C4          9130 COLOR0 =    $2C4
              9135 ;
              9140 ; miscellany
              9145 ;
00FF          9150 LOW    =    $FF
0100          9155 HIGH   =    $100
D20A          9160 RANDOM =    $D20A
```

The graphics subroutines

```
30E4          9165         .PAGE 'The graphics subroutines'
              9170 ;
              9175 ;
30E4 00       9180 SAVECOLOR .BYTE 0      ; where COLOR is saved
              9185 ;
30E5 53       9190 SNAME   .BYTE 'S:',0   ; the filename for open
30E6 3A
30E7 00
              9195 ;
              9200 ;
              9205 ; GRAPHICS g
              9210 ;
              9215 ; ENTRY: A-reg contains graphics mode 'g'
              9220 ;  EXIT: Y-reg has completion status
              9225 ;
              9230 GRAPHICS
30E8 48       9235         PHA            ; save 'g'
30E9 A260     9240         LDX  #6*$10    ; file 6
30EB A90C     9245         LDA  #CCLOSE
30ED 9D4203 9250           STA  ICCOM,X
30F0 2056E4 9255           JSR  CIO       ; First, we must close file #6
              9260 ; (we ignore any errors from the close)
              9265 ;
30F3 A260     9270         LDX  #6*$10    ; again, file 6
30F5 A903     9275         LDA  #COPN     ; we will open this 'file'
30F7 9D4203 9280           STA  ICCOM,X
30FA A9E5     9285         LDA  #SNAME&LOW
30FC 9D4403 9290           STA  ICBADR,X  ; we use the file name 'S:'
30FF A930     9295         LDA  #SNAME/HIGH
3101 9D4503 9300           STA  ICBADR+1,X ; by pointing to it
              9305 ;
              9310 ; all is set up for OPEN, now
              9315 ; we tell CIO (and S:) what kind of open
              9320 ;
3104 68       9325         PLA            ; our saved 'g' graphics mode
3105 9D4B03 9330           STA  ICAUX2,X  ; is given to S:
              9335 ; (note that S: ignores the upper bits of AUX2)
3108 29F0     9340         AND  #$F0      ; now we set just the upper bits
310A 4910     9345         EOR  #$10      ; and flip bit 4
              9350 ; (Read the text.  S: expects this bit inverted
              9355 ;    from what normal BASIC usage is.)
310C 090C     9360         ORA  #$0C      ; allow read and write access (f
                                          or CIO)
310E 9D4A03 9365           STA  ICAUX1,X  ; make CIO and S: happy
3111 2056E4 9370           JSR  CIO       ; and do the OPEN of S:
3114 60       9375         RTS
              9380 ;
              9385 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              9390 ;
              9395 ; COLOR c
              9400 ;
              9405 ;  ENTER: Color 'c' in A-register
              9410 ;   EXIT: Unchanged
              9415 ;
              9420 COLOR
3115 8DE430 9425           STA  SAVECOLOR
3118 60       9430         RTS            ; exciting, wasn't it?
              9435 ;
              9440 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
              9445 ;
              9450 ; POSITION h,v
```

you to understand how simple assembly language can be.

Are there any tricky spots in the code? Not really. Though, if you are like me, you will have to pause each time you use a CMP and figure out if you really want BCS or BCC (or whether you also need a BEQ or...). Again, some of the CMP's could have been made simpler (for example, by using 'CMP #40' on line 1630 and omitting line 1640). And, again, I opted for consistency with the BASIC program.

The program does work. Try it. It took me about three hours to type it in and debug it (including about an hour of debugging the debugger). This represents much less time than it would have taken if I had not had the BASIC program as a working model. You might omit lines 1930 to 1980 the first time you run it. I won't tell you what will happen, but I will tell you that the lines are used to synchronize ball movement with the clock.

On Assembling And Debugging

You may have noted that the master origin ('*=') for this program is at $3000. If you use that origin and don't do anything special, assembling the program will wipe out the source code and *kablooey!* What can you do? Personally, I prefer to direct the object code to disk when I assemble. (I usually use 'ASM ,#R:,#D:file.OBJ' where "file" is the same name as the source file and I use "R:" because I list to a DIABLO or DEC serial printer.) Then, with the source also safely LISTed to disk, I can use NEW and reLOAD the object and proceed to run and debug it. Using this method, it makes sense to place the origin somewhere fairly high in EASMD's (or the Assembler/Editor's) working

```
                9455 ;
                9460 ;  ENTER: h (horizontal) position in X,A
                9465 ;                 registers (LSB,MSB)
                9470 ;           v (vertical) position in Y-register
                9475 ;
                9480 ;    EXIT: unchanged
                9485 ;
                9490 POSITION
3119 8655       9495          STX  HORIZONTAL
311B 8556       9500          STA  HORIZONTAL+1 ; read the text
311D 8454       9505          STY  VERTICAL  ; too simple, right?
311F 60         9510          RTS
                9515 ;
                9520 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                9525 ;
                9530 ; PLOT h,v
                9535 ;
                9540 ;  ENTER: must have done a previous COLOR call
                9545 ;         X,A,and Y registers set as in POSITION
                9550 ;
                9555 ;   EXIT: Y-register has completion status
                9560 ;
                9565 PLOT
3120 201931     9570          JSR  POSITION
3123 A260       9575          LDX  #6*$10     ; file 6, again
3125 A90B       9580          LDA  #CPBINR    ; Command Put BINary Record
3127 9D4203     9585          STA  ICCOM,X
312A A900       9590          LDA  #0
312C 9D4803     9595          STA  ICBLEN,X
312F 9D4903     9600          STA  ICBLEN+1,X ; if buffer length is zero...
3132 ADE430     9605          LDA  SAVECOLOR ; then CPBINR puts one char from
                                 A-reg
3135 2056E4     9610          JSR  CIO        ; and this is how we PLOT
3138 60         9615          RTS
                9620 ;
                9625 ;
                9630 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                9635 ;
                9640 ; DRAWTO h,v
                9645 ;
                9650 ;  ENTER: must have done a previous PLOT
                9655 ;         X,A,and Y registers as in POSITION
                9660 ;
                9665 ;   EXIT: Y-register has completion code
                9670 ;
                9675 DRAWTO
3139 201931     9680          JSR  POSITION
313C ADE430     9685          LDA  SAVECOLOR
313F 8DFB02     9690          STA  DRAWCOLOR ; where DRAWTO expects its color
3142 A260       9695          LDX  #6*$10     ; file 6...once more
3144 A911       9700          LDA  #CDRAW     ; just a command to "S:"
3146 9D4203     9705          STA  ICCOM,X
3149 A90C       9710          LDA  #$0C
314B 9D4A03     9715          STA  ICAUX1,X  ; insurance
314E A900       9720          LDA  #0
3150 9D4B03     9725          STA  ICAUX2,X  ; ...guaranteed to work
3153 2056E4     9730          JSR  CIO        ; do the actual DRAWTO
3156 60         9735          RTS
                9740 ;
                9745 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                9750 ;
                9755 ;
                9760 ; RND r
                9765 ;
                9770 ;  ENTER: r in A-register
                9775 ;   EXIT: A-register contains a random
                9780 ;          number from 0 to r, inclusive
                9785 ;
                9790 ;
                9795 RND
3157 8D6631     9800          STA  RTEMP
315A EE6631     9805          INC  RTEMP      ; makes CMP easier
                9810 RNDWAIT
315D AD0AD2     9815          LDA  RANDOM     ; set a random number
3160 CD6631     9820          CMP  RTEMP      ; too big?
3163 B0F8       9825          BCS  RNDWAIT    ; yes...wait
3165 60         9830          RTS
                9835 ;
3166 00         9840 RTEMP    .BYTE 0         ; RESERVE SPACE
                9845 ;
                9850 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                9855 ;
3167            9999          .END
```

memory.

An alternative method is to keep the object code in memory *below* all my source listing. With EASMD this is easy to do. For example, with this program, I simply used a 'LOMEM 3800' command to tell EASMD not to use any memory below $3800. With the Assembler/Editor cartridge, it is almost as easy: simply use BUG to issue "C2E5 < 00,38" and then "G A000". ($02E5 is system LOMEM, which the Assembler picks up and uses for its own when it is coldstarted at $A000.) In both instances, make sure you have LISTed off any program in memory before changing the LOMEM bound, since it is the occurrence of NEW which forces the change.

Actually, I often use *both* of the above measures. And even then I can run into problems. When I was working on this month's program, for example, I could assemble and then load the program fine. But when I went to use "G3000" from BUG, the system looped madly. I'm still trying to figure out why, but I solved it by loading the OBJect file from the operating system and then reentering the Assembler via a cold start. BUG then worked fine. I hope that by next month I will have figured out the reason for this strange behavior and will report a fix to you. (To be fair, I am using a *very* early pre-release version of the cartridge...perhaps you won't have this problem.)

Breakpoint Setting

Possibly the biggest fault of BUG (both versions) is the lack of easy breakpoint capabilities. Changing instructions to BRKs ($00) and back often gets so tiresome that I tend to say the heck with it and try out an otherwise unchecked portion of code. When I'm lucky, it all works. When I'm not, I turn off the power and start again. Thank goodness I'm not trying to do this with just a cassette. The corollary? If you are using a cassette-only system, proceed with utmost caution and take the trouble to set lots of breakpoints.

That's about it for this month. Next month we will add several complications to the bouncing ball program. We will also explore some news, trivia, and gossip. And, whatever you do, don't believe everything that people say about the Atari and Atari BASIC: we may have some surprising benchmarks for you. ©

For PET/CBM and VIC, this handy utility should solve some memory space problems, especially when instructions can be safely deleted from a program after they're no longer needed. It cannot work, however, on Original ROM PETs.

Electric Eraser

Louis F. Sander
Pittsburgh

If you program in BASIC, you'd sometimes like to delete certain program lines after they've been executed, either to protect your program from piracy, or to free up memory for the rest of the program to use. The lines that print your on-screen instructions, for example, are good candidates for deletion as soon as they've been run. Having served their purpose, they do nothing but take up space, which can really be at a premium in small-memory machines like the VIC and the 8K PET.

It would be a real help if there were an easy way to delete such lines under program control. Well, now there *is* one: Electric Eraser is a two-line routine that deletes itself and all subsequent lines as soon as it's called.

Lines 210 and 220 in the accompanying program are the Electric Eraser for Upgrade and 4.0 ROM PET/CBM machines. If you have a VIC, your eraser appears in the REMarks following line 300. Move it up to lines 210-220 before you run the forthcoming demo. In all cases, line 300 activates the Eraser. There is nothing special about this choice of line numbers, and the three lines can be renumbered at will when you use them in other programs. They consume just over 100 bytes of memory.

To use the Eraser, you must set up the lines to be erased as the last lines in your program. There can be as many of them as you wish, and they should preferably include the activator line, since you'll have no need for it once the other lines have been erased. Put the Eraser immediately before the first line you want to erase. Then your program can execute any of its lines, except for the activator, to its heart's content.

There's no need to bypass the Eraser, since it has no meaningful effect until it's activated. When it's time for the Electric Eraser to do its work, execute the activator line. This will clear all variables and make the Eraser and everything after it disappear from the program. You can, if you like, replace the END in the Eraser with another statement, and it will be executed after it is deleted (!). If you leave out the END altogether, the subsequent lines may be executed, depending on what's in them, or your program may crash.

Watch It Work

Right now, let's see the Electric Eraser at work. Type in the demo program and SAVE it. Don't RUN it first to check your work, or you'll have to type it in again! LIST the program and carefully check lines 210, 220 and 300 for errors. Now RUN the program, and see for yourself that all its lines are actually executed, which should be obvious from the text that prints on the screen. RUN the program again, and you'll see that lines 210 and up do not execute this time, and that you now have several hundred more bytes of free memory. LIST the program to verify that lines 210-350 are no longer there. They have been electrically erased. You *could* say that these lines were executed, then they were executed. Or maybe they were just RUN to death. Anyway, they are gone without a trace, replaced by usable memory.

Eraser's Secret

Here is where they went: the first two PEEKs in line 210 are the keys to Electric Eraser's success. These locations contain a pointer to the start of the line currently being executed. When activated, the Eraser POKEs zeros into the link for that line and, using the USR vector as a temporary storage area, sets the Start of Variables pointer to the location just above that. As a result, BASIC thinks the program ends with the last line before the Eraser, which of course it now does. If all this is over your head, the System Information chapter of Osborne's *PET/CBM Personal Computer Guide* holds the keys to understanding. If you don't care about such matters, don't worry – you can use the Electric Eraser without understanding how and why it works.

Now you've seen the Electric Eraser in all its simple splendor, and maybe you've been impressed. If so, your next step is to add it to your bag of programming tricks, and to make equally impressive use of its powerful erasatorial punch. You could exercise your talents on the demo program, by replacing the END in line 220 with a RUN.

```
100 PRINT"{DOWN}FRE(0)=";FRE(0)
110 PRINT"{DOWN}WHERE IS THE REST OF THIS"
120 PRINT"LITTLE PROGRAM?"
200 REM ** 210 - 220 ARE THE ERASER
210 A=PEEK(58)+256*PEEK(59)+3:POKE2,INT(A/2
    56):POKE1,A-256*PEEK(2)
220 IFERTHENPOKEA-2,0:POKEA-1,0:POKE42,PEEK
    (1):POKE43,PEEK(2):CLR:END
230 PRINT"{DOWN}IF YOU LIST IT, YOU WON'T"
240 PRINT"FIND IT! IF YOU RUN IT ONCE"
250 PRINT"MORE, YOU'LL SEE THAT YOU"
```

```
260 PRINT"HAVE GAINED SOME MEMORY."
270 PRINT"{DOWN}THE ELECTRIC ERASER IS"
280 PRINT"POWERFUL MEDICINE!!"
300 ER=1:GOTO210:REM ** ACTIVATOR
310 REM
320 REM ** ERASER FOR THE VIC:
330 REM
340 A=PEEK(61)+256*PEEK(62)+3:POKE2,INT(A/2
    56):POKE1,A-256*PEEK(2)
350 IFERTHENPOKEA-2,0:POKEA-1,0:POKE45,PEEK
    (1):POKE46,PEEK(2):CLR:END                    ©
```

CBM Maintenance In-House

ALL UNDER ONE ROOF

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

31 years

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

VIC 20 - VIC 1515 PRINTERS - VIC 1540 DISK DRIVES

DES ANNOUNCES

"VIC-VILLE"©

commodore

*— ONE STOP **VIC 20** CENTER —*

SOFTWAREHARDWARE**EXPANSIONS**PERIPHERALS**ACCESSORIES**

GAMES: From our professional programmers
(Robert Winter, Ralph Orton, Dan Haste, Robert Burnett, Doug Weick, Doug Cornish)

GOLDBRICK Many levels of play, sound and color	**$14.00**
A MAZE ING Travel through the maze game of skill and tense action.	**$12.00**
GOBBLER Sounds Easy? You have 25 seconds to get him and the time gets shorter at each higher level.	**$11.00**
HANG-U Traditional Hangman plays against the VIC's 250 word dictionary OR another person.	**$12.00**
COGGLE Computerized version of Boggle.	**$11.00**
BASEBALL STRATEGY The excitement of baseball as a video strategic game.	**$12.00**
ATTACK ON SILLO III You are the commander of Silo III. Defend your base.	**$12.00**
YAHTZEE Solitaire version of this famous dice game.	**$12.00**
AIR STRIKE Fly the new super bomber V-20 on a mission.	**$11.00**
PANZER ATTACK Enemy tanks are attacking, and you must destroy them in the Combat Zone.	**$14.00**
PEDESTRIAN POLO Drive the car through the streets of America.	**$14.00**
ASTRO MINERS Hi-res graphics and sound space game requires 3k or 8k expansion	**$17.00**
VIC POKER Play Poker against the VIC. Hi-res graphics and sound.	**$14.00**
SIMPLE INVENTORY CONTROL LIFO System works with 5K VIC to 32K VIC. Complete documentation.	**$49.95**
GALACTIC BLASTER The fate of the Earth is in your hands.	**$12.00**
COMING SOON! VIC ADVENTURE # 1	**$20.00**

3 Slot Mini-Mother Memory Exp. Bd. **$69.95**

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆

" ONLY GAME IN TOWN"
VIC-20 " BOSS " CHESS

- 10 LEVELS
- 2 CLOCKS
- CHOOSE COLORS
- FULL SOUND
- HI-RESOLUTION GRAPHICS
- FOLLOWS COMPLETE RULES OF CHESS

This is the best VIC-20 game we've seen !
ONLY $39.95

** 8K exp. required*

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆

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** _____

VIC 20 - VIC 1515 PRINTERS - VIC 1540 DISK DRIVES

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

Built on a firm foundation!

Put a digital clock on the Atari screen which ticks away, regardless of what's going on in BASIC. Type in this BASIC program and, after you run it, the clock will operate until you hit RESET. This clock has several worthwhile uses, not to mention the general applicability of this technique to other independent operations you might want to perform.

System Clock For The Atari

Bill Zimmerman
Littleton, CO

Run this BASIC program. Enter the correct time. A small digital clock will appear in the upper right-hand corner of the screen. The program ends. You are back to BASIC, but the little clock is still there counting the seconds. It stays there until you press RESET or disable the display. In the meantime, both you and your programs enjoy instant access to the correct time of day.

The key to this is the vertical blank routine of the Atari operating system. Sixty times every second, while the electron beam leaps to the top of the TV screen, your Atari steals a little time for itself. During this short interval the Atari restores numerous system values, maintains its real time clock and looks at the keyboard and game controllers.

A Polite Operating System

The Atari operating system even has manners! If a time-critical operation (like a disk read) is interrupted, the OS will add one to its clock and immediately return to the waiting operation. If a normal operation is interrupted, the entire vertical blank routine is executed. For a detailed description see page 99 of the *Atari Operating System User's Guide* (C016555).

The Atari documentation claims that page six (addresses 1536 to 1791) is used by the system only when the power is turned on. During testing, I discovered that page six is used by the BASIC LIST function. For this reason the clock routines are stored in high memory behind RAMTOP where they are safe. Safe? Did you know that shift CLEAR spreads 64 zeroes past RAMTOP? So the routines end up a full page (256 bytes) above RAMTOP. Safe? Not quite. If you list more than a few lines of your program at one time through a text window, the system may go into an endless internal loop. Do a GR.0 before scanning through your program.

At line 110, the program finds the current value of RAMTOP and sets PAGE to one less. At line 120, RAMTOP is reset to one page lower than PAGE and the address PAGEADDR is calculated. Lines 150-180 POKE the clock routines into memory, beginning at PAGEADDR. Since the assembler routines are compiled relative to page six, all sixes are changed to the new base – PAGE.

Lines 160 and 170 POKE six back into locations which really were sixes. Lines 200 through 280 accept the current time and perform some elementary editing checks. Lines 300 to 330 POKE the time values one digit at a time into the clock, and line 340 sets the clock in motion.

The OS vertical blank routine is reached by the computer through a special address called a vector. The system clock program changes the vertical blank vectors to point to its own code.

The clock and its control byte may be accessed by any BASIC program. The following routine will recalculate PAGEADDR:

```
10 RAMTOP=6*16+10
20 PAGE=PEEK(RAMTOP)+1
30 PAGEADDR=PAGE*256
```

The control byte is at PAGEADDR. To temporarily disable the display, POKE PAGEADDR,0. You might want to do this for games or when the clock would interfere with your screen. To redisplay the clock, POKE PAGEADDR,1.

The clock is stored in the six bytes following the control byte. Hours are stored in PAGEADDR+1 and PAGEADDR+2, minutes are stored in PAGEADDR+3 and PAGEADDR+4, and seconds are stored in PAGEADDR+5 and PAGEADDR+6. A program needing the current time could execute the following routine:

```
40 SAVETIME=0
50 CURRTIME=1000*PEEK(PAGEADDR+1)+100
   *PEEK(PAGEADDR+2)+10*PEEK(PAGEADDR
   +3)+PEEK(PAGEADDR+4)
60 IF CURRTIME<>SAVETIME THEN SAVETIME
   =CURRTIME:GOTO50
```

Type GR.2 for the big-screen effect, then GR.0 when you are ready to use your computer again.

Be careful when typing the DATA statements. A mistake will probably have dire consequences. In fact, it would be wise to save your work before RUNning the first time.

```
10 REM SYSTEM CLOCK FOR ATARI
20 REM WRITTEN BY BILL ZIMMERMAN
```

```
30 REM CLOCK CONTROL AT (RAMTOP)+1
40 REM 1 = DISPLAY
50 REM 0 = NO DISPLAY
60 REM CLOCK VALUE AT ((RAMTOP)+1)+1
70 REM SIX BYTES - HHMMSS
80 REM
100 DIM A$(3)
110 RAMTOP=6*16+10:PAGE=PEEK(RAMTOP)-1
120 POKE RAMTOP,PAGE-1:GRAPHICS 0:PAGEADDR=P
    AGE*256
130 ? :? :? ,"WINDING THE CLOCK":? :? :?
140 REM ARK POKE CLOCK INTO RESERVED MEMORY
150 FOR I=0 TO 237:READ X:IF X=6 THEN X=PAGE
160 POKE PAGEADDR+I,X:NEXT I
170 POKE PAGEADDR+9,6:REM REAL SIXES
180 POKE PAGEADDR+45,6
200 ? "WHAT TIME (HHMM)";:INPUT TIME
210 THH=INT(TIME/100):TMM=TIME-THH*100
220 IF THH>23 THEN 200
230 IF TMM>59 THEN 200
240 IF THH<>12 THEN 270
250 ? "MIDDAY";:INPUT A$:IF A$(1,1)<>"Y" THE
    N THH=0
260 GOTO 300
270 IF THH>12 THEN 300
280 ? "AM OR PM";:INPUT A$:IF A$(1,1)="P" TH
    EN THH=THH+12
290 REM POKE IN TIME AND START CLOCK
300 X=INT(THH/10):POKE PAGEADDR+1,X
310 Y=INT(THH-X*10):POKE PAGEADDR+2,Y
320 X=INT(TMM/10):POKE PAGEADDR+3,X
330 Y=INT(TMM-X*10):POKE PAGEADDR+4,Y
340 X=USR(PAGEADDR+8)
350 END
1000 DATA 1, 0, 0, 0, 0, 0, 0, 196, 169, 6
1001 DATA 160, 28, 162, 6, 32, 92, 228, 169,
      7, 160
1002 DATA 54, 162, 6, 32, 92, 228, 104, 96,
     238, 7
1003 DATA 6, 240, 3, 76, 95, 228, 173, 0, 6,
      73
1004 DATA 128, 141, 0, 6, 238, 6, 6, 169, 19
     6, 141
1005 DATA 7, 6, 208, 235, 162, 4, 138, 208,
     24, 173
1006 DATA 1, 6, 41, 2, 240, 17, 173, 2, 6, 4
     1
1007 DATA 4, 240, 10, 169, 0, 141, 1, 6, 141
     , 2
1008 DATA 6, 240, 50, 169, 9, 221, 2, 6, 176
     , 45
1009 DATA 56, 189, 2, 6, 233, 10, 157, 2, 6,
      254
1010 DATA 1, 6, 169, 197, 141, 7, 6, 169, 5,
      221
1011 DATA 1, 6, 176, 105, 169, 0, 157, 1, 6,
      202
1012 DATA 48, 97, 202, 169, 196, 141, 7, 6,
     254, 2
1013 DATA 6, 16, 179, 240, 84, 173, 0, 6, 74
     , 144
1014 DATA 78, 10, 141, 0, 6, 165, 204, 72, 1
     65, 205
1015 DATA 72, 24, 173, 48, 2, 105, 4, 133, 2
     04, 173
1016 DATA 49, 2, 105, 0, 133, 205, 160, 1, 1
     77, 204
1017 DATA 72, 136, 177, 204, 105, 30, 133, 2
     04, 104, 105
1018 DATA 0, 133, 205, 162, 4, 160, 8, 189,
     2, 6
1019 DATA 32, 222, 6, 189, 1, 6, 32, 222, 6,
      202
1020 DATA 48, 8, 169, 10, 32, 222, 6, 202, 1
     44, 233
1021 DATA 104, 133, 205, 104, 133, 204, 238,
      0, 6, 76
1022 DATA 98, 228, 9, 16, 13, 0, 6, 145, 204
     , 136
1023 DATA 96, 13, 0, 6, 145, 204, 136, 96
```

©

This is an explanation for PET and VIC owners who wonder what BASIC looks like to the computer (it's not exactly what's on the screen). Also, have you ever needed to send text to your printer from machine language? This article explores both of these topics.

Inner BASIC

Jim Butterfield
Toronto

Question: When I type in a line of BASIC, how is it stored in memory? I've looked at the contents of hexadecimal addresses 400 and up in my PET and can't recognize anything.

Question: How do I print on my printer from machine language?

The two questions are partly related.

When a BASIC line is typed with a line number (so that it goes into memory), it will be stored almost as typed. In the PET, it will go into the area from hex 0400 and up. In the VIC it depends on the system: a minimum 5K VIC uses the area from hex 1000 and up. Without explaining in detail, here are the parts of a BASIC line stored in memory:

First two bytes: address link to next line ... or, if zero, end of program.

Next two bytes: line number in binary

Remainder: BASIC text with tokens

End-of-line: zero byte

If you don't know about tokens, you might read Herman's "Tokens Aren't Just For Subways" in *COMPUTE!'s First Book of PET/CBM*. So: 10 PRINT"XXX" will become: 0C 04 (link to next line at hex 040C); 0A 00 (line number 10); 99 (PRINT token); 22 58 58 58 22 ("XXX"); 00 (end of line).

That's not machine language; it's just tokenized BASIC. If you'd like to see where the interpreter does its machine language work, look up PRINT in a memory map; you can then disassemble and try to make sense out of it.

To PRINT in machine language, LOAD the A register with the ASCII character and call (JSR) hex FFD2. The character will print to the "standard" output – the screen.

To PRINT to a device other than the screen, the file must be OPENed first; this is most easily done from BASIC. When the machine language program is ready to PRINT, select the device with LDX (logical file number)/JSR $FFC9 – this sequence is equivalent to CMD (logical file number). Now PRINT as above. When you have finished for the moment, disconnect the device with JSR $FFCC. Eventually, you should CLOSE the file. Again, this is most easily done in BASIC.

If you have a PET, try entering the following information in hex:

```
0400 00 13 04 0A 00 9F 31 2C
0408 34 3A 9E 31 30 34 35 3A
0410 A0 31 00 00 00 A2 01 20
0418 C9 FF A9 41 20 D2 FF 18
0420 69 01 C9 5B D0 F6 A9 0D
0428 20 D2 FF 20 CC FF 60 00
```

This will also work on a VIC with a 3K expansion module. However, if you have the minimum 5K VIC, try entering:

```
1000 00 13 10 0A 00 9F 31 2C
1008 34 3A 9E 34 31 31 37 3A
1010 A0 31 00 00 00 A2 01 20
1018 C9 FF A9 41 20 D2 FF 18
1020 69 01 C9 5B D0 F6 A9 0D
1028 20 D2 FF 20 CC FF 60 00
```

You'll need a monitor for the VIC to do this, of course.

We have entered a program that is both BASIC and machine language. BASIC is contained in the first two and a half lines; the rest is machine language. Check it carefully. You can go back to BASIC and LIST the BASIC part. To see the ML part you'll need a disassembler.

The program as given should RUN, but to wrap things up neatly we should do one more thing: set the Start-of-Variable pointer. It's good practice and will make our program SAVE-able. On the PET, we should put address 042F into this pointer (located at hex 7C and 7D on Original ROM PETs; hex 2A and 2B on newer machines). On the VIC, we should put address 102F into the pointer at hex 2D and 2E. Don't forget that addresses go in backwards, or low order first, so that in the case of newer PETs, value 2F would go into address 2A and value 04 into address 2B.

Whether the pointers are fixed up neatly or not, you may go back to BASIC and say RUN. The program, all 47 bytes of it, causes the alphabet to be output to the printer.

Sometimes a working example is worth many pages of explanation. Try this one. Dissect it. See if you can see how it works.

We've written a BASIC program in hexadecimal, manufacturing line number, tokens, and all. Then we wrote a linked machine language program, and made it all work together.

Now see if you can output the numeric digits as well as the alphabet. ©

Skyles Electric Works

WORD-WRITER

Add this one small device and greatly increase the power of your 8032, 8096 or SuperPet.

This one small device can double, triple or quadruple the usefullness of your Commodore Business Machine.

Open your CBM computer and place this modern wonder chip in the rear empty socket. Close the lid and trun on your CBM with **WORD-WRITER a built-in word processing system.** Type "SYS36864", "RVS", key and "L" for letter. That is all there is to it — no diskettes or tapes to hassle with. Start writing!! You never knew a word processor could be so easy to use. Make a mistake? **No problem,** cursor to the front of the mispelling type "RVS" key "G key" and gobble up the offending word. Now strike "RVS," "I" (for insert) and type in the correct word or phrase. You only want to change a letter, or a line? **No problem,** a couple simple key strokes will do it.

WORD-WRITER

Want to move or copy a line or paragraph? Want to insert a something? **No problem,** a few simple practical keystrokes and it is done. Want to change a word because you have thought of a better word? Want to change a name everywhere it appears? Want to change some occurrences of a word? **No problem,** two (2) count them, two keystrokes set you up to do this.

Do most of your letters and reports have standard paragraphs in them? **No problem,** write them once save them on disk or cassette and use them thousands of times with only a few keystrokes of typing for each paragraph.

Have a lengthy report, memo or letter to write? **No problem, Word-Writer** holds up to 15 — count them — 15 pages of text at one time.

Want to search, scroll, skim, skip, flip or jump through the text? **No problem,** a couple of keystrokes and you are on your way.

What about typing out or printing the text? **No problem,** a few keystrokes and you can print out a rough draft or final copy on whatever printer you have attached to your PET/CBM

This modern wonder chip:

WORD-WRITER

Available Immediately from your local dealer at a modest cost of only **$85.00** complete with an easy reading manual.

Try Word-Writer for 10 days, if you aren't completely amazed that such a small item could make such a major improvement to your CBM8032, CBM8096 or SuperPet, please return it for a full refund.

INSTALLATION: Installs into socket UD11 ($A000)

PRICE: For CBM 8032, 8096 or SuperPET SP9000 WORD-WRITER $85.00
Please specify your PET/CBM model when ordering

AVAILABILITY: Immediately from your LOCAL DEALER

or

VISA, MASTERCHARGE ORDERS CALL (800) 227-998 (except California residents)
CALIFORNIA ORDERS PLEASE CALL (415) 965-1735

Skyles Electric Works

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

Making backup copies of a disk can present problems when you are using a single-drive device. Owners of the Commodore 2031 disk drive should find this file copying program indispensable.

Programs 1 and 2 are BASIC loaders. You type in the version for your PET (either Upgrade or 4.0) and it will create the machine language for you. Then, to start the program, type SYS 634.

Copy 2031 Files

G. H. Watson
University of Delaware
Newark, DE

Mass data storage and retrieval has been made convenient, fast, and reliable for the microcomputer user with access to a floppy disk drive. With the introduction of the CBM 2031 Single Disk Drive, Commodore has allowed the benefits of disk storage to be available to PET/CBM VIC owners with even the smallest computing budgets. However, while a single drive is more affordable than a dual drive, certain handicaps soon become apparent.

A major problem is the inability to quickly produce backup copies of disk files on a different diskette (handled easily on a dual drive with a single command). With the program here, Copy 2031 Files, the contents of a disk file are transferred to PET's programmable memory and then transferred back to a different diskette, all at machine language speed. The user simply enters the filename and switches diskettes at the appropriate time.

Operation of the program may be understood through comparison with its BASIC counterpart. In *OPEN ERROR CHANNEL* a channel is prepared for input of disk error messages.

```
100 OPEN 1,8,15
```

The name of the file to be copied is entered in *OPEN FILE FOR READ*. The filename is then appended with ",P,R" (or ",S,R") and the file is opened for reading.

```
110 PRINT:INPUT "FILENAME";FL$
120 OPEN 2,8,2,FL$+",P,R"
130 GOSUB 500
```

READ FILE loads the file into the memory of the PET (normally occupied by a BASIC program). The end of the file is detected via a change in the status word ST.

```
140 XFR=TP
150 GET#2,C$
160 IF C$="" THEN C=0:GOTO 180
170 C=ASC(C$)
180 POKE XFR,C
190 IF ST THEN 210
200 XFR=XFR+1:GOTO 150
210 EOF=XFR
220 CLOSE 2
```

At this point the diskettes are switched and a file is opened for writing in *OPEN FILE FOR WRITE*

```
230 PRINT:PRINT"SWITCH DISKETTES,"
240 PRINT"THEN HIT RETURN."
250 GETC$:IF C$="" THEN 250
260 OPEN 2,8,2,FL$+",P,W"
270 GOSUB 500
```

The reverse process is carried out in *WRITE FILE*. The file contents are transferred byte by byte until the end of the file is indicated.

```
280 XFR=TP
290 C=PEEK(XFR)
300 C$=CHR$(C)
310 PRINT#2,C$;
320 IF XFR<EOF THEN XFR=XFR+1:GOTO 290
```

When the file is completely transferred, all files are closed in *EXIT*.

```
330 CLOSE 2:CLOSE 1:END
```

The subroutine *DERROR* allows disk errors to be detected and displayed.

```
500 INPUT#1,EN$
510 IF EN$="00" THEN RETURN
520 PRINT:PRINT"DISK ERROR ";EN$;"!"
```

For the BASIC equivalent to work correctly a safe storage space must be allocated in memory for the file.

```
10 POKE 53,8:CLR
20 TP=PEEK(53)*256+PEEK(52)
```

Copy 2031 Files has been assembled to reside in the first and second cassette buffers of a BASIC 4.0 PET. [*The BASIC loaders provided (Programs 1 and 2) are for 4.0 and Upgrade BASIC.*] The program might run on a VIC-20 if the system variables and subroutine calls can be supplied by a knowledgeable VIC owner. Incidentally, the program will also work with the CBM 4040 Dual Disk Drive.

ECLECTIC SYSTEMS CORPORATION
Order TOLL FREE 1-800-527-3135

ECLECTIC SYSTEMS

P.O. Box 1166 • 16260 Midway Road
Addison, TX 75001 • (214) 661-1370

ANNOUNCES

EM© for the SuperPET

EM© is ANSI standard MUMPS

Make your SuperPET Super-Load in EM© and Dramatically Improve your Programming Productivity

EM© is more than just a programming language. It is a well integrated data management system combining with one syntax what other operating systems would call 1) an application programming language; 2) a job control language; 3) a linkeage editor; 4) a database management system; and 5) a communications monitor.

PROGRAM MANAGEMENT:
EM© provides all programming management facilities needed to manage programs and program files. Programs can be created, edited, cataloged and debugged from within EM©. Programs can be as large as disk capacity. A resident algorithm rids memory of least frequently used variables and program modules so that what you need off-disk normally resides in memory.

STRING POWER:
EM© makes string handling easy with its extensive set of string operations and functions. Variable length strings can be used routinely without the obstacles presented by most other programming languages.

PATTERN MATCHING:
EM© can "filter" user input with a useful pattern matching that will result in fewer user or device errors. For example: dates, zip codes and names can be tested for validity with a single statement.

GLOBALS:
EM© obviates the need for traditional read and write operations on secondary storage devices by allowing data elements to be directly referenced as a set of subscripts; all the details of file organization and retrieval are handled by the system.

TIMING:
EM© enables a programmer to associate timing constraints with several operations. This feature allows testing for terminal malfunctions as well as prompting users in time-critical dialogue.

DATA BASE MANAGEMENT:
Sorts and merges are not necessary as EM© automatically stores data in a dynamically allocated balanced tree structure. Random access to any data item requires at most three disk reads.

EM© UNMATCHED IN PROGRAMMING PRODUCTIVITY:
System houses that program in EM© (MUMPS) find that their costs are lower than those of their competitors using other languages. Fewer lines of code are necessary per application. Dimension statements are not required. Subscripts may be alpha, numeric or any legal string. Data types need not be defined and can change freely throughout as EM© can recognize when it is dealing with alpha, numeric, integer or floating-point data types. EM© gives the professional programmer a full set of software tools designed for real-life tasks and problems he consistently encounters in the production and maintenance of application software. EM© adheres rigidly to ANSI MUMPS standards, which make it transportable to larger processors manufactured by DEC, TANDOM, DATA GENERAL, HARRIS and others. Additionally EM© gives the less-experienced programmer the tools to do a professional job on formidable programming applications.

You may order EM© or SuperPET by calling ECLECTIC SYSTEMS toll-free at 1-800-527-3135 from 10 AM to 4 PM CDT Monday through Friday, or you can order by mail using the form below. Texas residents call 1-214-661-1370.

ECLECTIC SYSTEMS CORPORATION
P.O. Box 1166, 16260 Midway Road, Addison, Texas 75001

Here's my order for EM©@ $299 plus $3.75 for shipping and handling (UPS surface unless specified otherwise). Residents of Texas, Louisiana, Oklahoma City and Tulsa, Oklahoma must add applicable taxes.

☐ My certified check or money order is enclosed.

☐ Please charge my VISA # _____ or
MasterCard # _____ Expiration date _____

Name _____ Signature _____

Address _____

City _____ State _____ Zip _____

The program resides happily in the cassette buffers unless 1) the cassette drive is accessed, or 2) the advanced DISK BASIC commands in BASIC 4.0 are used. For example, entering DIRECTORY D0 would cause part of the program in the second cassette buffer to be overwritten. If this creates a problem, assemble the program elsewhere. Using DOS Wedge commands will not harm the program though.

As shown, Copy 2031 Files will copy program files (BASIC programs, *WordPro* files, MAE files, ...). This is controlled by the appendix ",P,R" stored in *STRING TABLE* in reverse order. In order to copy sequential files (Data files, *PaperMate* files, ASM/TED files, ...) the *P* in the appendix should be replaced with an *S*. This change may be accomplished before running the program (SYS 634) by changing the byte with a POKE (POKE 952,80 for program files and POKE 952,83 for sequential files). For copying a large number of files, you may consider changing JMP READY to JMP BEGIN.

Program 1. 4.0 Version

```
500 FOR ADRES=634TO954:READ DATTA:POKE
    ADRES,DATTA:NEXT ADRES
634 DATA 169, 1, 133, 210, 32, 226
640 DATA 242, 169, 8, 133, 212, 169
646 DATA 15, 133, 211, 169, 0, 133
652 DATA 209, 32, 99, 245, 160, 3
658 DATA 169, 115, 32, 29, 187, 32
664 DATA 226, 180, 169, 0, 133, 218
670 DATA 169, 2, 133, 219, 160, 255
676 DATA 200, 177, 218, 208, 251, 162
682 DATA 4, 189, 181, 3, 145, 218
688 DATA 200, 202, 208, 247, 132, 209
694 DATA 169, 2, 133, 210, 32, 226
700 DATA 242, 169, 8, 133, 212, 169
706 DATA 2, 133, 211, 32, 99, 245
712 DATA 32, 77, 3, 162, 2, 32
718 DATA 198, 255, 169, 4, 133, 1
724 DATA 169, 3, 133, 0, 160, 0
730 DATA 32, 21, 242, 145, 0, 166
736 DATA 150, 208, 7, 200, 208, 244
742 DATA 230, 1, 208, 240, 132, 5
748 DATA 165, 1, 133, 6, 169, 2
754 DATA 32, 226, 242, 32, 204, 255
760 DATA 160, 3, 169, 127, 32, 29
766 DATA 187, 32, 228, 255, 240, 251
772 DATA 164, 209, 136, 169, 87, 145
778 DATA 218, 32, 99, 245, 32, 77
784 DATA 3, 162, 2, 32, 201, 255
790 DATA 169, 4, 133, 1, 169, 3
796 DATA 133, 0, 160, 0, 177, 0
802 DATA 32, 102, 242, 165, 1, 197
808 DATA 6, 208, 4, 196, 5, 240
814 DATA 14, 200, 208, 238, 230, 1
820 DATA 208, 234, 160, 3, 169, 164
826 DATA 32, 29, 187, 169, 2, 32
832 DATA 226, 242, 169, 1, 32, 226
838 DATA 242, 32, 204, 255, 76, 255
844 DATA 179, 162, 1, 32, 198, 255
850 DATA 32, 21, 242, 141, 176, 3
856 DATA 32, 21, 242, 141, 177, 3
862 DATA 32, 204, 255, 173, 176, 3
868 DATA 201, 48, 208, 206, 173, 177
874 DATA 3, 201, 48, 208, 199, 32
880 DATA 204, 255, 96, 13, 70, 73
886 DATA 76, 69, 78, 65, 77, 69
892 DATA 63, 32, 0, 13, 83, 87
898 DATA 73, 84, 67, 72, 32, 68
904 DATA 73, 83, 75, 69, 84, 84
910 DATA 69, 83, 44, 13, 84, 72
916 DATA 69, 78, 32, 72, 73, 84
922 DATA 32, 82, 69, 84, 85, 82
928 DATA 78, 46, 13, 0, 13, 68
934 DATA 73, 83, 75, 32, 69, 82
940 DATA 82, 79, 82, 32, 0, 0
946 DATA 33, 13, 0, 0, 82, 44
952 DATA 80, 44, 246, 230, 1, 76
```

Program 2. Upgrade ROM Version

Change these lines in Program 2.

```
634 DATA 169, 1, 133, 210, 32, 174
652 DATA 209, 32, 36, 245, 160, 3
658 DATA 169, 115, 32, 28, 202, 32
664 DATA 111, 196, 169, 0, 133, 218
694 DATA 169, 2, 133, 210, 32, 174
706 DATA 2, 133, 211, 32, 36, 245
730 DATA 32, 225, 241, 145, 0, 166
754 DATA 32, 174, 242, 32, 204, 255
760 DATA 160, 3, 169, 127, 32, 28
766 DATA 202, 32, 228, 255, 240, 251
778 DATA 218, 32, 36, 245, 32, 77
802 DATA 32, 50, 242, 165, 1, 197
826 DATA 32, 28, 202, 169, 2, 32
832 DATA 174, 242, 169, 1, 32, 174
838 DATA 242, 32, 204, 255, 76, 137
844 DATA 195, 162, 1, 32, 198, 255
850 DATA 32, 225, 241, 141, 176, 3
856 DATA 32, 225, 241, 141, 177, 3
```

BATTERIES INCLUDED

village by the grange, 71 mccaul st. (f6) toronto m5t 2x1 telephone 596-1405

ARBITER 1.4 MULTI-USER DISK SYSTEM FOR COMMODORE 4.0 COMPUTERS

OVER THREE HUNDRED IN USE ACROSS ONTARIO

Since September 1981 **BATTERIES INCLUDED** has been installing the ARBITER system in classrooms of Commodore BASIC 4.0 computers. The computers are connected to CBM Disk Drives and printers. All users have access to all disk drives and printers plus a host of commands to make this system configuration really usable!

THE ARBITER 1.4 SYSTEM IS READY TO GO!

FEATURES

1) Easy installation.
2) Uses no RAM or Utility Sockets.
3) Up to 32 computers in one system.
4) System self initializes on power up.
5) Operation is completely transparent to the user.
6) Extended commands allow a friendly multi-user environment.
7) System design virtually eliminates interleaved printer output.

$150.00

SPECIAL COMMANDS

@S – Allows students to protect files with a five character password. A three character user ID is forced into the file name.

@L – Allows the students to load protected files if the password code is known.

LISTC – Used to produce program listings with a Commodore printer. Clumsy OPEN, CMD, LIST, PRINT#, CLOSE sequence not needed. It overcomes the listing problems found on other multi-user hardware systems.

LISTP – Used to get program listings on systems which have an ASCII printer. The cursor control characters are expanded and displayed in brackets. e.g. <home>

ALL FILE TYPES ARE SUPPORTED – During relative or sequential file access a delay has been built in so the computer will retain control of the system until the file is closed.

TEACHER UTILITY – A utility is supplied on disk to allow the teacher to produce a hardcopy listing and output from any of the protected or unprotected files selected. Once the files are chosen from the disk directory the teacher may do other tasks while the job is completed.

IF YOUR CLASSROOM WAS DESIGNED TO TEACH COMPUTER LITERACY OR STRUCTURED BASIC THEN THIS SYSTEM WAS DESIGNED FOR YOU.

SWARM-100

JUST PLUG IT IN

• No soldering • No messy wires

SOFTWARE SELECTABLE

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

$150.00

Master Charge and VISA accepted.

For 24 Pin ROM Machines Only.

BATTERIES INCLUDED

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

Hit one key and a whole BASIC word is printed for you on the screen. This programming shorthand is adapted to the VIC from a PET program, "Keyword." There is also an example of how to go about converting PET machine language to work on the VIC.

VIC-Key

Thomas Henry
Mankato, MN

VIC-Key is a utility for the Commodore VIC-20 computer written in machine language. Like Charles Brannon's Keyword (**COMPUTE!**, August, 1981, #15), it lets one keystroke do a lot of work. For example, hit SHIFT-A and the statement "ASC" shoots out. SHIFT-B gives "STEP", SHIFT-C gives "CHR$" and so on. In short, 26 of the most common BASIC statements now have one-keystroke equivalents. And, unlike the standard two keystroke abbreviations that Commodore provides (for example, "I,SHIFT-N" is equivalent to "INPUT"), this version spells out the entire phrase instantly. Now when you hit SHIFT-I the entire word "INPUT" dashes out on the screen. As you can tell, this is a real time saver.

An Important Addition

Although VIC-Key is based on the article mentioned above, one important addition has been made to the program. Using capital letters for the various keywords is a great idea since the VIC-20 doesn't like to see shifted letters in a BASIC statement anyway. However, there is one time when you want a capital letter to really be a capital letter (not a keyword), and that's when you're inside quotes. For example, line 10 of a program may read:

```
10 PRINT "I AM YOUR QUIZ-MASTER. HIT RETURN."
```

You clearly want the capital "I" and the capital "H" to be just that, ordinary capitals. Well, VIC-Key has been written in such a way that it keeps track of whether you're inside quotes or outside quotes and adjusts accordingly.

VIC-Key is able to decide if you're in quotes or not by inspecting location $D4 in the zero page. If this location contains a zero then the quotes are OFF, and it's safe to perform the key-statement transformation. If this location contains a one, then the quotes are ON and the transformation must be skipped. (If you are a PET user, you may want to modify the original Keyword program mentioned above to also keep track of quotes. The quotes flag location for the PET is $CD).

Modifying PET Machine Language To Run On VIC

Changing Brannon's Keyword program into VIC-Key was mostly a matter of disassembling the original, finding all the zero page locations called out, finding their equivalents in the VIC-20 memory map and changing them accordingly. However, there was one tricky point that almost made me give it up as hopeless. Since the VIC-20 is a relatively new computer, very little has been published on its BASIC in ROM. In short, I couldn't figure out where the needed Table of BASIC Keywords was located. All I knew was that it was somewhere between $C000 and $FFFF!

After just about giving up, I hit upon the idea of inspecting the VIC's ROMs with my CBM 8032. First I transferred the VIC's ROMs to DATA statements 500 bytes at a time using H. Linder's Automatic Data Statement program (**COMPUTE!**, October, 1981, #17) (modified for use with the VIC-20). After doing this I loaded the tape just made into my CBM 8032. I did this with the help of L. Jordan's "Train Your PET to Run VIC Programs" (**COMPUTE!**, October, 1981, #17). In effect, I recreated the VIC ROMs in my CBM 8032's RAM. I then disassembled this "pseudo-ROM" using Cochrane's Micromon (**COMPUTE!**, January, 1982, #20), an extended monitor, and eventually found the table I needed. To save yourself this work, you may want to make a note that the start of the Table of BASIC Keywords is $C09E.

To use the program, follow these steps:

1) Enter the program.

2) After inspecting it for accuracy, SAVE it to tape.

3) RUN it, then SYS7501. VIC-Key is now activated.

4) Give it a try. The table shows the keyword equivalents. Confirm that VIC-Key knows whether you're in quote mode or not.

5) If you want to deactivate the program, simply SYS7501 again. VIC-Key is now dormant, but not wiped out from memory. You can reactivate it again at any time by doing another SYS7501.

6) Since the top of memory pointers has been lowered, VIC-Key is safe from BASIC program interference. In addition, typing NEW will not affect it. However, hitting the STOP/RESTORE key combination will wipe it out completely.

The keyword equivalents in the table are very easy to memorize if you note the following:

1) Most commands are simply alphabetical.

WUNDERWARE® PRESENTS MORE GAMES FOR YOUR VIC-20

These games and many more! Send 50¢ (refundable with order) for catalog and free program listing.

THE MAD PAINTER
This game is a little unique and a lot of fun. You control a paint brush, moving it around a colorful maze. Your job is to paint the entire maze. This is not as easy as it sounds, because in the maze with you are two voracious Bristle Biters (they love paint brushes). Occasionally you will receive a visit from an Invisible Stomper who leaves footprints in your fresh paint. Requires joystick.

$9.95

GALAXY INVASION
Deeper and deeper you go into the hostile alien galaxy. Game points by maneuvering your ship to rescue men as they drift by. Deep space fuel stations so you can continue your trek, asteroids and space mines which you must avoid at all cost, are all a part of this fun and exciting one player game. Joystick required.

$9.95

SNAKE!
A fast and fun action game for one player. You're a big snake roaming around the screen. Mice, rabbits, eggs, and feet appear at random. Your mission in life is to bite these targets. You have to be quick—the targets don't stay for long. The main problem is; you always seem to be running into the wall or into yourself (the longer you play, the longer, and harder it gets to avoid your tail)! Snake keeps high score and requires a joystick.

$9.95

• Price includes Postage & Handling. • Foreign orders and COD's; Please add $3.00.
• Catalog is included with order. • Prices are subject to change without notice.
• Send check or money order to:

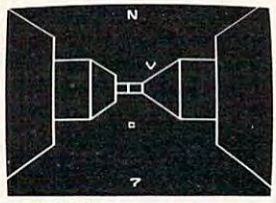
wunderware
P.O. Box 1287, Jacksonville, Oregon 97530

VIC-20 is a registered trademark of Commodore Business Machines.

VIC-20® VIC-20®

VICTORY SOFTWARE INC.®

WE SELL FUN!™



TREASURES OF THE BAT CAVE **$19.95**
Battle the vampire bats as you search their cave for gold bullion. Fast, real time action will keep you playing for hours. Of course, you are in a different cave every time you play.

COSMIC DEBRIS **$14.95**
This highly addictive arcade type game will keep you battling the aliens for days.

GRAVE ROBBERS **$13.95**
Introducing the first GRAPHIC ADVENTURE ever available on the VIC-20! With realistic audio-visual effects, you explore an old deserted graveyard and actually see the perils that lie beyond.

NIGHT RIDER **$12.95**
High speed night time driving simulator.

STREET SWEEPERS **$14.95**
Gobble up all of the dots in the maze before the ensuing nemesis gets you. The maze is different every time, and if you succeed in getting all the dots, you get progressively harder mazes to complete as your skills increase. Does this sound like Pac Man? It isn't! Highly recommended and extremely addicting.

Send for free catalog.
All programs fit in the standard VIC memory, and can be controlled from the keyboard.
All programs on cassette tape.
Ordering—Please add $1.50 postage and handling per order.
PA residents please add 6% sales tax.

VICTORY SOFTWARE INC.
2027-A S. J. Russell Circle, Elkins Park, PA 19117
(215) 576-5625

VIC-20® VIC-20®

VIC 20* PROGRAMS

TOTL. TEXT 1.0 $25.00
Full capability word processing:
- Margin and spacing control
- Centered title lines
- Indentation and tabs
- Upper and lower case and graphics
- Full screen editing
- Scroll up and down
- Long documents created using tape files

TOTL. TEXT 1.5 $35.00
ALL TOTL. TEXT 1.0 features plus:
- Heading lines (up to 4)
- Footing line
- Footnotes
- Keyboard input
- Additional working memory

RESEARCH ASSISTANT 1.0 $35.00
For authors, students, researchers:
- Compile reference information
- Create cross reference lists by keyword
- Save bibliographic data

TOTL. LABEL 1.0 $20.00
Flexible mailing list and label program
- User defines label size
- Optional, non-printing data line
- Print all or just selected labels
- Easy editing
- Automatically alphabetized

All programs designed to run on the VIC 20 with 8K expander, cassette tape, and printer.

Shipping included. Send check or money order to: TOTL Software, P.O. Box 4742, Walnut Creek, Ca 94596

(Calif. residents add 6% sales tax. $3.00 charge for C.O.D.)

Dealer Inquiries Welcome

*VIC 20 is a trademark of Commodore Business Machines, Inc.

PET/CBM OWNERS

WALLBANGER - Blast your way through the dodge'm, blast'm, and attack modes. If you destroy the bouncing balls before they destroy you, the walls close in for the next round. Wallbanger is written in machine language, has great sound, and encourages complex strategies.

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

MILLIPEDE - Exterminate the oncoming millipedes and fleas as they descend through the mushroom patch. Blast giant bouncing spiders before they pounce on you. Shoot a millipede in the body and suddenly two millipedes descend toward your ship. Millipede is written in machine language, has excellent graphics, and great sound.

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

WARNING! These games cause high panic levels!

PET/CBM OWNERS

For example, SHIFT-A equals "ASC", SHIFT-C equals "CHR$", etc.

2) SHIFT-W, X and Y are DATA type commands, i.e., "DATA", "READ", "RESTORE".

3) For SHIFT-H think "halt" (equals STOP).

4) SHIFT-P is POKE and, one letter later, (SHIFT-Q) is PEEK.

VIC-Key consumes 174 bytes of memory, which leaves plenty left over for BASIC programming even with the limited memory of a stock VIC-20. With the new quote mode detector, VIC-Key is so easy to use that I think you'll agree that it will more than "pay" for the little memory that it uses. So rest those tired hands; let VIC-Key do the typing.

Table of BASIC Keywords

A	ASC	O	OPEN
B	STEP	P	POKE
C	CHR$	Q	PEEK
D	DIM	R	RIGHT$
E	END	S	STR$
F	GET	T	TAB(
H	STOP	U	USR
I	INPUT	V	VAL
J	GOTO	W	DATA
K	GOSUB	X	READ
L	LEFT$	Y	RESTORE
M	MID$	Z	SYS
N	NEXT		

Program.

```
100 POKE55,77:POKE56,29
110 PRINT"WAIT..."
120 FORI=7501TO7679
130 READA:POKEI,A:X=X+A
140 NEXT
150 PRINT"SYS7501 TO ACTIVATE.";
160 IF X <> 22351 THEN PRINT" THERE IS AN E
    RROR IN YOUR TYPING OF THE DATA LI
    NES"
170 NEW
180 DATA120,173,20,3,72,173,21,3,72,173,116
    ,29,208,2,169,118
190 DATA141,20,3,173,117,29,208,2,169,29,14
    1,21,3,104,141,117
200 DATA29,104,141,116,29,88,96,0,0,72,138,
    72,152,72,165,215
210 DATA72,165,212,240,4,104,76,221,29,104,
    201,193,144,82,201,219
220 DATA176,78,56,233,193,170,189,229,29,16
    2,0,134,198,170,160,158
230 DATA132,34,160,192,132,35,160,0,10,240,
    16,202,16,12,230,34
240 DATA208,2,230,35,177,34,16,246,48,241,2
    00,177,34,48,17,8
250 DATA142,255,29,230,198,166,198,157,119,
    2,174,255,29,40,208,234
260 DATA230,198,166,198,41,127,157,119,2,16
    9,20,141,119,2,230,198
270 DATA104,168,104,170,104,76,191,234,198,
    169,199,134,128,129,161,144
280 DATA133,137,141,200,202,130,159,151,194
    ,201,196,163,183,197,131,135
290 DATA140,158,127
```

References

1) C. Brannon, "Keyword," **COMPUTE!** *#15, August 1981, pp. 120, 122.*

2) H. Linder, "Automatic DATA Statements for CBM and Atari," **COMPUTE!** *#17, October 1981, p. 22.*

3) L. Jordan, "Train Your PET to Run VIC Programs," **COMPUTE!** *#17, October 1981, p. 138.*

4) R. A. Cochrane, "MICROMON: An Enhanced Machine Language Monitor," **COMPUTE!** *#20, January 1982, pp. 160-173.*

COMPUTE!'s Listing Conventions

Many of the programs which are listed in **COMPUTE!** use special keys (cursor control keys, color keys, etc.). To make it easy to tell *exactly* what should be typed in when copying a program into the computer, we have established the following listing conventions.

For The Atari

In order to make special characters, inverse video, and cursor characters easy to type in, **COMPUTE!** magazine's Atari listing conventions are used in all the program listings in this magazine.

Please refer to the following tables and explanations if you come across an unusual symbol in a program listing.

Atari Conventions

Characters in inverse video will appear like: INVERSE VIDEO
Enter these characters with the Atari logo key, {♣}.

When you see	Type	See	
{CLEAR}	ESC SHIFT <	↖	Clear Screen
{UP}	ESC CTRL -	↑	Cursor Up
{DOWN}	ESC CTRL =	↓	Cursor Down
{LEFT}	ESC CTRL +	←	Cursor Left
{RIGHT}	ESC CTRL *	→	Cursor Right
{BACK S}	ESC DELETE	◂	Backspace
{DELETE}	ESC CTRL DELETE	◧	Delete character
{INSERT}	ESC CTRL INSERT	◨	Insert character
{DEL LINE}	ESC SHIFT DELETE	▣	Delete line
{INS LINE}	ESC SHIFT INSERT	▣	Insert line
{TAB}	ESC TAB	▸	TAB key
{CLR TAB}	ESC CTRL TAB	◪	Clear tab
{SET TAB}	ESC SHIFT TAB	◪	Set tab stop
{BELL}	ESC CTRL 2	◱	Ring buzzer
{ESC}	ESC ESC	◖	ESCape key

Graphics characters, such as CTRL-T, the ball character ● will appear as the "normal" letter enclosed in braces, e.g. {T}.

A series of identical control characters, such as 10 spaces, three cursor-lefts, or 20 CTRL-R's, will appear as {10 SPACES}, {3 LEFT}, {20 R}, etc. If the character in braces is in inverse video, that character or characters should be entered with the Atari logo key. For example, {■} means to enter a reverse-field heart with CTRL-comma, {5▣} means to enter five inverse-video CTRL-U's.

For PET/CBM/VIC

Generally, any PET/CBM/VIC program listings will contain bracketed words which spell out any special characters: {DOWN} would mean to press the cursor-down key; {3DOWN} would mean to press the cursor-down key three times.

To indicate that a key should be *shifted* (hold down the SHIFT key while pressing the other key), the key would be underlined in our listing. For example, S would mean to type the S key while holding the shift key. This would result in the "heart" graphics symbol appearing on your screen.

Sometimes in a program listing, especially within quoted text when a line runs over into the next line, it is difficult to tell where the first line ends. How many times should you type the SPACE bar? In our convention, when a line breaks in this way, the ~ symbol shows exactly where it broke. For example:

```
100 PRINT "TO START THE GAME  ~
    YOU MAY HIT ANY OF THE KEYS
    ON YOUR KEYBOARD."
```

shows that the program's author intended for you to type two spaces after the word *GAME*.

For The Apple

Programs listed as "Microsoft" are written for the PET/CBM, Apple, OSI, etc. Although the programs are general in nature, you may need to make a few changes for them to run correctly on your Apple. Microsoft BASIC programs written for the PET/CBM sometimes contain special cursor control characters. The following table shows equivalent Apple words. Notice that these Apple commands are *outside* quotations (and even separate from a PRINT statement). PRINT"[RVS]YOU WON" becomes INVERSE: PRINT"YOU WON":NORMAL

[CLEAR[(Clear Screen) HOME

[DOWN] (Cursor down)
Apple II +: Call -922
POKE 37,PEEK(37)+(PEEK(37)<23)

[UP] (Cursor up)
POKE 37,PEEK(37)-(PEEK(37)>0))

[LEFT] (Cursor left) PRINT CHR$(8);

[RIGHT] (Cursor right)
PRINT CHR$(21)

[RVS] (Inverse video on. Turns off automatically after a carriage return. To be safe, turn off inverse video after the print statement with NORMAL unless the PRINT statement ends with a semicolon.)
INVERSE

[OFF] (Inverse video off) NORMAL

Shifted characters can represent either graphics characters or uppercase letters. If within text, just use the non-shifted character, otherwise substitute a space. Some "generalized" programs contain a POKE such as POKE 59468,14. Omit these from the program when typing it in. One final note: you will probably want to insert a question mark or colon within an INPUT prompt. PET/CBM and many other BASICs automatically print a question mark:

INPUT "WHAT IS YOUR NAME";N$
becomes
INPUT "WHAT IS YOUR NAME?";N$

All Commodore Machines

Clear Screen	{CLEAR}	**Cursor Left**	{LEFT}
Home Cursor	{HOME}	**Insert Character**	{INST}
Cursor Up	{UP}	**Delete Character**	{DEL}
Cursor Down	{DOWN}	**Reverse Field On**	{RVS}
Cursor Right	{RIGHT}	**Reverse Field Off**	{OFF}

VIC Conventions

Set Color To Black	{BLK}	**Function Two**	{F2}
Set Color To White	{WHT}	**Function Three**	{F3}
Set Color To Red	{RED}	**Function Four**	{F4}
Set Color To Cyan	{CYN}	**Function Five**	{F5}
Set Color To Purple	{PUR}	**Function Six**	{F6}
Set Color To Green	{GRN}	**Function Seven**	{F7}
Set Color To Blue	{BLU}	**Function Eight**	{F8}
Set Color To Yellow	{YEL}	**Any Non-implemented Function**	{NIM}
Function One	{F1}		

8032/Fat 40 Conventions

Set Window Top	{SET TOP}	**Erase To Beginning**	{ERASE BEG}
Set Window Bottom	{SET BOT}	**Erase To End**	{ERASE END}
Scroll Up	{SCR UP}	**Toggle Tab**	{TGL TAB}
Scroll Down	{SCR DOWN}	**Tab**	{TAB}
Insert Line	{INST LINE}	**Escape Key**	{ESC}
Delete Line	{DEL LINE}		

A Monthly Feature

You often need to know on which screen you defined a particular word. If your system supports, say, 300 screens, it's tiresome to index through them, looking for something. This search routine combines machine language with FORTH and is a fast, efficient way to find "lost" definitions.

If you have come up with some interesting FORTH applications or techniques, send them in to The FORTH Page, **COMPUTE!** *Magazine, P.O. Box 5406, Greensboro, NC, 27403 and share them with the rest of us.*

The FORTH Page

Speed Search

Richard Mansfield, Assistant Editor

These three screens compile the word HUNT, which will locate anything on disk. Assume that you are writing a game and you remember that somewhere on your disk you defined RND to provide a random number. Unfortunately, you cannot now recall exactly where RND is located, but you think it might be between screens 50 and 70. All too often, you must laboriously list each screen and read through it, looking for that "missing" definition.

This fast search routine will fly effortlessly through your disk, reporting the screen and line number where it finds matches. To find RND, you first introduce the target by typing " RND" and then type:

```
50 70 HUNT
```

and each screen number is printed as it is checked. Any line containing a match is printed out beneath the screen number. To hunt only for the actual definition of the word, use the colon as well:

```
" :RND"
```

FORTH Compatibility

Ideally, FORTH would be system independent: it wouldn't matter what computer you are using, you could type in a screen from **COMPUTE!** and it would work on your machine as printed. In practice, however, there always seem to be a few minor adjustments to make to a FORTH program of any significant length before it will work for your particular setup.

This search routine was developed on "FORTH For PET" which includes a word, ?TERMINAL, which checks to see if the PET STOP key is pressed. The user then can exit a loop from the keyboard as illustrated in line 13 of Screen 112. HUNT contains the modifications necessary to make it work on the APX figFORTH for the Atari. ?TERMINAL is not available on the APX version of FORTH.

Line three, Screen 110 is an Atari specific definition for ?TERMINAL. It reads the console switches and returns a three-bit result between one and seven. Each bit (1, 2, 4) represents either the START, SELECT, or OPTION keys. Any combination of these keys could be tested by using AND, but here we are merely seeing if any are pressed and, if so, we LEAVE the HUNT.

A second, minor, variation between these FORTHs requires the substitution of IFEQ for 0= on line ten, Screen 111, within the machine language character comparison. There is a major difference, on the other hand, in the way that Atari handles BLOCK.

BLOCK Modifications For The Atari

On the PET, the word BLOCK (n1 — addr) returns the memory address of the start of a 1024-byte block. On the Atari, the word BLOCK returns the address of a 132-byte block and the value of n1 is a disk sector number (not a screen number). The Atari block is 128 bytes plus four additional bytes which are perhaps for sector management.

To simulate the PET method of handling BLOCK, line one of screen 110 defines the word BLOK. It multiplies the screen number by eight to get the correct sector and then reads in eight sectors. The address of the first sector is then left on the stack. The following sectors are in memory as required along with the four-byte tags. If you want to try to eliminate the four tag bytes, beware of damage to disk management caused by any subsequent FLUSHes.

The translation between PET and Atari FORTH is not perfect. Because of those tag bytes, a false match will be reported now and then in the Atari version. What's more, the original PET (80 column) version included a superior alternative to .LINE. When a match was found, HUNT listed the screen and flashed the target word on and off while ringing the bell. Calculating the exact video screen position of the target word is, of course, especially machine-specific, but it is impressive to watch. It requires the following modifications to MARKSTRING and the addition of the word WHITEIT:

```
: WHITEIT OVER OVER Ø DO DUP I + 8Ø TOGGLE LOOP
    7 EMIT DROP ;
: MARKSTRING ( SCR# ADDR --- SCR# )
```

```
    OVER DUP SCR @ = 0=
     IF DUP LIST CR ENDIF
    BLOCK - 40 /MOD 1+ 50 * 4 + + 8050 + PAD C@
      BEGIN WHITEIT WHITEIT GET UNTIL
    DROP DROP ;
```

HUNT

```
SCR # 110
 0 FORTH DEFINITIONS HEX      0 VARIABLE 1STCHAR
 1 : BLOK 8 * DUP BLOCK SWAP DUP 7 + SWAP DO I BLOCK
  DROP LOOP ;
 2
 3 : ?TERMINAL -2FE1 C@ 7 XOR ;  ( READS ATARI CONSO
  LE SWITCHES)
 4 ( BLOK AND ?TERMINAL ARE FOR ATARI USERS ONLY )
 5 : MATCH ( ADDR1 ADDR2 N --- F )
 6   -DUP IF OVER + SWAP
 7          DO DUP C@ I C@ -
 8              IF 0= LEAVE ELSE 1+ THEN
 9          LOOP
10      ELSE DROP 0= THEN ;
11
12  : CHECKIT PAD 1+ PAD C@ MATCH ;   ( ADDR --- F )
13
14  : HEADER CR ." SEARCHING FOR " 22 EMIT SPACE PAD
15    1+  PAD C@ TYPE 22 EMIT  SPACE  ." ON SCR # ..
   ." ; -->


SCR # 111
 0
 1  : MARKSTRING  ( SCR# ADDR --- SCR# )
 2     OVER BLOCK - C/L / CR DUP . SPACE  ( ATARI,
   USE BLOK)
 3   OVER .LINE CR ;
 4
 5
 6
 7
 8  CODE ?CHAR  ( ADDR --- ADDR F )
 9    1 # LDA,  SETUP JSR,
10    N )Y LDA,  1STCHAR CMP,  0=  ( ATARI, USE IFEQ
  , NOT 0=)
11      IF,  1 # LDA, PUSH0A JMP,  THEN,
12      0 # LDA,  PUSH0A JMP,            FORTH
13  -->
14
15


SCR # 112
 0 : ONEBLK          ( SCR# ADDR ---
     )
 1    DUP 400 + SWAP  ( ATARI, USE 410, NOT 400)
 2      DO I ?CHAR
 3         IF I CHECKIT
 4            IF I MARKSTRING ENDIF
 5         ENDIF
 6      LOOP DROP ;
 7
 8 : "  22 WORD HERE DUP C@ 1+ PAD SWAP CMOVE ;
 9
10 : HUNT   ( SCR#1  SCR#2  ---   ;WITH STRING AT P
  AD )
11   0 SCR !  PAD 1+ C@ 1STCHAR !  HEADER  1+  SWAP
12     DO I DUP DUP CR 2 SPACES . BLOCK ONEBLK ( ATA
  RI, USE BLOK)
13       ?TERMINAL IF LEAVE ENDIF
14     LOOP CR CR ." END SEARCH" CR ;
15   DECIMAL  ;S
```

Turn To The Future With COMPUTE! Publications

The Beginner's Guide To Buying A Personal Computer

A Novice's handbook of useful, helpful information designed to teach you the basics of evaluating and selecting a personal computer. Written in plain English for the interested beginner. Complete with personal computer specification charts and buyer's guide. Applicable to home, educational, and small business buyers. ISBN 0-942386-03-5. Paperback. $3.95.

COMPUTE!'s First Book Of Atari

192 pages of useful, informative applications and programs from **COMPUTE!** magazine issues now out of print. Includes previously unpublished information including Memory Map. Contents include such articles and programs as "Adding A Voice Track to Atari Programs," "Designing Your Own Atari Graphics Modes," and "Inside Atari BASIC." Spiral bound for ease of access to listings. For Beginner level to Advanced Atari users. ISBN 0-942386-00-0. Paperback. $12.95.

Inside Atari DOS

From the authors of the Atari Disk Operating System, an exciting step-by-step guide to the DOS software. Complete with listings of commented source code and detailed explanations of each module of code. Author: Bill Wilkinson, Optimized Systems Software, Inc. Spiral bound for ease of access to listings. For Intermediate to Advanced Atari Users. ISBN 0-942386-02-7. Paperback. $19.95

COMPUTE!'s First Book Of PET/CBM

256 pages of Commodore PET and CBM articles from **COMPUTE!** magazine issues now in print. Includes such classic articles and programs as "Feed Your PET Some Applesoft," "Disk Lister: A Disk Cataloging Program," and "Cross Reference For The PET." Spiral bound. ISBN 0-942386-01-9. Paperback. $12.95

COMPUTE! Books invites dealer inquiries. Call the Toll Free Number below for Dealer Information.

COMPUTE! Magazine

A Monthly encyclopedia of informative applications articles and programs. **COMPUTE!** features articles, programs, and columns covering the spectrum of home and educational computing. Monthly reviews, complete BASIC and machine language listings of games, utilities, applications such as "Programming Your Home Insurance Inventory," "Real Estate Investment Analysis," "Telecommunications: How To Use A Modem," and much more. Written for children and parents, educators, novices to advanced programmers. Principal editorial coverage is Atari, Apple, Commodore PET/CBM, and VIC-20. Editorial coverage is expanding to include TI-99/4A, Sinclair ZX-81, and Radio Shack Color Computer. Latest issue: 224 pages.

Mail to: **COMPUTE!** Publications, P.O. Box 5406, Greensboro, NC 27403 USA

COMPUTE! MAGAZINE

My Computer Is:

☐ Commodore PET/CBM ☐ OSI
☐ VIC-20 ☐ Radio Shack Color Computer
☐ Apple ☐ TI 99/4A
☐ Atari ☐ Other _____
☐ AIM ☐ Don't yet have one

For Fastest Service Call Our **Toll-Free** US Order Line **800-345-8112**
In Pennsylvania Call 800-662-2444

☐ $20.00 One Year US Subscription ☐ $25.00 Canada and Int'l Surface Mail
☐ $36.00 Two Year US Subscription ☐ $38.00 Europe Air Delivery (Foreign orders must be pre-paid in US Funds)
☐ $54.00 Three Year US Subscription

☐ Payment Enclosed ☐ VISA ☐ MasterCard ☐ American Express
Acct. No. Expires /

Name
Address
City State Zip
Country

COMPUTE! Books

For fastest service, in the US call **Toll Free 800-334-0868.**
In NC call 919-275-9809.

Quantity		Price		Shipping/Handling	
____	**Beginner's Guide**	$ 3.95 ea.	+	$ 1.00 ea.	_____
____	**First Book of Atari**	12.95 ea.	+	2.00 ea.	_____
____	**Inside Atari DOS**	19.95 ea.	+	2.00 ea.	_____
____	**First Book of PET/CBM**	12.95 ea.	+	2.00 ea.	_____
				Total	_____

All orders must be prepaid (money order, check or charge). All payments must be in US funds. (Outside the US add $4.00 shippinng and handling for air mail, $2.00 for surface mail.) NC residents add 4% sales tax.

☐ Payment Enclosed ☐ VISA ☐ MasterCard ☐ American Express
Acct. No. Expires /

Name
Address
City State Zip
Country

Allow 4-6 weeks for delivery. Foreign surface delivery 2-3 months.

COMPUTE! Back Issues

Here are some of the applications, tutorials, and games from available back issues of **COMPUTE!**. Each issue contains much, much more than there's space here to list, but here are some highlights:

January 1981: Load PET Programs Into The Apple II, Player-Missile Graphics for Atari, The Atari DOS, The Kernel of the OSI Operating System, Fixing LOADing Problems on the PET, Spooling with the PET Disk, Expanding KIM.

February 1981: Simulating PRINT USING, Using the Atari as a Terminal for Telecommunications, Attach a Printer to the Atari, Double Density Graphing on C1P, Commodore Disk Systems, PET Crash Prevention, A 25¢ Apple II Clock.

May 1981: Named GOSUB/GOTO in Applesoft, Generating Lower Case Text on Apple II, Copy Atari Screens to the Printer, Disk Directory Printer for Atari, Realtime Clock on Atari, PET BASIC Delete Utility, PET Calculated Bar Graphs, Running 40 Column Programs on a CBM 8032.

June 1981: Computer Using Educators (CUE) on Software Pricing, Apple II Hires Character Generator, Ever- expanding Apple Power, Color Burst for Atari, Mixing Atari Graphics Modes 0 and 8, Relocating PET BASIC Programs, An Assembler In BASIC for PET, QuadraPET: Multitasking?

July 1981: Home Heating and Cooling, Animating Integer BASIC Lores Graphics, The Apple Hires Shape Writer, Adding a Voice Track to Atari Programs, Machine Language Atari Joystick Driver, Four Screen Utilities for the PET, Saving Machine Language Programs on PET Tape Headers, Commodore ROM Systems, The Voracious Butterfly on OSI.

August 1981: Minimize Code and Maximize Speed, Apple Disk Motor Control, A Cassette Tape Monitor for the Apple, Easy Reading of the Atari Joystick, Blockade Game for the Atari, Atari Sound Utility, The CBM "Fat 40," Keyword for PET, CBM/PET Loading, Chaining, and Overlaying.

September 1981: The Column Calculator, What is a Modem and Why Do I Need One?, PET, Apple, Atari: On Speaking Terms, A Tape "EXEC" for Applesoft, A Self-altering Program for Apple II, Positioning P/M Graphics and Regular Graphics in Memory, An Atari BASIC Sort, Shoot, an Arcade Game for Atari, Exploring OSI's Video Routine, PET Tape Append and Renumber, All About LOADing PET Cassettes.

October 1981: Automatic DATA Statements for CBM and Atari, VIC News, Undeletable Lines on Apple, PET, VIC, Budgeting on the Apple, Switching Cleanly from Text to Graphics on Apple, Atari Cassette Boot-tapes, Atari Variable Name Utility, Atari Program Library, Train your PET to Run VIC Programs, Interface a BSR Remote Control System to PET, A General Purpose BCD to Binary Routine, Converting to Fat-40 PET.

November 1981: SuperPet: A Preview, Japanese Micros: A First Look, Introduction to Binary Numbers, An Apple Primer, Page Flipper for Apple, An Atari Database System, A Program for Writing Programs on the Atari, Atari Textplot, OSI Relocation, The PET Speaks, Inversion Partitioning, A Personal News Service on PET, Bits, Bytes, and Basic Boole.

December 1981: Saving Fuel $$ (Multiple Computers: versions for Apple, PET, and Atari), Unscramble Game (multiple computers), Maze Generator (multiple computers), Animating Applesoft Graphics, A Simple Printer Interface for the Apple II, A Simple Atari Wordprocessor, Adding High Speed Vertical Positioning to Atari P/M Graphics, OSI Supercursor, A Look At SuperPET, Supermon for PET/CBM, PET Mine Maze Game.

January 1982: Invest (multiple computers), Developing a Business Algorithm (multiple computers), Apple Addresses, Lowercase with Unmodified Apple, Cryptogram Game for Atari, Superfont: Design Special Character Sets on Atari, PET Repairs for the Amateur, Micromon for PET, Self-modifying Programs in PET BASIC, Tinymon: a VIC Monitor, Vic Color Tips, VIC Memory Map, ZAP: A VIC Game.

February 1982: Insurance Inventory (multiple computers), Musical Transposition (multiple computers), Multitasking Emulator (multiple computers), Disassemble Apple Programs from BASIC, Plotting Polar Graphs on Apple, Atari P/M Graphics Made Easy, Atari PILOT, Put A Rainbow in your Atari, Marquee for PET, PET Disk Disassembler, VIC Paddles and Keyboard, VIC Timekeeping.

March 1982: Word Hunt Game (multiple computers), Infinite Precision Multiply (multiple computers), Atari Concentration Game, VIC Starfight Game, CBM BASIC 4.0 To Upgrade Conversion Kit, Apple Addresses, VIC Maps, EPROM Reliability, Atari Ghost Programming, Atari Machine Language Sort, Random Music Composition on PET, Comment Your Apple II Catalog.

April 1982: Track Down Those Memory Bugs (multiple computers), Shooting Stars Game (multiple computers), Intelligent Input Subroutines (multiple computers), Ultracube for Atari, Customizing Apple's Copy Program, Using PET/CBM In The High School Physics Lab, Grading Exams on a Microcomputer (multiple computers), Atari Mailing List, Renumber VIC Programs The Easy Way, Browsing the VIC Chip, Disk Checkout for PET/CBM.

May 1982: VIC Meteor Maze Game, Atari Disk Drive Speed Check, Modifying Apple's Floating Point BASIC, Fast Sort For PET/CBM, Extra Atari Colors Through Artifacting, Life Insurance Estimator (multiple computers), PET Screen Input, Getting The Most Out Of VIC's 5000 Bytes.

Home and Educational COMPUTING! (Fall 1981 and Summer 1981 – count as one back issue): Exploring The Rainbow Machine, VIC As Super Calculator, Custom Characters, Alternate Screens, Automatic Line Numbers, Using The Joystick (Spacewar Game), Fast Tape Locater, Window, VIC Memory Map.

Back issues are $3.00 each or six for $15.00. Price includes freight in the US. Outside the US add $1.00 per magazine ordered for surface postage. $3.00 per magazine for air mail postage. All back issues subject to availability.

In the Continental US call
TOLL FREE 800-334-0868
(In NC Call 919-275-9809)

Or write to **COMPUTE!** Back Issues, P.O. Box 5406, Greensboro, NC 27403 USA. Prepayment required in US funds. MasterCard, Visa and American Express accepted. North Carolina Residents add 4% sales tax.

COMPUTER CLASS M. HUMPHREY

COMPUTE! Subscriber Services

Please help us serve you better. If you need to contact us for any of the reasons listed below, write to us at:

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

or call the Toll Free number listed below.

Change Of Address. Please allow us 6-8 weeks to effect the change; send your current mailing label along with your new address.

Renewal. Should you wish to renew your **COMPUTE!** subscription before we remind you to, send your current mailing label with payment or charge number or call the Toll Free number listed below.

New Subscription. A one year (12 month) US subscription to **COMPUTE!** is $20.00 (2 years, $36.00; 3 years, $54.00. For subscription rates outside the US, see staff page). Send us your name and address or call the Toll Free number listed below.

Delivery Problems. If you receive duplicate issues of **COMPUTE!**, if you experience late delivery or if you have problems with your subscription, please call the Toll Free number listed below.

COMPUTE!
800-334-0868
In NC 919-975-9809

CAPUTE!:

Modifications Or Corrections To Previous Articles

Improved Search For Apple II

Our thanks to Jim Gordon for the following improvement to the Apple version of "Search For PET And Apple II Plus," June 1982, #25, pg. 43. Change line 700 to:

```
700 FOR ADRES=768 TO 902: READ DTA: POKE
    ADRES,DTA: NEXT
```

and revise the following lines:

```
852 DATA 4, 200, 76, 76, 3, 162
870 DATA 76, 76, 3, 76, 119, 3
888 DATA 163, 32, 237, 253, 32, 32
894 DATA 237, 169, 160, 32, 237, 253
900 DATA 76, 108, 3
```

Self-Modifying P/M Graphics Utility Updated

Ken Grace, the author of "A Self-modifying P/M Graphics Utility," June 1982, #25, pg. 120, sent in the following update to his article.

Line 420 of Program 2 should be changed to:

```
420 IF A=15 THEN 310
```

Further testing of the program revealed that certain combinations of inputs lead to the famous "keyboard lockup" problem. The problem results from having all the deletions bunched together in lines 57-68. By splitting them up and sprinkling them among the earlier lines, the problem does not show up. Some renumbering of lines 3-55 will be needed to make room for these deletion steps. For example, lines 3-12 could be deleted by inserting a new line 13:

```
13 GOSUB 90:FOR I=3 TO 12:? I:NEXT I:GOSUB 91
```

Similarly, the number of players is obtained in line 20; therefore, the deletions in lines 64 and 65 could be done after line 20. Avoid putting the deletions inside the loops from 21 to 40 and from 42 to 51. Line 56 will no longer be needed. The final cleanup, as in line 68, would have to remain at the end, with appropriate changes in the line numbers in the PRINT (?) statements.

Shooting Stars

The following changes should be made to the PET/CBM version of "Shooting Stars" from **COMPUTE!**, April 1982, #23.

```
440 GET K$:IF K$<>" " THEN 480
550 PRINT  " SHOTS FIRED:";SH;"{LEFT} SCORE: {
    REV}";INT(H*100/SH);"{OFF} HITS:";H;
```

New Products

SuperPILOT Added To Instructional Development Software

SuperPILOT, an extension of the Apple PILOT software language, has been announced by Apple Computer, Inc. It joins several new products in Apple's PILOT series that help educators and industrial trainers create lessons and illustrations for computer-aided instruction.

SuperPILOT offers all the capabilities of Apple PILOT plus added features for graphic enhancement, easy debugging, and external video control.

The SuperPILOT program:

– controls external videodisc and videotape through user and computer command and response

– presents "turtle" graphics for graphics programming and discovery learning

– allows for immediate debugging of a program-in-progress, which reduces programming frustration

– displays color text on color background

– displays double-sized characters for emphasis

Also announced are two support products in Apple's PILOT family, Co-PILOT and SuperPILOT Log. Co-PILOT is a completely self-contained, self-paced interactive tutorial on two diskettes which teaches how to program in Apple PILOT. SuperPILOT Log works with SuperPILOT as an administrative record keeping program that automatically tracks test scores by item, student, or class, and can also analyze non-computer test scores entered manually.

Price And Distribution

SuperPILOT (product #A2D0051) will be available mid-July from authorized Apple dealers. Included in the SuperPILOT package is the diskette tutorial Co-SuperPILOT. The program requires an Apple II or Apple II Plus personal computer with 64K of RAM (such as a 48K Apple II Plus with a language card). The suggested retail price is $200.

A price reduction has been announced for Apple PILOT (product #A2D0028). It is now $100, a 33% reduction.

Co-PILOT (product #A2D0050) is priced at $35, and SuperPILOT Log (product #A2D0052) has a suggested retail price of $50.

Apple Computer, Inc.
20525 Mariani Ave.
Cupertino, CA 95014
(408)973-3019

Commodore Introduces New Letter Quality Printer

The new Commodore 8300P Letter Quality Printer, designed especially for use with PET and CBM Computers, has been announced.

A version of the Diablo Model 630 Receive-Only Terminal, the 8300P includes the following standard features: immunity to electrostatic discharge, end-of-ribbon sensor, paper-out detection, cover-open interlock,

internal self-test diagnostics, 320-byte printer buffer and automatic bi-directional printing.

Standard control panel features include: form-feed, pause and reset switches, as well as two lights indicating ready/error and power-on. Switch selectable features available by raising the access cover are printwheel select, pitch, parity, protocol, baud-rate and self-test.

Optional support of languages other than English is available. Optional accessories include an adjustable-width continuous forms tractor mechanism.

The standard ribbon supplied with the CBM 8300P is the Diablo multi-strike film ribbon. The CBM 8300P directly supports use of most Diablo metal or plastic printwheels.

An IEEE to RS-232C printer adaptor will be supplied with the printer. All CBM printers are equipped with a standard PET-IEEE interface connector.

Retail price is $3400.

Commodore Business Machines Ltd.
3370 Pharmacy Avenue
Agincourt, Ontario
M1W 2K4
(416)499-4292

ATARI HOME COMPUTERS

ATARI

ATARI 800	ATARI 400
16K . . . $649	16K . . . $269
32K . . . $729	32K . . . $389
48K . . . $769	48K . . . $489

410 Recorder	$76.00
810 Disc Drive	$449.00
822 Printer	$269.00
825 Printer	$589.00
830 Modem	$159.00
820 Printer	$259.00
850 Interface	$169.00
New DOS 2 System	$29.00
CX30 Paddle	$18.00
CX40 Joy Stick	$18.00
CX853 16K RAM	$77.95
Microtek 16K RAM	$74.95
Microtek 32K RAM	$119.95
Ramdisk (128K)	$429.95
Intec 48K Board	$219.95
Intec 32K	$119.95
One year extended warranty	$70.00
481 Entertainer	$69.00
482 Educator	$130.00
483 Programmer	$49.00
484 Communicator	$344.00

ATARI HOME COMPUTER PROGRAMS

HOME OFFICE	
CX404 ATARI Word Processor	$119.00
CX8102 Calculator	$29.00
CX412 Dow Jones Investment Evaluator	$99.00
CX4109 Graph It Joystick optional	$17.00
CX4104 Mailing List	$20.00
CX4115 Mortgage & Loan Analysis	$13.00
CX4103 Statistics I	$20.00
CX8107 Stock Analysis	$20.00
CXL4015 TeleLink I	$23.00
HOME STUDY	
CX4101 An Invitation to Programming I	$20.00
CX4106 An Invitation to Programming 2	$23.00
CX4117 An Invitation to Programming 3	$23.00
CX4107 Biorhythm	$13.00
Conversational Languages (ea.)	$46.00
CX4121 Energy Czar	$13.00
CX4114 European Countries & Capitals	$13.00
CX4108 Hangman, Joystick optional	$13.00
CX4102 Kingdom	$13.00
CXL 4007 Music Composer	$34.00
CX4123 Scram, uses joystick	$20.00
CX4112 States & Capitals	$13.00
CX4110 Touch Typing	$20.00
HOME ENTERTAINMENT	
PAC MAN	$35.00
CENTIPEDE	$35.00
CAVERNS OF MARS	$32.00
CXL4013 Asteroids	$29.00
CXL4004 Basketball	$27.00
CX4105 Blackjack	$13.00
CXL4009 Computer Chess	$29.00
CXL4012 Missile Command	$29.00
CXL4008 Space Invaders	$29.00
CXL4011 Star Raiders	$39.00
CXL4006 Super Breakout	$29.00
CXL4010 3-D Tic-Tac-Toe	$27.00
CXL4005 Video Easel	$24.00
PROGRAMMING LANGUAGES AND AIDS	
CXL4003 Assembler Editor	$47.00
CXL4002 ATARI BASIC	$47.00
CX8126 ATARI Microsoft BASIC	$70.00
CXL 4018 PILOT	$72.00
CX405 PILOT (Educational)	$105.00
Visicalc	$189.00
Letterperfect (Word Processor)	$109.00
Data Soft Textwizzard	$89.00
Canyon Climber	$24.00
Tumble Bug	$24.00
Shoot Arcade	$24.00
Pacific Coast	$24.00
Bishops Square	$24.00
Micro Painter	$27.00
ATARI PROGRAM EXCHANGE:	
Eastern Front '41	$25.50
Avalanche	$15.50
Outlaw	$15.50
747 Landing Simulation	$15.50
Babel	$15.50
Dog Daze	$15.50
Downhill	$15.50
Attack!	$15.50
Blackjack-Casino	$15.50
Reversi II	$15.50
Domination	$15.50
Solitare	$15.50
Disk Fixer	$15.50
Supersort	$15.50
Data Management	$15.50
Chameleon	$15.50
Instedit	$15.50
Insomnia	$15.50
My First Alphabet	$25.50
Mapware	$18.00
Number Blast	$11.50
Family Cash Flow	$15.50
Weekly Planner	$15.50
Bowler's Data Base	$13.00
Banner Generator	$11.50
Crystal Software	
Bermuda Fantasy	$26.00
Beneath Pyramids	$20.00
Galactic Quest	$26.00
House of Usher	$20.00
Forgotten Island	$26.00
Haunted Palace	$33.00
Compumax (Acct. Rec., Gen. Ledger, Inventory, Payroll, ea.)	$110.00
Synapse	
File Manager 800	$79.95
Dodge Racer	$19.00
Chicken	$24.00
Slime	$24.00
Nautilus	$24.00
Disk Manager	$24.00
Fort Apocalypse	$24.00
Assembler	$39.00
Protector	$24.00
EXPY (Automated Simulation)	
Ricochet	$14.50
Crush, Crumble & Chomp	$24.00
Star Warrior	$29.00
Rescue at Rigel	$24.00
Datestones	$16.00
OnLine	
Jaw Breaker	$27.00
MouskATTACK	$31.00
Invasion Orion	$18.50
Mission Asteroid	$22.00
The Next Step	$34.00
Softporn	$27.00
Wizzard & Princess	$29.00
Arcade Plus	
Ghost Hunter (cassette)	$24.00
Ghost Hunter (disk)	$30.00
K-Byte	
K-BYTE Krazy Shoot Out (ROM)	$39.00
K-DOS	$69.00
K-Razy Kritters	$39.00
K-Star Patrol	$39.00
K-Razy Antiks	$39.00
Stick Stand	$6.99

Texas Instruments

TI-99/4A $299

PHA2100 R F Modulator	$29.00
PHP1600 Telephone Coupler	$179.00
PHP1200 Peripheral Expansion Box	$199.00
PHP1220 RS 232 Card	$143.00
PHP1240 Disk Controller	$199.00
PHP1250 Disk Drive	$319.00
PHP1100 Wired Remote Controllers (pair)	$31.00
PHP1260 32K RAM	$229.00
PHP Printer Solid State	$319.00
PHM3006 Home Financial Decisions	$26.00
PHM3013 Personal Record Keeping	$43.00
PHD5001 Mailing List	$60.00
PHD5021 Checkbook Manager	$18.00
PHM3008 Video Chess	$60.00
PHM3010 Physical Fitness	$26.00
PHM3009 Football	$26.00
PHM3018 Video Games I	$26.00
PHM3024 Indoor Soccer	$26.00
PHM3025 Mind Challengers	$22.00
PHM3031 The Attack	$35.00
PHM3032 Blasto	$22.00
PHM3033 Blackjack and Poker	$22.00
PHM3034 Hustle	$22.00
PHM3036 Zero Zap	$18.00
PHM3037 Hangman	$18.00
PHM3038 Connect Four	$18.00
PHM3039 Yahtzee	$22.00
Tombstone City 21st Century	$34.00
Munch Man	$34.00
T I INVADERS	$34.00
CAR WARS	$34.00

Printers

Smith Corona TP1	$699.00
Centronics 739-1	$519.00
Centronics 739-3	$619.00
Diablo 630 Special	$1799.00
Epson	
MX80 w/Graftrax	$449.00
MX80FT III	$Call
MX100	$Call
NEC	
8023	$549.00
7710/7730	$2399.00
3510/3530	$1789.00
Okidata	
82A	$499.00
83A	$749.00
84	$1129.00
Citoh Starwriter	
F10-40 CPS	$1469.00
F10-55 CPS	$Call
Prowriter	$499.00
Talley	
8024-L	$1629.00
IDS	
Prism	$Call
MPC Apple Parallel Board & Cable	$69.00
2 Meter RS232-RS232	$29.95

Cables Available For Most Interfacing Purposes

Computer Covers

ATTRACTIVE DUST COVERS FOR YOUR COMPUTER AND DISK DRIVE

Atari 400	$6.99	Commodore VIC-20	$6.99
Atari 800	$6.99	Commodore 8032	$14.99
Atari 810	$6.99	Commodore 8050/4040	$10.99
All Atari Covers are Beige.		All Commodore covers are Royal Blue.	

Most software for Atari 400/800 available on cassette or disk.

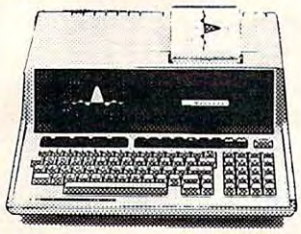
computer mail order west

CALL TOLL FREE **800-648-3311**

IN NEVADA, CALL (702) 588-5654

P.O. BOX 6689, STATE LINE, NEVADA 89449

HEWLETT PACKARD



HP•85 $1899

HP•86	$Call
HP•87	$1769.00
HP•125	$1999.00
HP•85 16K Memory Module	$169.00
5¼" Dual Master Disc Drive	$1769.00
Hard Disk w/Floppy	$4349.00
Hard Disk	$3440.00
"Sweet Lips" Plotter	$1149.00
80 Column Printer	$799.00
87 CP/M	$399.00
87 128K Memory	$610.00
87 Visicalc	$205.00

HP•41CV Calculator $239.00

41C	$189.00
11C	New Low Price $79.00
12C	$114.00
34C	$114.00
38C	$114.00
HP•41 Printer	$340.00

HPIL CALCULATOR PERIPHERALS

IL Modual	$104.00
Digital Cassette	$449.00
Printer/Plotter	$419.00
Card Reader	$164.00
Optical Wand	$99.00

NEC

8001-A	$749.00
8031	$749.00
8012	$549.00
Accounts Receivable	$295.00
Word Processing	$395.00
General Accounting	$295.00
Inventory Control	$295.00
Job Cost	$295.00

Maxell Disks

MD I (box of 10)	$36.00
MD II (box of 10)	$46.00
MFD I (8")	$44.00
MFD II (8" Double Density)	$54.00
Syncom (box of 10)	$29.00

Apple

Call for availability and prices
on all Apple computers and peripherals.

commodore

BUSINESS MACHINES

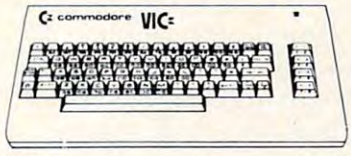
SOFTWARE

Word Pro 5 Plus	$319.00
Word Pro 4 Plus	$299.00
Word Pro 3 Plus	$199.00
Commodore Tax Package	$589.00
Visicalc	$189.00
Medical Billing	$449.00
The Source	$89.00
OZZ Information System	$289.00
Dow Jones Portfolio	$129.00
Pascal	$239.00
Legal Time Accounting	$449.00
Word Craft 80	$289.00
Power	$79.00
Socket-2 Me	$20.00
Jinsam	$Call
MAGIS	$Call
CPA	$Call
Real Estate Package	$Call
The Manager	$209.00
Softrom	$129.00
BPI Inventory Control	$319.00
BPI Job Costing	$319.00
BPI Payroll	$319.00
BPI General Ledger	$329.00
Creative I SAM	$79.00
Creative General Ledger	$229.00
Creative Accounts Receivable	$229.00
Creative Inventory	$229.00

CBM 8032 $1039

4032	$969.00
4016	$769.00
8096 Upgrade Kit	$369.00
Super Pet	$1599.00
2031	$529.00
8250 Doubled Sided Disk Drive	$1699.00
D9060 5 Megabyte Hard Disk	$2399.00
D9090 7.5 Megabyte Hard Disk	$2699.00
8050	$1299.00
4040	$969.00
8300 (Letter Quality)	$1799.00
8023	$769.00
4022	$499.00
Pet to IEEE Cable	$37.00
IEEE to IEEE Cable	$46.00
Tractor Feed for 8300	$240.00

VIC 20 $239
VIC 64 Call



VIC 6 Pack Program	$44.00
VIC 1530 Commodore Datassette	$69.00
VIC 1540 Disk Drive	$499.00
VIC 1515 VIC Graphic Printer	$339.00
VIC 1210 3K Memory Expander	$32.00
VIC 110 8K Memory Expander	$53.00
16K VIC Expansion	$99.00
VIC 1011 RS232C Terminal Interface	$43.00
VIC 112 VIC IEEE-488 Interface	$86.00
VIC 1211 VIC 20 Super Expander	$53.00
VT 232 VICTerm I Terminal Emulator	$9.00
VIC 1212 Programmers Aid Cartridge	$45.00
VIC 1213 VICMON Machine Language Monitor	$45.00
VIC 1901 VIC AVENGERS	$23.00
VIC 1904 SUPERSLOT	$23.00
VIC 1906 SUPER ALIEN	$23.00
VIC 1907 SUPER LANDER	$23.00
VIC 1908 DRAW POKER	$23.00
VIC 1909 MIDNIGHT DRIVE	$23.00
VIC 1910 RADAR RAT RACE	$23.00
Terminal	$13.00
Un Word	$13.00
Grafix Menagerie	$11.00
VIC PICS	$15.00
Ticker Tape	$13.00
Banner Headliner	$13.00
RS 232	$39.00
VT 106A Recreation Pack A	$44.00
VT 107A Home Calculation Pack A	$44.00
VT 164 Programmable Character/Gamegraphics	$12.00
Household Finance	$27.00
VIC Games	$19.00
VIC Home Inventory	$13.00
VIC Rec/Ed II	$13.00
VL 101 Introduction to Computing	$19.00
VL 102 Introduction to BASIC Programming	$19.00
VM110 VIC 20 Programmers Reference Guide	$15.00

In-stock items shipped same day you call. No risk, no deposit on C.O.D. orders. Pre-paid orders receive free shipping within the continental United States with no waiting period for certified checks or money orders. All prices shown are cash prices. Add 3% for Mastercard and Visa. NV and PA residents add sales tax. All items subject to availability and price change.

Televideo

910	$579.00
912C	$699.00
920C	$749.00
925C	$749.00
950	$939.00

Call for computers

802	$Call
802H	$Call
816	$Call
806	$Call

Monitors

BMC 12" Green	$85.00
Amdek	
300G	$169.00
Color I	$339.00
Color II	$699.00
Color III	$429.00
TI 10" Color	$349.00
Zenith 9" (Green)	$119.00

Modems

Hayes	
Smart	$239.00
Chronograph	$199.00
Micromodem II	$279.00
Micromodem 100	$309.00
Novation Auto	$239.00
D Cat	$169.00
Cat	$159.00
Anchor Modem	$79.00

Nev. & Pa. residents add sales tax.

computer mail order east

CALL TOLL FREE **800-233-8950**

INTERNATIONAL CALLS AND IN PA. CALL (717) 327-9575

477 E. THIRD ST., WILLIAMSPORT, PA 17701

Starship Duel: A Two-Player Computer Game From Program Design

Program Design has released Starship Duel, a two-player computer game written by John Kanopa.

The object of the game is to destroy the opponent's fleet of starships, while losing as few of your own starships as possible. The greater the number of ships remaining in your fleet after the opposing fleet has been destroyed, the higher the score.

A starship's laser fires only in the direction that the ship moves. Thus, quick handling of the joystick is needed to chase the enemy, or to get out of its way.

Each of the ten ships in a fleet has a limited amount of ammunition. If it is used up, the ship is expended. It is possible to replenish a ship's ammunition supply by hitting a white "X" that occasionally pops up on the screen. But this requires quick action, for the "X" only remains on the screen for a brief moment.

Starship Duel consists of four games. Game 1 is the simplest: one-on-one starship combat until one fleet is destroyed. In Game 2 the starships become partially or totally invisible as they move toward the left and right edges of the screen. They can still be destroyed – if the opponent knows where they are hiding. Game 3 has a blinking phantom ship that moves independently across the field of battle. If the phantom ship collides with another ship, the second ship is destroyed. However, if a player hits a phantom ship with his or her laser fire, the phantom ship becomes that player's ally, and will only destroy the opponent's ships. Game 4 is a combination of Game 2 and 3.

Starship Duel is available for use on Atari 400/800 computers with a memory of at least 16K. Available on cassette, it retails for $19.95.

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

Computer Furniture

H.S.P. (Health Science Products, Inc.) introduces its ergonomically designed Computer Furniture, the DataLeggett.

This split-level CRT Workstation is comfort engineered to meet individual needs. A major feature of the Workstation is the frontal placement of the copy holder, or leggett. The

ECLECTIC SYSTEMS CORPORATION
Order TOLL FREE 1-800-527-3135

P.O. Box 1166 16260 Midway Road
Addison, TX 75001 (214) 661-1370

Authorized Commodore service center

Sales and Service of the complete line of Commodore products

In a hurry? Check our modular exchange program

HARDWARE:

CBM 8032 Computer, 80 Column	$1065
CBM 8050 Disk Drive	1299
CBM 4032 Computer, 40 Column	965
CBM 4040 Disk Drive	965
CBM 4022 Printer	595
CBM VIC 20 Computer	259
CBM VIC 1515 Printer	349
CBM 8300P (DIABLO)	1799
CBM VS100 Cassette	65
PET to IEEE Cable	33
IEEE to IEEE Cable	39

SOFTWARE:

OZZ	289
Wordcraft 80	289
The Manager (Data Base)	240
Wordpro 5+	319
Wordpro 4+	299
VISIC ALC	169
MUMPS for Super PET	299
CMS Accounting System	call
Assembler Development Package	77
BASF Diskette, Box of 10	30

ANS MUMPS Programmer's Reference Manual	$17.50
MUMPS Pocket Guide	2.00
Computer Programming in ANS MUMPS	17.50
RS232 Interface for Commodore VIC 20	39.95

EIO has: two serial asynchronous RS232 ports, two parallel ports with handshaking, one shift register, two 16 bit timers, and room for two optional buffer IC's. 188.00

EIO-C Alternate character generator ROM board for screen display allows you to display characters or graphics of your choice. Alternate characters are soft selectable. EIO board required. Call for price

EIO-TX Terminal ROM for EIO firmware to turn your CBM into a communications terminal. Store and transmit from disk to remote host and terminal. EIO board required. 49.95

EIO-RS232 Cable for EIO board. 28.00

BOOKS:

Commodore Software Encyclopedia 2nd edition	9.95
The PET Revealed	19.95
Library of PET Subroutines	19.95
PET Interfacing	16.95
PET Basic	12.95
PET and IEEE 488 Buss	15.00

OTHER:

EPROM Burner for CBM burns 2716, 2732 & 2532	89.95
Software for EPROM Burner on CBM	15.95
D.C. Hayes Smart Modem	239.00

ESC-100 RS232 interface manual selector—switch select between device A, B, or C; switching 25 conductors. Requires no input power and uses receptacle type DB25S. 140.00

ESC-120 M43 TTY-EIA/CURRENT LOOP interface unit—converts TTL level signals from the M43 to EIA or 20ma outputs. Convert EIA or 20ma inputs back to TTL level signals for the M43. Single printed circuit mounts inside the M43. Comes with interface cable. 65.00

ESC-140 RS232 standard interface cable kit. Build your own cable.

7 conductor	16.00
12 conductor	19.95

ESC-150 The Electronic Switch/Poller allows a single computer serial port to selectively communicate with an assortment of peripheral devices. As many as seven serial asynchronous and six parallel devices may be selected. This concept effectively creates a simplified multi-drop, polled environment. The computer maintains control of the network and may seize any of the peripherals by transmitting a simple escape sequence to the "ESC-150" 488.00

ESC-170 RICKETYMETEr-Indispensable hand-held tool for trouble-shooting, checkout and installation of RS232 type serial line communications systems. 69.95

ESC-180 RICKETYTRAp is a small hand-held device which when interposed in an EIA/RS232 line can monitor and trap on any specific ASCII, binary, or control character. The RICKETYTRAp can also monitor and trap on any specific range of characters bit selectable. This device can also check for type of parity being sent and proper framing. The RICKETYTRAp is switch selectable for 1 or 2 stop bits and can operate at speeds of 150 to 9600 baud. Call for price

ESC-200 PREDITOR is a small microprocessor with ROM firmware, RAM, and 8 RS232 asynchronous serial input/output ports. The PREDITOR is available in either stand-alone, self-contained, or rack-mounted models. The PREDITOR can be programmed by firmware to function as a prompter and editor in a distributed processing network, transforming 4 dumb asynchronous terminals into 4 intelligent terminals communicating with 4 computer asynchronous input/output ports. Call for price

ESC-861M The DATA CONCENTRATOR accepts data from a multitude of inputs—printers, CRTs, parallel ports, status lines, etc.—then transmits the composite (multiplexed) data stream down a high speed synchronous serial line to a remote DATA CONCENTRATOR. The remote DATA CONCENTRATOR separates (demuxes) the composite data into its original form to be transmitted to the computer ports. Alternately, multiple computer port data is multiplexed, transmitted down the high speed synchronous serial line, demuxed, and finally dispatched to the individual terminals. 850.00

Call toll-free for shipping charges.

Order TOLL FREE 1+800-527-3135

10 AM to 4 PM CDT Monday through Friday

Texas residents call 1+214-661-1370

VISA, MASTER CHARGE, MONEY ORDERS, AND C.O.D. "Certified Check" accepted.

PERSONAL CHECKS REQUIRE 2-WEEK WAITING PERIOD.

Units in stock shipped within 24 hours, F.O.B. Dallas, Texas.

All equipment shipped with manufacturer's warranty.

Residents of Texas, Louisiana, Oklahoma City and Tulsa, Oklahoma must add applicable taxes.

Be sure to write to the address at the top of this ad for more information. Dealer inquiries welcome.

DataLeggett holds the source document in front of the operator between keyboard and screen, so the operator doesn't need to constantly look off-side at reference material. It can be adjusted in degree of tilt and lifts up for access to a storage area.

The height adjustable video display platform slants for optimum glare control from overhead lighting and windows. This design provides proper viewing position and distance from video display to reduce eye strain.

H.S.P. Computer Furniture
P.O. Box 5545
Birmingham, AL 35207

File Management Software For The Apple III

Apple Computer, Inc. announces Quick File III, a filing system for managing small to medium size collections of information on the Apple III personal computer.

With Quick File III, a doctor, small business owner, homemaker, or scientist can quickly turn receipts, notes, lists, and schedules into coherent files and reports. Quick File III allows for simple arrangement of records in alphabetic, numeric, date, or time order and saves time and effort in producing repetitive reports, calculations, and corrections. Two types of report formats – tables (rows and columns) and labels or index cards – can be easily created and printed.

Quick File III...

– allows the user to design forms to meet special needs

– allows categories to be added and deleted without retyping previously-held information

– can selectively search, display, and summarize records

– can view many records simultaneously

– can "talk" to Apple Writer III and other ASCII character files

The program provides these additional convenient reporting features:

– calculates totals and subtotals of numeric information

– contains a calculated column (for percentages, the sum of two other columns, etc.)

– allows for the choice of which rows and columns are printed and in what order.

The program (product #A3D0020) requires an Apple III system with at least 128K bytes RAM. It has a suggested retail price of $100, and will be available in late August from authorized Apple dealers.

Apple Computer, Inc.
20525 Mariani Ave.
Cupertino, CA 95014
(408)973-3019

Colorport Cartridge For TRS-80 Color Computer

The Colorport plug-in cartridge adds I/O capability to the TRS-80 Color Computer, resulting in a cost-effective 6809-based control system. This unit adds two fully programmable 8-bit bidirectional parallel ports with full handshaking, which can be configured by the user for versatile interfacing to peripherals. Interrupts are supported, and important computer voltage and logic lines are brought out to the standard 44-pin edge connector. The Colorport has its own power supply, ensuring no system power degradation.

A socket in the cartridge allows insertion of either 2K bytes of RAM or 2K bytes of EPROM. This allows software for the control of I/O operations to be stored separately from the main user memory space. Provision is also made for selection of autostart of the memory in the cartridge and of synchronous reset of the Colorport and the computer.

The Colorport cartridge comes complete with power supply and full instructions, and sells without any memory for $129.95. 2K RAM chips are available for $19.95 each, 2K EPROMS are available for $12.95 each.

Maple Leaf Systems,
P.O. Box 2190
Station "C", Downsview
Ontario, Canada M2N-2S9

Educational Shows Scheduled

ECCO, The Educational Computer Consortium of Ohio, presents the Second Annual Educational Computer Fair on October 16, 1982, at Cleveland State University.

Forty workshops for beginning and experienced computer users, small discussion groups, audio-visual displays, vendor exhibits, and student demonstrations will be held. This is a fair for educators K through College, *by* educators, *for* educators.

For further information contact:

Ellen Richman
ECCO Coordinator
4777 Farnhurst Rd.
Cleveland, OH 44124

* * * * * * *

Commodore is planning a series

DISCOUNT COMPUTER SOFTWARE ACCESSORIES

apple® commodore® IBM® ATARI®

APPLE	Retail	Discount
Crossfire	$ 29.95	$ 21.00
Cannonball Blitz	34.95	25.00
Mouskattack	34.95	25.00
Bandits	34.95	25.00
Lemmings	29.95	21.00
A2-FS 1	29.95	21.00
AP-PBI (Pinball)	29.95	21.00
Goldrush	34.95	25.00
Deadline	49.95	36.00
Eliminator	24.95	18.00
Raster Blaster	29.95	21.00
PFS	125.00	90.00
D.B. Master	229.00	165.00
T.G. Game Paddles	39.00	29.00
T.G. Joystick	59.00	44.00
Visicalc 3.3	250.00	190.00
Frogger	34.95	25.00
The Joyport	74.95	54.00
Snack Attack	29.95	21.00
Gorgon	39.95	29.00
Hi-Res Adv #2 Wiz and Princess	32.95	24.00
Hi-Res Adv #1 Mission Asteroid	19.95	14.00
Hi-Res Adv #5 Time Zone	99.95	72.00
David's Midnight Magic	34.95	25.00
The Home Accountant	74.00	54.00
Apple Panic	29.95	21.00
Bug Attack	29.95	21.00
Magic Window	99.95	72.00
Super Text II	150.00	100.00
Visitrend/Visiplot	300.00	240.00
Castle Wolfenstein	29.00	21.00

CPM	Retail	Discount
Adventures 1-12	$129.00	$ 97.00
WordStar	495.00	350.00
DataStar	350.00	275.00
Mailmerge	150.00	100.00
Supersort	250.00	175.00
SpellStar	250.00	175.00
WordMaster	150.00	100.00
CalcStar	295.00	190.00
Basic Compiler	395.00	295.00
Basic - 80	350.00	260.00
dBase II	700.00	520.00
SuperCalc	295.00	225.00
Graham Dorian - Accounts Payable	1000.00	720.00
Graham Dorian - Accounts Receivable	1000.00	720.00

IBM	Retail	Discount
Temple of Apshai	$ 39.95	$ 29.00
The Home Accountant Plus	150.00	110.00
Mathemagic	89.95	70.00
IBM Joysticks	64.95	48.00
Visicalc	200.00	160.00
Visicalc/256 K	250.00	200.00
Deadline	49.95	36.00
SuperCalc	295.00	220.00

TRS-80	Retail	Discount
Attack Force (d)	$ 19.95	$ 16.00
Galaxy Invasion (t)	15.95	12.00
Invasion Orion (d) (t)	24.95	18.00
Sorcerer of Siva (d) (t)	29.95	22.00
Rescue at Rigel (d) (t)	29.95	22.00
Crush, Crumble & Chomp (d) (t)	29.95	22.00
Hellfire Warrior (d) (t)	39.95	29.00
Galactic Trader (t)	14.95	11.00
Galactic Trilogy (d)	39.95	29.00
Le Stick	39.95	29.00
Sargon II (t)	29.95	21.00
Battle of Shiloh (t)	24.95	18.00
Tigers in the Snow (t)	24.95	18.00
Flight Simulator (t)	25.00	19.00
Alien Armada (d)	18.95	14.00
Adventures' 1-12 (Gold Edition) (t)	100.00	75.00

SUPER SPECIALS
Zenith 12" Green Monitor $120.00
Intec 32K Board (Atari) $85.00

T=Cassette
D=Disk
C=Cartridge

ATARI	Retail	Discount
Bug Attack (d) (t)	$29.95	$21.00
Crossfire (d) (t)	29.95	21.00
Deadline (d)	49.95	36.00
Megalegs (t)	34.95	25.00
Ghost Hunter (d)	34.95	25.00
PacMan (c)	44.95	33.00
Centipede (c)	44.95	33.00
Tumble Bugs (d)	29.95	21.00
Action Quest (d) (t)	29.95	21.00
Battle Trek (d)	39.95	29.00
Star Warrior (t)	39.95	29.00
S.C.R.A.M. (t)	24.95	18.00
Invasion Orion (d) (t)	24.95	18.00
Survival/Adventure (t)	24.95	18.00
Personal Finance Management	74.00	54.00
Jawbreaker (d) (t)	29.95	21.00
Frogger (d)	34.95	25.00
Raster Blaster (d)	29.95	21.00
Apple Panic (d)	29.95	21.00
Text Wizard (d)	99.95	75.00
Match Racers (d)	29.95	21.00
Visicalc (d)	250.00	195.00
Hi-Res Adv Wiz and Princess (d) (t)	32.95	24.00
Star Raiders (c)	49.95	36.00
Asteroids (c)	44.95	33.00
K-Razy Shootout (c)	49.95	36.00
Midway Campaign (t)	16.00	12.00
Crush, Crumble and Chomp (t)	29.95	21.00
Canyon Climber (d)	29.95	21.00

MANY MORE PROGRAMS AVAILABLE — **VISA AND MASTERCARD ACCEPTED**

TERMS: Send check or money order for total purchase price, plus $2.00 for shipping. MI residents add 4% tax. C.O.D. accepted.

® MFGS. TRADEMARK

STRÖM SYSTEMS INC.
P.O. Box 197
Plymouth, Mi. 48170
(313) 455-8022

WRITE OR CALL FOR FREE CATALOG
PHONE ORDER HOURS
4PM - 7PM MON. - FRI.
INCLUDE CARD NUMBER AND EXPIRATION DATE WITH CREDIT CARD ORDERS.
INCLUDE TYPE OF COMPUTER.

Software St.

810 Disk Drive $429.00

We also feature tremendous savings from:

Atari
Atari Program Exchange
Adventure International
Dynacomp
Quality Software
Avalon Hill
Epyx
Computer Consultants
P.D.I.
L.J.K.
Synapse
Datasoft
United Software
On-Line
O.S.S.

800 Computer 16K $639.00

Call or write for your FREE catalog.

Shipping costs:
Software - minimum $2.50
Hardware - prices will vary (please call)

Software Street
3392 Clipper Dr.
Chino, CA 91710
(714) 591-3061

ATARI®
ATARI IS A REGISTERED TRADEMARK

MasterCard VISA

of educational shows for the fall of 1982. At each of the locations listed below a one-day conference on Computers in Education will be offered to teachers and administrators.

These conferences will offer a number of components including workshop sessions on various aspects of computers in education, "hands-on" experience, a keynote speaker, numerous handouts and training material, a drawing for a free computer and the debut of many new Commodore products.

Denver	– September 15
San Francisco	– September 22
Los Angeles	– September 29
San Diego	– October 13
Seattle	– October 20
Orange County	– October 27
Salt Lake City	– November 10
Phoenix	– November 17
Portland	– November 30

To register for any of these shows, write to:

Commodore Business Machines
Att'n: Jim Bussey
3330 Scott Boulevard
Santa Clara, CA 95050

or call one of the following numbers:

In Calif. (toll-free) 800-422-2122
Outside Calif. (toll-free) 800-854-8055
or 408-727-1130, ext. 213

Compumax Announces Micropers

Compumax Associates, Inc. of Menlo Park, California, announces the availability of Micropers for the Atari 800. Micropers contains both a complete payroll system and a personnel management system.

As in the previous Micropers versions, the payroll system calculates the payroll for both hourly and salaried employees and figures federal and California withholding*, social security tax, disability insurance, miscellaneous deductions, and gross and net pay. Using these figures, it prints the actual paychecks. Micropers also fills out W-2 forms and provides the values for the quarterly 941 Report. The Job Cost Report/Labor tells you how much has been spent on labor for each job, and may be used in conjunction with the Job Cost Report/Materials in Microinv to provide total job costing.

One feature that has been added is the Recap Summary Report, which gives company totals for such categories as wages, job costs, and taxes. Another feature unique to this version of Micropers is menu selections for copying your data files, making it even easier to safeguard your data.

In its personnel management capacity, Micropers provides a complete employment history for each employee, including vital statistics, status, position, and earnings, both current and previous. The master file also keeps track of accumulated deductions for each employee.

Micropers retails for $200 and comes complete with program, sample data, and thorough user documentation. BASIC source code is also included, enabling you to modify the program to suit your own particular needs. Hardware requirements include: Atari 800, 48K, 2 disk drives, and printer (optional).

Compumax Associates, Inc.
P.O. Box 7239
Menlo Park, CA 94025

*Micropers is a California payroll package. It must be customized for other states or foreign countries.

80-Column Text Editor From Metaresearch

The Metatext package by Metaresearch, Inc. comes on a single master disk, giving the user many Apple II system options.

Features of the package include: full ASCII 80-column software-packed alphanumerics, 40-column option for enhanced readability, creation routines allowing user to make custom fonts, a text formatter, and various line-oriented text editors. The package includes a serial output program which will drive most RS232 printers from the existing Game I/O connector.

The Metatext user can mix alphanumerics with graphics in arbitrary ways. This is because the font display routines, which use Apple II high-resolution graphics, have a memory-forcible blind cursor option for positioning characters.

The 80-column option is useful for editing and formatting, because the Apple display appears like the true printed page. Because CRTs vary in their resolution (a composite video monitor is best for Metatext), the package comes with 40-column font which is highly readable. As an example of arbitrary font, a Cyrillic (Russian language) text editor is supplied on the standard disk master. Editors which handle such foreign fonts, or even symbol tables for process control, are, in principle, capable of driving dot-matrix printers so that arbitrary font hard-copy can be obtained. All that is required is a dot-matrix printer which allows random-access dot printing. Then the user can create custom subroutines with which to drive the

printer from editing mode.

Metatext is written in Applesoft, except for numerous instances in which machine-code speed is required. The essential machine routines can be called from within BASIC programs, as spelled out in the user manual. Thus, the user can print out in upper or lower case from BASIC, switch scrolling on and off, and so on. Graphs created in HGR (high-resolution graphics) mode can be labelled due to the blind-cursor forcing option.

Metatext also allows for data processing of *mixed* structures. Specifically, the user can first use a MEDIT program to create columns of data, where each column is either all strings or all numbers. But different columns can be of different type. Then a BASIC germ program called PRO.DS, which processes one Data / one String in a two-column format, can be modified to handle the edited data.

With Metatext, there are no hardware modifications to the Apple II. A printer is normally driven out of pins 8 and 15 of the Apple Game I/O. The signals involved are unipolar, so a few rare printers cannot be so driven. In such a case, the user adds the circuit suggested in the Metatext manual to generate bipolar drive. The parts cost for such a unipolar-to-bipolar circuit is a few dollars. Metatext programs require the full 48K memory option for the Apple II.

The Metatext package, purchased as a single disk master along with the forty page user's manual, sells for $79.00.

For further information contact:

Metaresearch, Inc.
1100 SE Woodward St
Portland, OR 97202
(503)232-1712

Estate Tax Plan For Apple II

Aardvark Software, Inc. announces the release of its Estate Tax Plan program. Designed specifically for accountants, attorneys, insurance agents, trust officers, and financial planners, the program allows complex estate tax planning problems to be solved in a short time.

Estate Tax Plan allows the estate planner to enter a variety of factors affecting the gross estate, allowable deductions, and disposition of the client's assets via trust arrangements or bequests. It will then calculate the related effects attributable to changes in one or more of these items.

The program can construct a comparative analysis among up to four alternatives simultaneously. Estate tax planning considerations which may be examined are listed below.

– various dates of death for the client and spouse

– various valuations of the client's asset inventory

– selected marital deduction formula clauses in the client's will (e.g., maximum, "zero-tax," and equalization clause formulas)

– analysis of possible charitable bequests

– available estate tax deferral under IRC Section 6166

– available special use valuation under IRC Section 2032A

– availability and magnitude of redemptions of closely-held stock at capital gains rates under IRC Section 303

– growth rate assumptions concerning property passed to the surviving spouse

– present value analysis relative to impending estate tax liabilities

– cash needs and liquid assets available at death

Calculations performed by Estate Tax Plan result in the following seven reports: Gross Estate, Estate Tax Liability, Present Value Analysis of Estate Taxes, Deferred Payment of Estate Taxes, Deferred Payment Schedule, Liquidity Analysis, and IRC Section 303 Capital Gain.

The program was developed under the supervision of William A. Raabe, Ph.D., CPA, and is currently available for the Apple II (48K) or Western Digital Microengine. It is also expected to be available for a variety of CP/M systems in the near future.

Aardvark Software, Inc.
783 North Water Street
Milwaukee, Wisconsin 53202
(414)289-9988

Fabric Covers For The Atari

A new line of custom-tailored fabric dust covers for Atari home computers is being marketed by Empulse, a Massachusetts-based computer accessory firm.

Called "Cover-Ups," the dust covers are sewn of water-resistant rainwear poplin and are tailored to fit specific Atari

models precisely, while allowing ready access to I/O ports.

Cover-Ups are designed to provide a high-quality alternative to loose-fitting vinyl covers with no I/O access.

The dust covers are available by direct mail from Empulse in three colors: beige and chocolate brown – to match Atari computer colors – and navy blue.

Lyco Computer Marketing & Consultants

TO ORDER CALL US

TOLL FREE 800-233-8760

In Pa. (717) 398-4077

ATARI®

A Warner Communications Company

AUGUST SPECIAL

ATARI 800 48K RAM $735.00
ATARI 400 16K RAM $299.00

MICROTEK MEMORY BOARDS
16K RAM $70.00
32K RAM $129.00
16K RAM (for VIC-20) $99.00

ATARI HARDWARE

ATARI 800 16K $635.00
ATARI 400 16K RAM $299.00
810 DISK DRIVE $455.00
410 CASSETTE RECORDER ..... $ 75.00
850 INTERFACE $164.00
830 PHONE MODEM $159.00
825 PRINTER $585.00

PROGRAMMING SOFTWARE

CX4101	INVITATION TO PROG I	$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

ENTERTAINMENT & EDUCATION SOFTWARE

CXL4012	MISSILE COMMAND	$ 30.00
CXL4009	CHESS	$ 29.00
CXL4004	BASKETBALL	$ 30.00
CXL4005	VIDEO EASEL	$ 30.00
CX4015	BLACKJACK	$ 13.00
CX4107	BIORHYTHM	$ 13.00
CXL4013	**ASTEROIDS**	**$29.00**
CXL4008	**SPACE INVADERS**	**$29.00**
CXL4011	**STAR RAIDERS**	**$32.00**
CXL4006	**SUPER BREAKOUT**	**$29.00**
CXL4022	PACMAN	$ 35.00
CX8130	CAVERNS OF MARS	$ 33.00
	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**	**$35.00**
CX4110	TOUCH TYPING	$ 19.00
CX4103	STATISTICS	$ 19.00
CX404	WORD PROCESSOR	$129.00
CX406	PERSONAL FINANCE	$ 62.00
CX40	JOYSTICKS	$ 18.00
CX853	16K RAM	$ 85.00

THIRD PARTY SOFTWARE
for atari 800 or 400

AUTOMATED SIMULATION:

Invasion Orion	$ 22.00
Rescue at Regel	$ 24.00
Crush, Crumble, & Chomp	$ 24.00
Star Warrior	$ 35.00

ATARI PROGRAM EXCHANGE:

Eastern Front	$ 25.50
My First Alphabet	$ 25.50

K BYTE: K-RAZY SHOOT OUT ..... $ 35.00

commodore

VIC-20 ..... $249.00

VIC1010	EXPANSION MODULE	$135.00
VIC1530	DATASSETTE	$ 67.00
VIC1540	DISK DRIVE	$499.00
VIC1515	PRINTER	$355.00
VIC1210	3K RAM	$ 35.00
VIC1110	8K RAM	$ 52.00
VIC1211A	SUPER EXPANDER	$ 53.00

VIC-20 SOFTWARE

VIC1212	PROGRAMMER AID	$ 45.00
VIC1213	VICMON	$ 45.00
VIC1906	SUPER ALIEN	$ 19.00
VIC1914	ADVENTURE LAND ADVENTURE	$ 3500
VIC1915	PRIVATE COVE ADVENTURE	$ 35.00
VIC1916	MISSION IMPOSSIBLE	$ 35.00
VIC1917	THE COUNT ADVENTURE	$ 35.00
VIC1919	SARGON II CHESS	$ 35.00

THIRD PARTY SOFTWARE

ALIEN BLITZ	$ 21.00
SIMON	$ 10.00
SATELLITES & METEORITES	$ 21.00
KOSMIC KAMIKAZE	$ 21.00
AMOK	$ 21.00
SUPER HANGMAN	$ 16.00
SPIDERS OF MARS	$45.00

To Help evaluate your needs or
If you wish to make a purchase

LYCO COMPUTERS
P.O. Box 10
COGAN STATION, PA 17728

NEW PHONE (717) 398-4079

Prices are $9.95 for the Atari 800 model and $8.95 for the Model 400.

Additional information is available from:

Empulse
22 Elm St.
P.O. Box 593
Great Barrington, MA 01230

Queue Computer Learning Centers

Queue, Inc. is offering a turnkey Computer Learning Center to schools and private investors, preferably educators. The Computer Learning Center will combine popular, inexpensive microcomputers and off-the-shelf software into an organized curriculum in computer programming, computer literacy, the traditional academic areas and specialized test preparation, such as SAT's. Queue's Computer Learning Centers are available for $15,000, and include all necessary hardware, software, course outlines and brochures, literature and training to run a complete profit or non-profit Learning Center. Lease plans are also available, starting as low as $500.00 per month. For information contact:

Jonathan D. Kantrowitz
c/o Queue, Inc.
5 Chapel Hill Drive
Fairfield, CT 06432
(203)335-0908.

New Journal Calls For Papers

A new quarterly, *The Journal of Computers Reading & Language Arts* (CRLA), is ready to receive papers. The journal's purpose is to support the rapidly growing interest in computers and their relationship to reading/language arts and related issues. The theme of the journal will be pragmatic in perspective. It will emphasize presenting papers which

60% MORE POWER! THAN VIC-20

$299.00

28K COMPUTER Sale Price

- **We Stock More VIC-20 Programs and Accessories than anyone in the USA! Over 270 Educational Tapes**
- **Expand to 60K 6 Slot Switch Selectable Reset Button Expansion Module Sale Price Only $109.00 In Stock Now!**
- **Save $30.95 get 6 games $89.70 value for $58.95—**
- **Free Catalog.**

FOR THE SPECIAL SALE PRICE OF $299.00 you get the COMMODORE VIC-20 computer *plus* WE ADD 3000, BYTES OF MEMORY to give you 60% MORE PROGRAMMING POWER! This powerful fullsized extra featured computer includes the 6502 microprocessor (LIKE APPLE) 20,000 bytes ROM with a 16K extended LEVEL II Microsoft BASIC, 8000 bytes RAM plug in expandable to 32,000 bytes RAM, 66 key typewriter professional expanded keyboard with graphic symbols on keys, color command keys, high resolution graphics, 512 displayable characters, text display is 22 lines 23 characters, sound and music, real time, upper lower case, full screen editing cursor, floating point decimal and trig functions, string arrays, scrolling, multi statement lines, file managment, PEEK AND POKE. Assembly machine language is available. We have easy to use self teaching books and programs. Accept TAPE-DISK-PLUG IN CARTRIDGES, connects to any TV, includes AC adaptor, R.F. modulator, switch box, self teaching instruction book, comes in a beautiful console case for ony $299.00.

LOW COST PLUG IN EXPANSION

Expansion accessories plug directly into this computer, extra RAM memory, Controllers, a Cassette, A Telephone Modem for only $109.00, an 80 Column Printer for $375.00, even the 170K Disk Drive plugs in direct. You do not have to buy an expensive expansion interface.

WHY SUCH A LOW PRICE

WE GIVE YOU 60% to 400% MORE PROGRAMMING POWER THAN VIC-20! You can't beat our prices for the VIC-20 with increased programming power added! We sell direct to customers. We save you the profit margin normally made by computer stores, department stores, and distributors. We are willing to take a smaller margin to develop volume!

INVEST IN YOUR CHILDREN

Educate your children while they play. Every kid wants to play electronic games. (We have some of the best). The next natural step for their curiosity is to try simple programming. They can do this in 20 minutes with our simple self teaching instruction book. High schools are teaching computer math, science and programming - some start in grammar school. If you provide this computer as a Teacher and Tutor at home, before you know it your child will be writing computer programs. You can use your T.V. to EDUCATE not frustrate your family and eliminate T.V. boredom with programs that challenge, stimulate and entertain the whole family. We have a wide variety of games, recreational, home finance and educational programs to choose from. Why pay $140.00 to $295.00 for an electronic game when you can buy this powerful computer for only $299.00.

IMMEDIATE REPLACEMENT WARRANTY

If your computer fails because of warranty defect within 90 days from date of purchase, you simply send your computer to us via United Parcel Service prepaid. We will "immediately" send you a replacement computer at no charge via United Parcel Service prepaid. No one we know gives you this kind of warranty service. Most computer warranty service takes 30 to 90 days to handle - this fantastic "immediate replacement warranty" is backed by COMMODORE COMPUTER, a MAJOR national brand electronics manufacturer.

TELEPHONE MODEM SALE $109

Plug in your VIC telephone modem. Now you can get a world of information through your telephone, plus electronic mail. Just dial up the information you want. UPI wire service, stock market, historical information by topic from over 60 magazines, including New York Times. Airline information, order tickets, get weather information anywhere in the world, restaurant and hotel information, thousands of categories are on line for you, business, finance, education, entertainment, games etc. YOU'LL BE THE TALK OF YOUR NEIGHBORHOOD. Our telephone modem price is only $109 and includes FREE! one year network membership and one hour on line!

SPECIAL SALE PRICE $299

FOR ONLY $299 you get the POWERFUL 28K COMMODORE VIC with 60% MORE PROGRAMMING POWER THAN VIC-20! 28,000 bytes total memory (20,000 bytes ROM, 8000 bytes RAM and extended LEVEL II BASIC), the professional 66 keyboard, color, sound, music self teaching instruction book, A.C. adaptor, R.F. modulator, T.V. switch box, owners manual plus all the other features listed, in a beautiful console.

SPECIAL SALE PRICE $379

FOR ONLY $379 you get the 41K COMMODORE VIC with 400% MORE PROGRAMMING POWER THAN VIC-20! We add 16,000 bytes memory to the VIC-20. You get a total of 41,000 bytes memory (20,000 bytes ROM, 21,000 bytes RAM and extended LEVEL II BASIC) plus all the extra features shown for the 28K COMMODORE VIC.

SPECIAL DATA CASSETTE SALE

THIS SPECIAL DATA CASSETTE has special electronics that eliminates loading problems and loss of programs recorded on tape! Includes tape counter and your selection of any $14.95 "GAME PACK" program FREE!!! Reg. Price $90.00 Sale Price $69.00

15 DAY FREE TRIAL

DON'T MISS THIS SALE -ORDER NOW

- ☐ Please send me the 28K Commodore VIC Computer for $299.00
- ☐ Please send me the 41K Commodore VIC Computer for $379.00
- ☐ Telephone Modem $109
- ☐ Special Data Cassette $69.00

We ship C.O.D. and honor Visa and Master Card.

Name _____

Address _____

City _____

State _____ Zip Code _____

☐ VISA ☐ MASTER CARD ☐ C.O.D.

Credit Card No. _____

Expiration Date _____

Add $10.00 for shipping, handling and Insurance. Illinois residents please add 6% tax. Add $20.00 for CANADA, PUERTO RICO, HAWAII orders. WE DO NO EXPORT TO OTHER COUNTRIES.

Enclose Cashiers Check, Money Order or Personal Check. Allow 14 days for delivery, 2 to 7 days for phone orders, 1 day express mail!

Canada orders must be in U.S. dollars.

we are **commodore** VIC experts!!

PROTECTO ENTERPRIZES

BOX 550, BARRINGTON, ILLINOIS 60010

Phone 312/382-5244 to order

have clear classroom/teaching implications.

The journal will be interdisciplinary and directed toward an audience of reading-language arts teachers, educational specialists, classroom teachers, educators of teachers and educational researchers.

Gerald H. Block, ALP Reading Clinics in Berkeley and Oakland, serves as editor of CRLA. Danny B. Daniel, University of Minnesota; Peter Joyce, York Board of Education, Toronto; Dorothy Judd, Northern Illinois University; George E. Mason, University of Georgia; Barbara Micetich, Diocese of Washington, DC; and Richard L. Shell, University of Cincinnati, serve as the editorial board.

Papers or requests for information should be sent to:

Gerald H. Block
CRLA
P.O. Box 13039
Oakland, CA 94661.

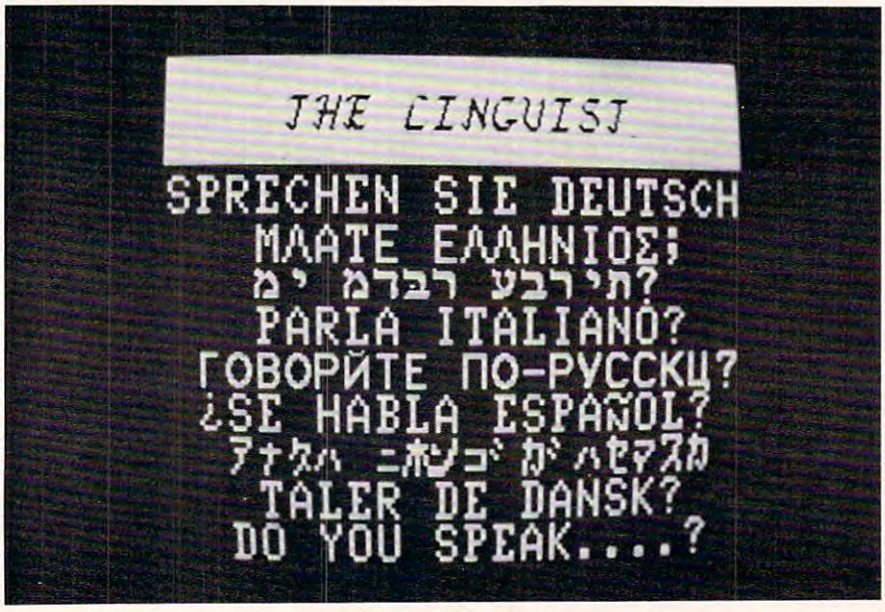
Foreign Language Program For Apple

Synergistic Software announces the release of a new educational software program called The Linguist, which is a general purpose foreign language translation and tutorial program for the Apple II computer. It allows the Apple to correctly print the foreign alphabets used by such languages as Hebrew, Russian, Japanese, Greek, German, plus the Romance languages and English. This unique program can work with words, phrases, definitions, technical terms, or phonetic pronunciations. Thus, it can be used by those learning a foreign language or those who wish to learn English.

The user of The Linguist types in the words, phrases, or definitions he would like to learn. Then the program will test the user on these words using hints. The Linguist will keep score and correct mistakes. If a phonetic pronunciation is desired, the user can input which pronunciation guide he would like to use (from American Heritage Dictionary, the International Phonetic Alphabet, or the Trager-Smith Phonemes). Then a word's correct pronunciation is easy to look up. The Linguist can operate with one or two stored languages with a maximum storage capacity of 4400 words, 2600 definitions, or 2000 foreign phrases.

The Linguist can be used for a number of purposes. If a foreign language teacher would like to drill a class on new vocabulary, The Linguist can be used to teach without supervision. If business people or tourists want to brush up on a foreign language before traveling, The Linguist can store the terms and quiz the person on their definitions and use. The Linguist requires an Apple II Computer, Applesoft, 48K, DOS 3.3. The price is $40.

Synergistic Software
830 North Riverside Drive
Suite 201
Renton, WA 98055
For ordering call (800)426-6505



File II For PET And VIC

File is a general purpose cassette-based file system for the PET/CBM, or VIC-20 computers. It will allow you to construct, sort, maintain, and print out a wide range of data types, such as mailing lists, accounts, book lists, etc. File allows the user to define the record format and is limited only by available memory. Commands include: LOAD, DUMP, PRINT (screen or printer options), ADD, CHANGE, REMOVE, SORT, and more.

Requires minimum of 8K PET/CBM or VIC-20 with 3K expansion cartridge. Expands automatically to available memory. Package includes software on cassette and complete documentation. Price is $9.95.

Complete catalog of products is available. Please specify type of computer.

Kinetic Designs
401 Monument Rd. #171
Jacksonville, FL 32211

New Book Aims At Consumer Protection

Questions concerning how to resolve computer consumer problems led to the publishing of

ATARI

800 16K $639
800 48K $789
400 16K $315

410 Recorder	$ 75
810 Disk Drive	$439
820 Printer	$259
822 Printer	$259
825 Printer	$579
830 Modem	$155
850 Interface	$169
481 Entertainer	$ 79
482 Educator	$125
483 Programmer	$ 55
484 Communicator	$299
853 16K Ram	$ 79

ATARI Software

CS406 Financial Management	$ 55
CX4104 Mailing List	$ 19
CX404 Word Processor	$115
CXL4007 Music Composer	$ 45
Programming 2 & 3	$ 22
Conversational Languages	$ 45
CX4018 Pilot	$ 59
CX405 Pilot	$ 99
CXL4003 Assembler Editor	$ 45
CX8126 Microsoft Basic	$ 67
CX412 Dow Jones Investment	$ 95
CXL4022 Pac-Man (new)	$ 33
CX8130 Caverns of Mars	$ 29
CXL4020 Centipede (new)	$ 33
CXL4006 Super Breakout	$ 32
CXL4008 Space Invaders	$ 32
CXL4009 Computer Chess	$ 32
CXL4010 3-D Tic Tac Toe	$ 26
CXL4011 Star Raiders	$ 35
CXL4012 Missile Command	$ 32
CXL4013 Asteroids	$ 32

New Software for Atari

Frogger	$26
Apple Panic	$23
Raster Blaster	$23
Temple of Apshai	$30
Bishops Square	$23
Graphics Master	$30
Tumble Bugs	$23
Action Quest	$23
Crossfire	$23
Threshold	$30
Computer Stocks & Bonds	$17
Guns of Fort Defiance	$17
Mousekattack	$26
K-Razy Shootout	$36
Jawbreaker	$23
Preppie	$19
Rear Guard	$19
Arcade Pro Football	$26
Bug Attack	$23
Pathfinder	$26
Deadline	$39
Zork I	$29
Zork II	$29
Crypts of Terror	$26
Pool 1.5	$26
Deluxe Invaders	$26
Gorf	$29
Wizard of Wor	$29
Battle Trek	$29
Canyon Climber	$23
Shooting Arcade	$23
Pacific Coast Highway	$23
Clowns and Balloons	$23

For Fast Delivery, send certified or cashier checks, money orders, or direct bank wire transfers. Personal checks allow 2 to 3 weeks to clear. 25% deposit on C.O.D.'s. Prices reflect a cash discount only and are subject to change. Shipping—Software ($2.00 Minimum), Hardware—call. Foreign inquiries invited — add 15% for shipping. Nevada residents add sales tax.

TOP SELLERS
Atari Software

Entertainment

Ghost Hunter	$ 26
Lunar Lander	$ 12
Reversi	$ 23
Gomoku	$ 23
Star Warrior	$ 29
Crush, Crumble, & Chomp	$ 23
Ricochet	$ 15
Empire of the Overmind	$ 26
Tanktics	$ 23
B-1 Nuclear bomber	$ 13
Kayos	$ 26
Match Racers	$ 23
Wiz & Princess	$ 24
Mission: Asteroid	$ 19
Softporn Adventure	$ 23
Ali Baba & the Forth Thieves	$ 24
The Shattered Alliance	$ 29
Galactic Chase	$ 23
Atari World	$ 45

Business & Utilities

Visicalc	$179
Mail Merge	$ 23
Data Perfect	$ 79
Letter Perfect	$105
Text Wizard	$ 69
Datasm 65 2.0	$105
Micropainter	$ 26
The Basic Compiler	$ 69
Color Print	$ 29

Educational

Compu-Read	$ 23
Compu-Math/Fractions	$ 29
Compu-Math/Decimals	$ 29
Vocabulary I	$ 16
Vocabulary II	$ 16
Number Series	$ 16
Analogies	$ 16
Story Builder/Word Master	$ 16
Let's Spell	$ 16
Astro Quotes	$ 16
All APX Software	15% off list

*** SPECIALS OF THE MONTH ***

ELEPHANT DISKS (BOX)	$ 22
HAYES SMARMODEN	$229
AXLON RAMCRAM	$129
MOSAIC 32K RAM	$ 99
RAMDISK (128K)	$429
MICROTEK 32K RAM	$109
AMDEK COLOR I MONITOR	$329
PERCOM DOUBLE DENSITY DRIVE	$679
EPSON MX-100 PRINTER	$729
OKIDATA MICROLINE 80	$329
K-DOS	$ 65
OS/A +	$ 59
BASIC A +	$ 59
FLIP N' SORT DISKETTE BOX (Holds 50 Diskettes)	$ 21
FLIP-SORT CARTRIDGE BOX (Holds 10 Atari Computer Cartridges)	$ 21

YOUR ONE STOP MARKETPLACE FOR ALL YOUR COMPUTER NEEDS
If it is not listed, please ask.

Computer Outlet

Park Place — Upper Level
1095 E. Twain — (702) 796-0296
Las Vegas, Nevada 89109

Call Toll Free **800-634-6766**
We accept Major Credit Cards
Mon.-Sat. 8 A.M.-6 P.M.

commodore

VIC 20 . . . . . $249

VIC1530Datasette	$ 67
VIC1540 Disk Drive	479
VIC1515 Graphics Printer	329
VIC1210 3K Memory Expander	$ 30
VIC1110 8K Memory Expander	$ 52
VIC1011 RS 232C Terminal Interface	$ 43
VIC1112 VIC-1EEE-488 Interface	$ 84
VIC1211 VIC 20 Super Expander	$ 52
VIC1212 Programmers Aid Cartridge	$ 45
VIC1213 Machine Language Monitor	$ 45
VM110 Vic Programmers Ref. Guide	$15

VIC Software

Avenger	$23
Superslot	$23
Super Alien	$23
Jupiter Lander	$23
Draw Poker	$23
Midnight Drive	$23
Spiders of Mars	$39
Meteor Run	$39
Amok	$29
Alien Blitz	$29
Renaissance	$39
Outworld	$29
Cloudburst	$39
Satellites & Meteorites	$39
Skymath	$12
Space Division	$12
Long Division	$12
Super Hangman	$15
3-D Maze	$12
Raceway	$13
Kosmic Kamikaze	$21
The Alien	$21
Simon	$10
Kiddie Pak I	$32
Super Four I	$39
Kiddie Checkers	$ 8
Snakman	$18
Astrobase-2001	$13
Space Barrier	$13
Snake	$10

(Programming Techniques)

Print Commands	$14
For-Next Loops	$14
Graphics	$14
Types of Variables	$14
Data Files	$14
Random Numbers	$14

Educational

Money Addition	$10
Math Whiz	$10
State Capitals	$10
World Capitals	$10
Spelling	$ 8
The Verb	$14
The Adverb	$14
The Adjective	$14
Fraction Reduction	$10
Adding Signed Numbers	$10
Plurals	$ 8
Memory	$10

★ ATARI ★
Programming Techniques
(Santa Cruz — Tricky Tutorials)

Display Lists	$ 17
Horiz/Vert Scroll	$ 17
Page Flipping	$ 17
Basics of Animation	$ 17
Player Missile Graphics	$ 24
Sound	$ 17

The Computer Outlet is an associate of The Computer Learning Center For Children. We are experts in educational technology and can customize educational software curriculums for school districts, individual schools, or for the child at home. Please contact us about your software and equipment requirements and feel free to stop by our school in Las Vegas.

We have one of the world's largest educational software inventories featuring our own Computer Learning Center software.

Ten Little Robots (ATARI)	$12.95
Pre-School Math (ATARI)	$19.95

Consumer Protection for the Microcomputer Owner. The author is Attorney L. J. Kutten.

The 35-page booklet covers the steps to consider before making a purchase. Information about the general Law of Sales is given to advise the Buyer of the legal issues that can arise if he is not careful, such as: When is a sale valid? Are advertised prices binding? What can be done about partial delivery? What and when must the Seller deliver? A general comparison of the local computer store and mail order is given along with a brief introduction to Federal Mail Order Law. There is also a discussion of the different ways to pay for a purchase and some of the problems that each one entails. An introduction into warranties is given along with a discussion of warranty disclaimers. Hints on how to legally reject and revoke a prior acceptance of a product are included. Finally, there are suggestions on how to complain effectively, and various miscellaneous hints every computer purchaser will find useful. The booklet aids not only the consumer, but also the seller in becoming more effective.

The booklet is available through the mail for $15.00. There is an additional $3.00 charge for C.O.D. orders. Missouri residents should add 4.625% sales tax. Send check or money order to:

L. J. Kutten, Attorney at Law
201 S. Central Ave., POB 16185
St. Louis, MO 63105

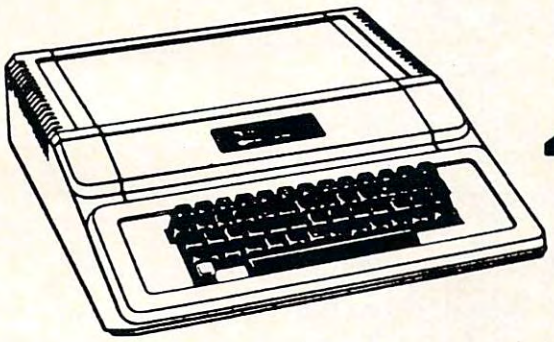
New Product releases are selected from submissions for reasons of timeliness, available space, and general interest to our readers. We regret that we are unable to select all new product submissions for publication. Readers should be aware that we present here some edited version of material submitted by vendors and are unable to vouch for its accuracy at time of publication. ©

BELL & HOWELL
MADE EXCLUSIVELY FOR BELL & HOWELL BY
apple computer inc.

PRICE BREAKTHROUGH

apple computer
Sales and Service

APPLE II PLUS 48K



48K ONLY **$1089**
APPLE II PLUS

BEST PRICES U.S.A.

AUTHORIZED DEALER AND LEVEL 1 SERVICE CENTER

DISK II DRIVE

WITH CONTROLLER CARD **$ 540.00**

CALL OR WRITE FOR A COMPLETE SOFTWARE LIST

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

VISA
MasterCard

COMPUTER AGE

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

Advertisers Index

COMPUTE! Publications

COMPUTE!

For Fastest Service,
Call Our **Toll-Free**
US Order Line
800-345-8112
In Pennsylvania call 800-662-2444

My Computer Is:

☐ PET ☐ Apple ☐ Atari ☐ OSI ☐ VIC-20 ☐ TI 99/4A ☐ Sinclair ZX-81
☐ Radio Shack Color Computer ☐ Other _____ ☐ Don't yet have one...

☐ $20.00 One Year US Subscription
☐ $36.00 Two Year US Subscription
☐ $54.00 Three Year US Subscription

(Readers outside of the US, please see our foreign readers subscription card or inquire for rates).

Name _____

Address _____

City _____ State _____ Zip _____

☐ Payment Enclosed ☐ VISA
☐ MasterCard ☐ American Express
Account No. _____ Expires ____/____

8 9 10 11 12

COMPUTE!

Foreign Readers

Subscription rates outside the US:

☐ $25.00 Canada FI=2
☐ $38.00 Europe/Air Delivery FI=3
☐ $48.00 Middle East, North Africa, Central America/Air Mail FI=5
☐ $88.00 South America, South Africa, Australasia/Air Mail FI=7
☐ $25.00 International Surface Mail (lengthy, unreliable delivery) FI=4,6,8

Name _____

Address _____

City _____ Postal Code _____

Country _____

Payment must accompany this card.
Payment in US Funds drawn on a US Bank; International Money Order; or charge card: ☐ VISA ☐ MasterCard ☐ American Express
Account No. _____ Expires ____/____

8 9 10 11 12

COMPUTE! Books

Quan.	Title	Price	S/H	Total
____	**The Beginner's Guide to Buying A Personal Computer**	$ 3.95	+ $1.00*	____
____	**COMPUTE!'s First Book of Atari**	12.95	+ 2.00*	____
____	**Inside Atari DOS**	19.95	+ 2.00*	____
____	**COMPUTE!'s First Book of PET/CBM**	12.95	+ 2.00*	____
____	**Programming the PET/CBM**	24.95	+ 3.00**	____
____	**Every Kid's First Book of Robots and Computers**	4.95	+ 1.00*	____
____	**COMPUTE!'s First Book of VIC**	12.95	+ 2.00*	____

*$4.00 / **$9.00 For air mail outside US.

For Fastest Service
Call Our **TOLL FREE**
US Order Line
800-334-0868
In NC call 919-275-9809

All orders must be prepaid (money order, check, or charge). All payments must be in US funds. NC residents add 4% sales tax.

☐ Payment enclosed

Please charge my:
☐ VISA ☐ MC ☐ Am. Express
Acc't No. _____
Expires ____/____

Name _____

Address _____

City _____ State _____ Zip _____

Country _____

Allow 4-5 weeks for delivery.

The Editor's Feedback:

Computer: ☐ PET ☐ ATARI ☐ APPLE ☐ VIC-20 ☐ OSI ☐ OTHER _____

Are you a **COMPUTE!** Subscriber? ☐ Yes ☐ No I would like to see:

☐More	☐Fewer	Specific applications programs.	☐More	☐Fewer	Games.
☐More	☐Fewer	BASIC programs.	☐More	☐Fewer	Reviews of game software.
☐More	☐Fewer	Machine language programs.	☐More	☐Fewer	Reviews of business software.
☐More	☐Fewer	Tutorials.	☐More	☐Fewer	Reviews of educational software.
☐More	☐Fewer	Educational articles.			
☐More	☐Fewer	Detailed explanations of programs.	☐More	☐Fewer	Reviews of hardware.

What do you like best about **COMPUTE!**?

What do you like least?

8 9 10 11 12

Place
Stamp
Here

COMPUTE! Magazine
Post Office Box 5406
Greensboro, NC 27403

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 236 HOLMES, PA

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE! Magazine
P.O. Box 636
Holmes, PA 19043

Place
Stamp
Here

COMPUTE! Magazine
Post Office Box 5406
Greensboro, NC 27403

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 2312 GREENSBORO, NC

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE! Books
Post Office Box 5406
Greensboro, NC 27403

The Library of Computer and Information Sciences

Please accept my application for trial membership and send me the ENCYCLOPEDIA OF COMPUTER SCIENCE (44900-3) billing me only $2.95. I agree to purchase at least three additional Selections or Alternates over the next 12 months. Savings range up to 30% and occasionally even more. My membership is cancelable any time after I buy these three books. A shipping and handling charge is added to all shipments.

No-Risk Guarantee: If you are not satisfied—for any reason—you may return the Encyclopedia of Computer Science within 10 days and your membership will be canceled and you will owe nothing.

Name_____

Name of firm_____
(if you want subscription sent to your office)

Address_____ Apt. _____

City_____

State_____ Zip_____
(Offer good in Continental U.S. and Canada only. Prices slightly higher in Canada.)

Compute 8/82 7-BL4

The Library of Computer and Information Sciences

Please accept my application for trial membership and send me the ENCYCLOPEDIA OF COMPUTER SCIENCE (44900-3) billing me only $2.95. I agree to purchase at least three additional Selections or Alternates over the next 12 months. Savings range up to 30% and occasionally even more. My membership is cancelable any time after I buy these three books. A shipping and handling charge is added to all shipments.

No-Risk Guarantee: If you are not satisfied—for any reason—you may return the Encyclopedia of Computer Science within 10 days and your membership will be canceled and you will owe nothing.

Name_____

Name of firm_____
(if you want subscription sent to your office)

Address_____ Apt. _____

City_____

State_____ Zip_____
(Offer good in Continental U.S. and Canada only. Prices slightly higher in Canada.)

Compute 8/82 7-BK7

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD
FIRST CLASS PERMIT NO. 230 RIVERSIDE, N J

POSTAGE WILL BE PAID BY ADDRESSEE

The Library of Computer and Information Sciences
Riverside, New Jersey 08075

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD
FIRST CLASS PERMIT NO. 230 RIVERSIDE, N J

POSTAGE WILL BE PAID BY ADDRESSEE

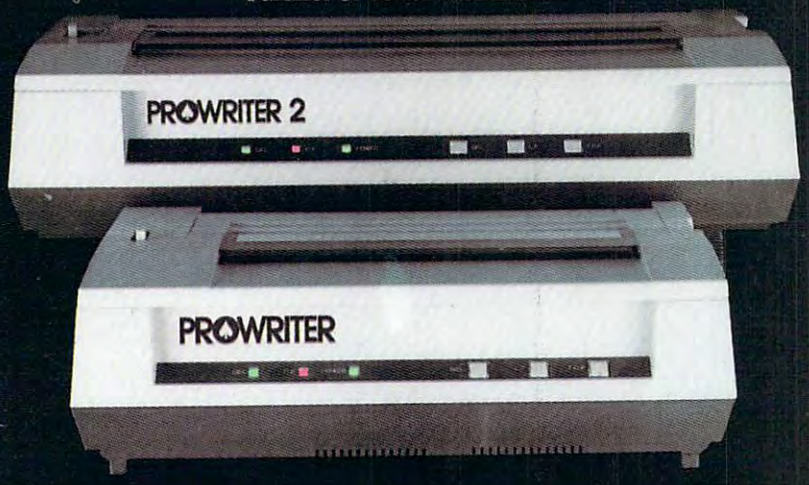
The Library of Computer and Information Sciences
Riverside, New Jersey 08075

THE LEADING EDGE IN PRINTERS

Finally, there's one full family of printers that covers every business or word processing application—all from C. Itoh, a company known for packing more product into less price; and all distributed exclusively by Leading Edge, a company known for searching out and providing that very thing. Which means that one call to one source can get you any printer, any time you need it, for any purpose.

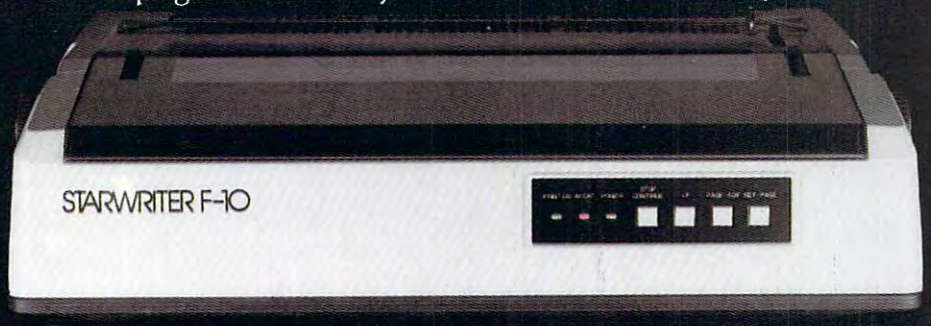
THE PRO'S.

The Prowriters: business printers—and more. The "more" is a dot-matrix process with more dots. It gives you denser, correspondence quality copy (as opposed to business quality copy, which looks like a bad job of spray-painting).
Prowriter: 120 cps. 80 columns dot matrix compressable to 136. 10" carriage. Parallel or serial interface.
Prowriter 2: Same as Prowriter, except 15" carriage allows full 136 columns in normal print mode. Parallel or serial interface.



THE STAR.

The Starwriter F-10. In short (or more precisely, in a sleek 6" high, 30-pound unit), it gives you more of just about everything—except bulk and noise—than any other printer in its price range. It's a 40 cps letter-quality daisy-wheel with a bunch of built-in functions to simplify and speed up word processing. It plugs into almost any micro on the market, serial or parallel.



AND THE MASTER.

The Printmaster F-10. Does all the same good stuff as the Starwriter except, at 55 cps, the Master does it faster.

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.

THE COMMODORE COMPUTERS

"FROM $300 TO $1995, THEY COST LESS AND GIVE YOU MORE FOR YOUR MONEY. READ OUR CHART."

—William Shatner

The idea of a computer in every office and home used to be science fiction. Now it's becoming a reality. The question is, with so many to choose from, which computer should you buy? When you consider the facts, the clear choice is Commodore.

COMPARE OUR $995 COMPUTER

FEATURES	COMMODORE 4016	APPLE II	IBM
Base Price	$995	$1,330	$1,565
12″ Green Screen	Standard	299	345
IEEE Interface	Standard	300	NO
TOTAL	$995	$1,929	$1,910
Upper & Lower Case Letters	Standard	NO	Standard
Separate Numeric Key Pad	Standard	NO	Standard
Intelligent Peripherals	Standard	NO	NO
Real Time Clock	Standard	NO	NO
Maximum 5½″ Disk Capacity per Drive	500K	143K	160K

Prices are as of the most recent published price lists, September, 1981 and approximate the capabilities of the (16K) PET® 4016. Disk Drives and Printers are not included in prices. Models shown vary in their degree of expandability.

Many experts rate Commodore Computers as the best desk-top computers in their class. They provide more storage power—up to 1,000,000 characters on 5¼″ dual disks—than any systems in their price range. Most come with a built-in green display screen. With comparable systems, the screen is an added expense. Our systems are more affordable. One reason: we make our own microprocessors. Many competitors use ours. And the compatibility of peripherals and basic programs lets you easily expand your system as your requirements grow. Which helps explain why Commodore is already the No. 1 desk-top computer in Europe with more than a quarter of a million computers sold worldwide.

WE WROTE THE BOOK ON SOFTWARE.
The Commodore Software Encyclopedia is a comprehensive directory of over 500 programs for business, education, recreation and personal use. Pick up a copy at your local Commodore dealer.

FULL SERVICE, FULL SUPPORT.
Commodore dealers throughout the country offer you prompt local service. In addition, our new national service contract with TRW provides nationwide support. Visit your Commodore dealer today for a hands-on demonstration.

Commodore Computer Systems
681 Moore Road
King of Prussia, PA 19406

Canadian Residents:
Commodore Computer Systems
3370 Pharmacy Avenue
Agincourt, Ontario, Canada, M1W 2K4

Please send me more information. CO-8

Name _____

Company _____ Title _____

Address _____

City _____ State _____ Zip _____

Telephone _____

Interest Area _____

☐ Business ☐ Education ☐ Personal

commodore COMPUTER

www.commodore.ca