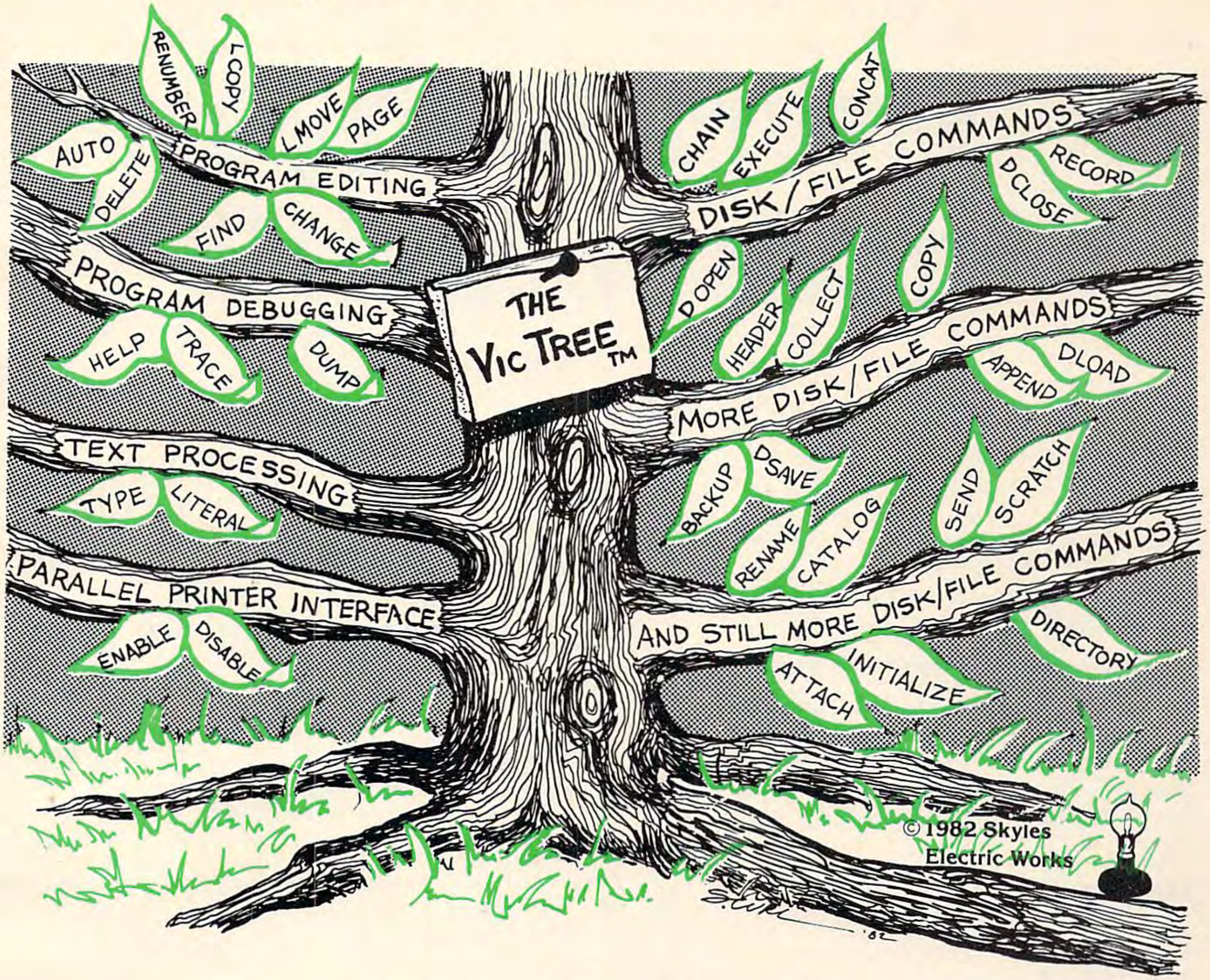
Skyles Electric Works Presents



The VicTree™

...Leaves your new Commodore 64 (or Vic 20) with 42 additional commands.
...Branches out to most BASIC 4.0 programs.
...Roots into most printers.

New from Skyles: the VicTree, a coordinated hardware and software cartridge that allows your Commodore 64 to branch out in unbelievable directions and makes it easier than ever to do BASIC programming, debugging and to access your disk. And the new VicTree provides routines to interface the Commodore 64 to all Centronics Parallel printers with an inexpensive cable. 8kb of ROM—4kb for the BASIC commands, 4kb for disk commands. Perfect not only for the new Commodore 64 but also for the Vic 20. Unbelievably simple to use with a 100+ page manual with many many program examples. Easy to install, plugs into cartridge port, the VicTree gives you all the additional BASIC 4.0 commands to allow most BASIC 4.0 programs to work on your new Commodore 64 and Vic 20.

And now with the BASIC command Commodore forgot!

Now only $89.95 or **$109.95** complete with Centronics standard printer cable. (Cable alone $29.95.) Available now from your local dealer or order through your Visa or MasterCard toll free: (800) 227-9998 (California, Canada, Alaska, Hawaii: (415) 965-1735) or send check or money order directly to:

Skyles Electric Works

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

String Arrays In Atari BASIC

Stephen Levy

This tutorial demonstrates an easy way to create string arrays in Atari BASIC. The author provides example programs and also includes a few handy ideas you can use with other programs.

If you want string arrays on your Atari computer, you'll just have to purchase Atari's Microsoft BASIC disk. A common belief, but not entirely true. Although string arrays are more easily handled with Microsoft BASIC, they can be created with Atari BASIC.

Creating The Array

What we actually create here is a long string which will hold all the elements of the array. So that the array will not contain any garbage, we must clean it out before using it.

There are two ways to accomplish this. You could simply DIMension a string to 1000 and then fill it with "*" using a FOR-NEXT loop.

```
90 DIM B$(1000)
100 FOR A=1 TO 1000
110 B$(A,A)="*"
120 NEXT A
130 PRINT B$
```

Here's another way to do the same thing a little differently and much more efficiently:

```
100 DIM B$(1000)
110 B$="*":B$(1000)="*":B$(2)=B$
120 PRINT B$
```

A lot faster, isn't it? You can use this method whenever you want to fill a large string with the same character. This is exactly what we must do to begin creating our string array. But with this next program, we need to fill the string with blanks.

Type in and RUN the following program. When you are asked for names, enter the names of ten friends, pressing RETURN after each. As written the program will allow only names with up to ten letters.

```
100 DIM ARRAY$(100),ELEMENT$(10):PRI
    NT CHR$(125)
110 ARRAY$=" ":ARRAY$(100)=" ":ARRAY
    $(2)=ARRAY$
120 FOR A=1 TO 10
130 PRINT "NAME FOR ARRAY$(";A;") PL
    EASE";:INPUT ELEMENT$
140 ARRAY$(A*10-9,A*10)=ELEMENT$
150 ELEMENT$=" ":NEXT A
160 PRINT
200 FOR A=1 TO 10
210 PRINT "ARRAY$(";A;") IS ";ARRAY$
    (A*10-9,A*10):NEXT A
300 TRAP 340
310 PRINT :PRINT "GIVE THE NUMBER (1
     TO 10)"
320 PRINT "OF THE ARRAY YOU WISH TO
    SEE";:INPUT A
330 PRINT ARRAY$(A*10-9,A*10):GOTO 3
    10
340 PRINT CHR$(253):GOTO 300
```

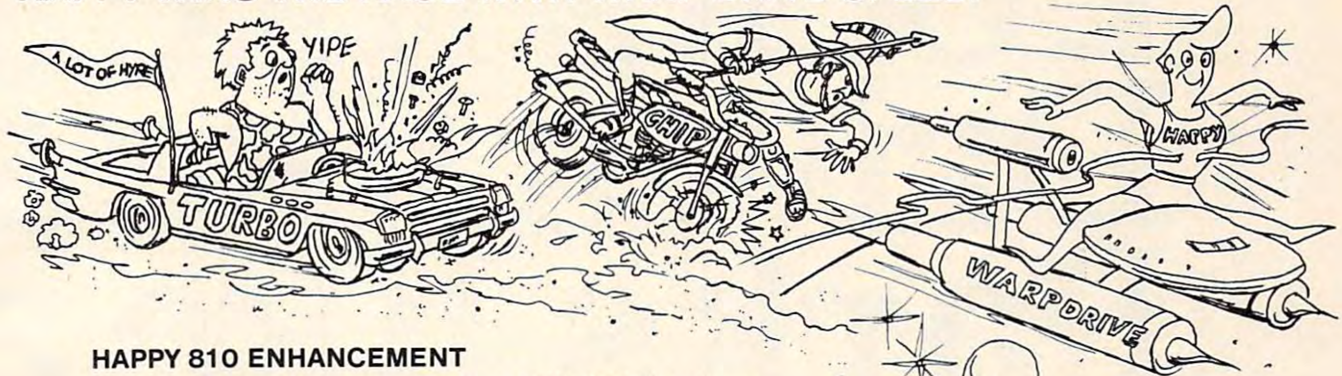
Notice that the program sets up an array with ten elements and allows you to pick from any of the ten.

How It Works

Line 100 DIMensions the array and clears the screen. Line 110 fills the array with blanks and line 120 establishes a loop so you can enter ten names. Line 130 gets your input.

Line 140 is the heart of the creation of the array. Within the parentheses the computer is told what *part* of the string should hold your input string ELEMENT$. The first time through, A=1; therefore, ARRAY$(A*10−9,A*10) will mean ARRAY$(1,10) or the first ten positions in the string. When A = 2, we place ELEMENT$ in the eleventh to twentieth positions (2*10−9 = 11 and 2*10=20). This continues until the string is full.

HAPPY WINS THE RACE WITH WARP DRIVE SPEED!



HAPPY 810 ENHANCEMENT

- The only change needed to run all WARP DRIVE software
- Plug in P.C. board requires no permanent modifications
- Proven reliable in thousands of installations, reduces disk drive wear
- Comes completely assembled and tested, just plug in and use
- Full one year parts and labor guarantee, compatible with existing software
- High quality printed circuit board with gold connectors

NO ONE ELSE HAS THIS PERFORMANCE
Unenhanced whole disk (ATARI rev B format) read time: 112 seconds
Unenhanced whole disk (ATARI rev C fast format) read time: 89 seconds
ENHANCED 810 whole disk (any format) read time with standard software: 68 seconds
ENHANCED 810 whole disk (any format) read time with WARP DRIVE software: 43 seconds
Standard software whole disk write and verify time: 238 seconds
WARP DRIVE software whole disk write and verify time: 62 seconds

NEW HAPPY WARP DRIVE SOFTWARE

WARP SPEED HAPPY BACKUP PROGRAM

- Completely automatic: nothing to figure out, insert disks and press return
- Only program on the market guaranteed to backup any disk
- Can write to a blank disk: format write and verify in one operation
- Automatic program tracing: copies only the tracks that are used
- Efficient memory utilization: reduces the number of disk insertions
- Requires only one ENHANCED disk drive, backups will work on a standard drive

WARP SPEED MULTI DRIVE HAPPY BACKUP PROGRAM

- Same features as above plus support of multiple ENHANCED drives
- Can be used with up to 4 ENHANCED drives
- Source and all destination drives read and write in parallel
- Format write and verify 3 complete disks in less than 3 minutes

WARP SPEED HAPPY COMPACTOR PROGRAM

- Reduces the number of disks required to backup your library
- Combines up to 8 self booting disks into 1 disk with a menu
- Compacted disks run only on an ENHANCED drive
- Pays for itself by saving on disks
- Single or dual ENHANCED drive operation

HAPPY WARP DRIVE DOS

- Improves ATARI DOS 2.0S to use warp speed reading and write with verify
- Use all features of BASIC, PILOT, FMS, and DUP at top warp speed
- Warp speed I/O software module available separate from DOS

HAPPY WARP DRIVE SECTOR COPY PROGRAM

- Standard format whole disk read, write and verify in 105 seconds
- Use with sngle or dual drives, mix ENHANCED and NON-ENHANCED drives

HAPPY CUSTOMIZER PROGRAM (sold separately $99.95)

- Creates custom format disks of any specification
- Any type bad sector, duplicate sector numbers, or interleave
- Easy to use but requires an advanced level user to interpret the results

REVIEWED IN POPULAR MAGAZINES

A.N.A.L.O.G. COMPUTING—July/August 1983 *"...The installation instructions for the Happy 810 Enhancement are among the best I have ever seen. ...The Happy 810 Enhancement is one of the most powerful hardware modifications available to ATARI computer owners."*

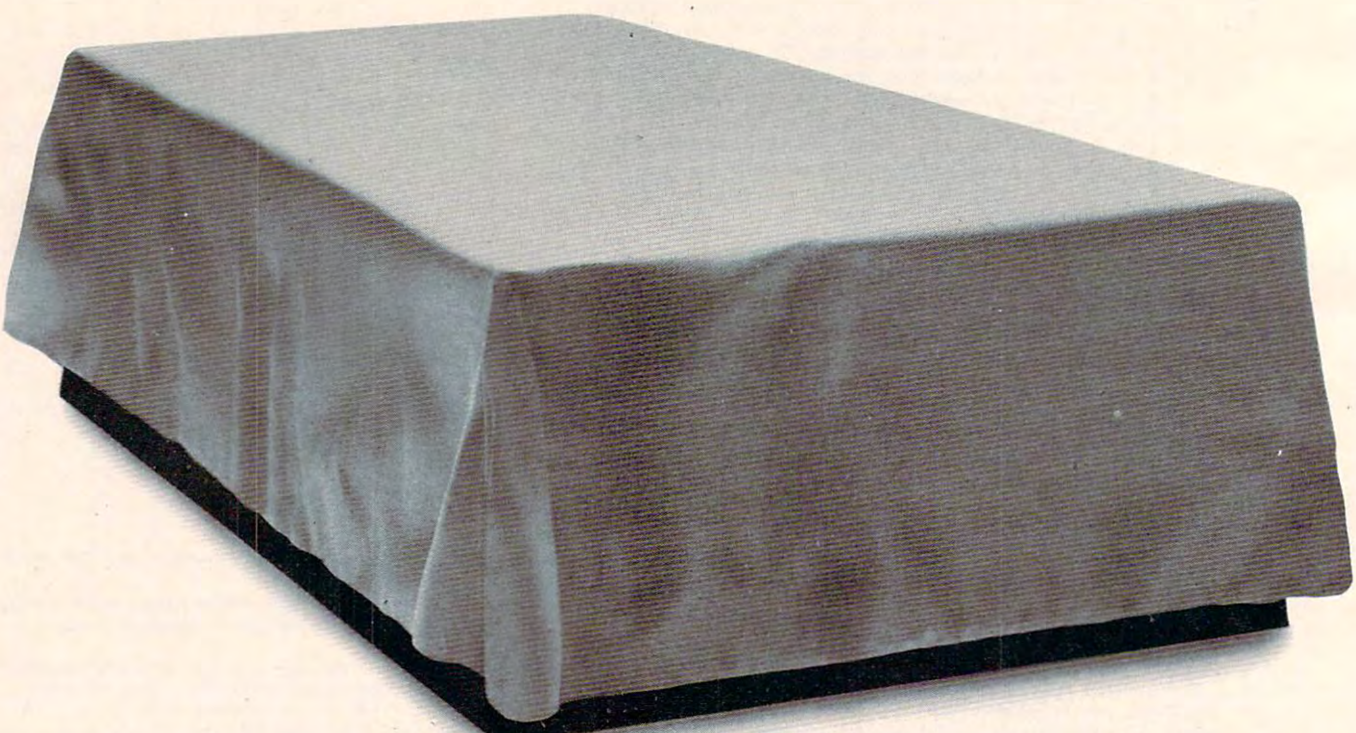
ANTIC—July 1983 *"The difference between a normal ATARI 810 disk drive and one equipped with Happy is like the contrast between mass transit and the automobile. A car costs you more initially, but improves the quality of your life. Similarly, if you use your disk drive a lot, installing Happy will markedly enhance your programming life."*

SPECIAL SUGGESTED RETAIL PRICE BEFORE DECEMBER 31, 1983: *Get the HAPPY 810 ENHANCEMENT with the single and multi drive HAPPY BACKUP PROGRAM, plus the HAPPY COMPACTOR PROGRAM, plus the HAPPY DRIVE DOS, plus the HAPPY SECTOR COPY, all with WARP DRIVE speed, including our diagnostic for $249.95. Existing registered ENHANCEMENT owners may upgrade to WARP DRIVE speed for $15.00 with no hardware changes.!*

Price includes shipping by air mail to U.S.A. and Canada. Foreign orders add $10.00 and send an international money order payable through a U.S.A. bank. California orders add $16.25 state sales tax. Cashiers check or money order for immediate shipment from stock. Personal checks require 2-3 weeks to clear. Cash COD available by phone order and charges will be added. No credit card orders accepted. ENHANCEMENTS for other ATARI compatible drives coming soon, call for information. Please specify -H model for all drives purchased new after February 1982, call for help in ENHANCEMENT model selection. Dealers now throughout the world, call for the number of the dealer closest to you.

HAPPY COMPUTING • P.O. Box 32331 • San Jose, California 95152 • (408) 251-6603

Looks like a
Drives like a
Parks like a



For additional information, call 1-800-334-4387. In California, call 1-800-544-4387.

Ferrari. Rolls. Beetle.

Introducing the all-new 1984 Indus GT™ disk drive. The most advanced, most complete, most handsome disk drive in the world.

A flick of its "Power" switch can turn an Atari into a Ferrari.

Or an Apple into a Red Hot Apple.

Ask your favorite computer dealer to unveil the remarkable Indus GT for you. Test drive it. Then, get ready to drive home a world-class driving machine.

Looks Like a Ferrari.

The Indus GT is incredibly sleek. Only 2.65" high! Its smoked-glass dust-cover front-loading front-end width is a perfectly proportioned 6.27." Eye-catching. Distinctively European-Gran styled.

Touch its LED-lit CommandPost™ function controls. Lay a finger on its AccuTouch™ buttons. Feel its elegant, fluid form.

Admire its smooth, sculptured, masculine lines. Wonder at its ingenius slimline engineering.

Marvel at how responsive it makes *every* Atari or Apple home computer.

Drives Like a Rolls.

Nestled into its soundproofed 10.4" length chassis is the quietest and most powerful disk drive power system money can buy. At top speed (which it can handle all day long), it's virtually unhearable. Whisper quiet.

Flat out, the GT will drive your Atari track-to-track 0-39 in *less than one second!* Increasing data transfer an incredible 400%.

Built-in, low-friction GT CruiseControl metal band head-positioning assures Apple and Atari owners of trouble-free performance. So you can finish every project you enter.

And quietly tucked into every 1984 Indus GT disk drive is the exclusive GT DrivingSystem™ of software programs (included as standard equipment):

The GT DrivingSystem allows you to shift smoothly into world-class word processing with the GT *Estate WordProcessor.*™ And your dealer will describe the two additional new GT Driving System programs that allow you to advance your driving skills. Instantly. Smoothly. Easily.

The Indus GT and GT DrivingSystem are covered with the GT PortaCase.™ Which doubles as a 20-disk storage file when you're at home. Or as a portable case to flaunt your good taste and judgement when you travel. Safely. Smartly.

Parks Like a Beetle.

And it's sticker-priced like a Beetle, too. $449 for Atari. $329 for Apple.

The all-new, 1984 Indus GT.

Its small size makes it easy to park. And its low price makes it easy to buy.

So see and test drive the GT at your nearest computer dealer today.

The drive will be well worth it.

INDUS GT™

The all-new 1984 Indus GT Disk Drive.

The cover comes off at your computer dealer's showroom.

Line 210 does the same thing, but in reverse order: it reads ARRAY$ and prints the proper part to the screen. Line 330 does the same thing, but only for the part of the string you request.

Try this: RUN the program and enter any ten names. Then press BREAK. Type PRINT ARRAY$ without a line number, press RETURN, and see what happens.

Now RUN the program again, but simply press RETURN without entering anything for the names. Although there appears to be nothing in ARRAY$, it is actually filled with blanks. Type PRINT ARRAY$ again and see what happens.

There are also a few other techniques here which may be helpful. Lines 300 and 340 prevent the program from crashing when an incorrect INPUT is entered. TRAP 340 sends the program to line 340 instead of printing ERROR 8 LINE 320 when you enter a Q (or whatever) instead of a required number between one and ten. PRINT CHR$(253) rings the buzzer, just as PRINT CHR$(125) in line 100 clears the screen.

With these techniques, you now should be able to use string arrays in your own programs. ©

Apple Sounds — From Beeps To Music

Part 2

Blaine Mathieu

In the conclusion of this two-part series, the author combines the ideas and programs from Part 1 and presents the "Apple Music Writer." An effective tool for composing or reproducing songs, this utility is also easy to use because of its great variety of commands. There's a thorough discussion of how to use each command.

"Apple Music Writer" is a program which will allow any Apple owner to easily reproduce his or her favorite songs. When you run the program, the first thing you'll notice is the title, and then you'll hear part of a tune that you may recognize. After the tune stops, you will be prompted by the word COMMAND? and a flashing cursor. At the top of the screen you should see a list of the possible commands and corresponding letters. On the right side of your screen you should see a list of note names with corresponding values.

It's important that you understand and know how to use the commands, so let's review them in some detail, in the order that they appear on the screen. These commands are usable only from the COMMAND? mode; you must also RETURN after each command. You may want to experiment with them as we go along.

The Commands

A = ADDNOTE. This command will let you begin your music file (song) and keep adding to it. Every time you press A (and RETURN) you will be prompted to enter the note, a comma, and the duration. For example:

```
NOTE#1
NOTE,DURATION 128,200
```

The maximum note value is 255 (actually 0 = 256). The same is true for the duration value. After you've entered your values, you will hear what the new note will sound like in the song.

E = EDIT. If you've made a mistake, you can fix it by typing E (and, as always, RETURN). You will then be asked the number of the note you want to edit. If the note you want to edit is not part of the music file, you will be reprompted for the note number. If you entered a valid note number, you will be given the old values for that note and prompted for new values. The same rules apply for entering data as in ADDNOTE. Let's say you want to edit note number one and replace the old values with new ones of 64 and 200:

```
COMMAND? E
EDIT NOTE#1
NOTE#1 OLD: NOTE=128 DUR=200
NOTE,DURATION: 64,200
```

P = PLAY. Typing P will put you into Play mode. This will play your song and print it to the screen at the same time. Because it is both listing and playing your music file, the playing will not be at the same speed as in your program. It will be slower and more pronounced. After entering P you will be prompted for the starting and ending note to Play/list. If you just press RETURN instead of entering values, the whole song will be played (defaults will be set; D is the default).

S = SAVE. This command will SAVE your music file to disk. First you will be prompted for a filename, which will be the name used when the file is SAVEd. Then you'll be prompted for the number of the first and last note of your file that you want saved to disk. The next question is FOR FUTURE ADDITION? A little explanation is in order here. There are two types of files which can be produced with this command. If you answer Y to the above question, a file will be created that can be reloaded into Apple Music Writer at any time. You should use this option if you feel you may want to add more notes or edit your song at a later date. If you enter N, a file will be created that you can easily turn into a BASIC program that will play your song when run.

If you answer the FUTURE ADDITION? question with an N, you will be asked for the starting line number of your soon-to-be-created BASIC music program. Then you will be asked if you want a FULL LOADER PROGRAM? If you answer Y, the BASIC program created will include the necessary information so that when your new program is RUN, the machine language "Note Producer" (see Part 1) routine will be POKEd in. If you answer N, the routine will not be included. You would answer N if the program you wanted to add the music to already included some sort of "Note" routine (the routine found in Program 5 of Part 1 of "Apple Sounds – From Beeps To Music").

Finally, you will be prompted to check for errors. If everything is all right, enter Y and the file will be SAVEd. If you enter N, you have to repeat the entire SAVE process. Here is an example of what the average SAVE command might encompass:

```
COMMAND? S
(Screen is cleared)
FILENAME? SONG.1
STARTING NOTE NUMBER: 2
ENDING NOTE NUMBER: 10
FOR FUTURE ADDITION? N
STARTING LINENUMBER: 100
FULL LOADER PROGRAM? Y
IS EVERYTHING OK? Y
```

Your music file would now be SAVEd under the filename SONG.1. The file would consist of notes two through ten, and the generated program would start at line 100. The generated program would include the machine language "Note" routine.

L = LOAD. If you answered Y to the FOR FUTURE ADDITION? question back in the SAVE command, you can LOAD an old music file back into the computer. The catch is that you will lose any data that you entered into the computer beforehand. If you don't want to lose your data, then answer N to the question about losing your data. Just enter the appropriate filename, and you can manipulate or add to your data once again.

N = NORPLAY. As mentioned earlier, when you P (Play/list) your song, it will play at a slower speed because it has to list the note values at the same time. To alleviate that problem, you can use the NORmal PLAY command. This will play your song in the same tempo as it will normally be played by your generated program. Just enter the proper values (or defaults will be used) and listen.

D = DELETE. Upon entering D from the COMMAND? mode, you will be asked which note or notes you want to delete. If you hit RETURN after the first question without typing anything else, the default will be used and the last note in the music file will be deleted. If you enter a value for the first question, you will be asked the number of the last note up to which you want to delete. The appropriate notes will then be deleted, and you're back to the COMMAND? mode.

I = INSERT. This command is the exact opposite of the Delete command. Simply answer the few setup questions and enter the data. Note: You cannot leave the Insert mode until you have entered all the data you specified you were going to enter.

R = RESTART. This command lets you start over with a clean slate beginning with note number one.

C = CATALOG will return a fairly standard DOS catalog.

Q = QUIT. Use this command to exit the program cleanly. You will lose all your data that hasn't been SAVEd to disk. If you quit by accident, a GOTO 200 will usually let you reenter the program with no data lost.

. = DOS. What this means is that typing a period followed by any normal DOS command will execute that command. A common use for this might be:

```
COMMAND? .DELETE FILENAME
```

Caution: Certain DOS commands will cause the Apple Music Writer to cease functioning, thus causing a loss of data. Take care.

H = HARD. If you have a printer connected to your Apple, you can get a simple hard copy of your music file by entering H from the COMMAND? mode. Note: You may have to edit lines 1210 and 1220 to accommodate different printers.

Hints For Easier Use

Saving. One good idea is to save two copies of your music file to the disk. One copy should be done in the FUTURE ADDITION? mode so you can edit or add to it at a later date. If you wish, the other copy can be done in the *create program*, or FUTURE ADDITION? N mode. Always remember to use a different filename.

Tempo. When you enter your durations, remember that if your quarter note has a value of 50, your half note will have a value of 100 and so on. You should set a plan of what duration you want a certain type of note to be and work from there. Rests are done with a note value of one.

Limits. The number of notes you can have in one song is limited. For our purposes the number is 500, but by changing the value of L in line 120, this limit can be raised.

Notes. The note listings on the side of the screen are especially helpful if you are transposing sheet music to disk. The numbers listed are for the middle octave. For the higher octave, divide the number by two; and for the lower octave, multiply the number by two. For example, the note F could be represented by the numbers 36, 72, and 144. You can also make a separate list of all the notes and their numbers. Remember, F# is the same as G-flat and so on. Also, once again, the number zero is equivalent to the number 256.

ADVERTISEMENT

Educational News

Orange Plus Educational News located at 23801 Calabasas Road/Suite 2050/Calabasas, CA 91302/(213) 999-5210

Legal Apple^Type^ Compatible

New Computer runs CP/M & Apple^Type^ Software

CALABASAS — Now instructors can teach Apple-type programs such as LOGO in one class and professional-style word processing in another without any hardware changes. Dual microprocessors (Z80-A & 6502, two computers in one case), with the radical new Orange Plus development the "EuroROM", allows the machine to read/write/work with Apple-type software as well as CP/M programs and access either CPU via the keyboard.

The ***"ORANGE+TWO"™*** is a brand new direction in the evolution of the personal computer. ORANGEFORTH-83, a derivative of Fig-FORTH — a readily available public domain language, is resident in the ROM*. Also included is CP/M 3.0, Digital Research's latest CP/M version. There is also a built-in disk drive controller for two Apple-type drives, a cassette interface, joystick port, color graphics and ASCII keyboard with numeric keypad. For expert word processing, the keyboard features upper and lower case (lower case characters are true descenders) with auto repeat.

This new breed of computer is a breakthrough for educators and school systems throughout the world. The ability to run both Apple-type and CP/M software on the same machine relieves financially-pressed educators from expensive equipment burdens, allowing them to spend more money where it counts...**on the teachers.**

*Available on disk at extrta cost.

THE LEASING ALTERNATIVE

CALABASAS — Through select leasing companies, Orange Plus Computer Systems will offer the following lease programs to qualified corporations and educational institutions: An ***"ORANGE+TWO"™*** computer with a green monitor and one disk drive for only $59.55 per month...36 month closed end lease. Also included is a full maintenance program and all revisions and upgrades that may be available during the term of the lease. Subject to credit approval. Call for additional information.

DISTRIBUTORS: CALL (213) 999-5210

CALABASAS — Due to the tremendous amount of individual inquiries and dealer applications, qualified stocking distributors are needed. Select areas available!

MORE FOR LESS

The ***"ORANGE+TWO"™*** 64K computer includes these standard features:

Built-in CP/M 2.2 compatibility	**no charge**
Digital Research's CP/M 3.0	**no charge**
Built-in disk drive controller for two Apple-type drives	**no charge**
Numeric keypad with separate return key	**no charge**
110/220 volt switch selectable power supply, 50-60 Hz	**no charge**
Full function ASCII keyboard with auto repeat	**no charge**
Cassette interface	**no charge**
Fully grounded metal base plate	**no charge**
Adjustable audio volume control	**no charge**
ORANGEFORTH-83, Z80-A FORTH language, resident in ROM	**no charge**
Z80-A CPU, a second computer	**no charge**
6 slot double sided logic board	**no charge**
Bank switchable RAM, fully socketed, expandable to 256K	**no charge**
Programmable 2764 EPROMs	**no charge**
TOTAL: *"ORANGE+TWO"™*	**$1095.00**

*Compare these features included on the **"ORANGE+TWO"™** with the competition's over $2000 for equivalent product.*

MORE PRODUCTS FROM ORANGE PLUS COMPUTER SYSTEMS TO ENHANCE YOUR *"ORANGE+TWO"™*, APPLE II/II PLUS, OR FRANKLIN ACE 1000

Disk-based ORANGEFORTH-83 language with full documentation	**$99.95**
Digital Research's™ CBASIC with disk and documentation	**$49.95**
Orange Plus KoalaPad™ Touch Tablet w/Micro Illustrator™	**$124.95**
Orange Plus Joystick (works on "ORANGE+TWO", Apple II, II+, & //e)	**$29.95**
Orange Plus self-centering joystick	**$39.95**
10MB (Formatted) 5¼" half-height Winchester Hard Disk Drive	**$1495.00**
Controller & Interface for Hard Disk**	**$395.00**

***Includes everything necessary to be installed in **"Orange+Two"**, Apple or Franklin Computer*

The above are suggested retail prices. Prices may vary from state to state.

Collins International Trading Corporation,
23801 Calabasas Road, Suite 2050, Calabasas, CA 91302
(213) 999-5210

Exec. In order to use a program that you made in the FUTURE ADDITION? N mode, you must EXEC it. EXEC is a DOS command that prints a sequential text file to the screen as if it were typed from the keyboard. In this way, you can EXEC your file and RUN it as a BASIC program. Later on, you can SAVE it. Another feature is that you can LOAD an old BASIC program (game or whatever) and EXEC your sound routine into it. For this to work properly, however, you must have specified the starting line number during the save of your music file such that the line numbers of the music routine do not conflict with those of the program to which the routine is being added.

Insert. If you have a large amount of repetitive data to type in, one trick is to enter the last note of that data, then Insert the rest. This saves you from repeatedly typing A from COMMAND? mode. (This is useful only if you know beforehand exactly what data you want to enter.)

Keys. There are a number of key codes that you can use with Apple Music Writer. If at any time the screen is getting too cluttered, an ESC-SHIFT-P should do the trick. You can stop a Catalog or a Play/list at any moment with a CTRL-S, and restart it with the touch of any key. Finally, in this program, CTRL-C RETURN can be a useful but sometimes dangerous command. I would recommend using CTRL-C only if you are caught in a never-ending loop or as a last resort. If for any reason you find yourself out of Apple Music Writer, you can usually reenter the program, without losing any data, by typing GOTO 200.

Experiment. No matter how long or well written a manual, nothing can take the place of hands-on experience with a program. Before you try any big projects, be sure you know what's going to happen at all times no matter what you enter during Apple Music Writer. Overall the program is very forgiving. One last thing – the best songs on the Apple seem to be songs with few or no rests. Try using longer notes instead of rests.

If you'd rather not type in the program, send $3, a stamped, self-addressed disk mailer, and an initialized blank disk (Apple DOS 3.3 compatible) to:

Blaine Mathieu
Box 2572
Peace River, Alberta
Canada, T0H 2X0

Apple Music Writer

```
10  REM    APPLE MUSIC WRITER
20  REM  INITIALIZATION    ,
30  TEXT : HOME : VTAB 1: PRINT "A=ADDN
     OTE E=EDIT    P=PLAY    S=SAVE
     L=LOAD    N=NORPLAY D=DELETE  I=IN
     SERT  R=RESTART C=CATALOG Q=QUIT .
     =DOS H=HARD": PRINT "-------------
    ---------------------------": POKE
    34,5
40  VTAB 6: HTAB 34: PRINT "G =64": PRINT
     TAB( 34)"F#=68": PRINT  TAB( 34)"
    F =72": PRINT  TAB( 34)"E =76": PRINT
     TAB( 34)"D#=81": PRINT  TAB( 34)"
    D =86": PRINT  TAB( 34)"C#=91"
50  PRINT  TAB( 34)"C =96": PRINT  TAB(
    34)"B =102": PRINT  TAB( 34)"A#=10
    8": PRINT  TAB( 34)"A =115": PRINT
     TAB( 34)"G#=121": PRINT  TAB( 34)
    "G =128": PRINT  TAB( 34)"/2 FOR":
     PRINT  TAB( 34)"HIGHER": PRINT  TAB(
    34)"*2 FOR": PRINT  TAB( 34)"LOWER
    ": POKE 33,32
60  FOR LOC = 770 TO 790: READ BYTE: POKE
    LOC,BYTE: NEXT
70  DATA  173,48,192,136,208,5,206,1,3,
    240,9,202,208,245,174,0,3,76,2,3,96
80  HOME : INVERSE : VTAB 10: HTAB 9: PRINT
    "APPLE MUSIC WRITER"
90  FOR R = 1 TO 26: READ P,D: POKE 768
    ,P: POKE 769,D: CALL 770: NEXT R
100  DATA 172,75,162,75,152,75,144,75,1
     08,100,1,30,144,75,108,100,1,30,14
     4,75,108,255,1,10,108,75,96,75,91,
     75,86,75,108,75,96,75,86,100
110  DATA 1,10,115,75,96,100,1,10,108,1
     50,144,150,216,200,
120  HOME :L = 500: DIM N(L),D(L),N$(L)
     ,D$(L),NN(L),ND(L)
130  REM  MAIN ROUTINES START
140  VTAB 5: GOTO 190
150 I = I + 1
160  PRINT : INVERSE : PRINT "NOTE#"I:
     NORMAL
165  INPUT "NOTE,DURATION ";N$(I),D$(I)
     : IF N$ =  CHR$ (8) OR D$ =  CHR$
     (8) THEN N$(I) = N$(I - 1):D$(I) =
     D$(I - 1)
170 N(I) =  VAL (N$(I)):D(I) =  VAL (D$
     (I)): IF N(I) > 255 OR N(I) < 0 OR
     D(I) > 255 OR D(I) < 0 THEN 160
180  POKE 768,N(I): POKE 769,D(I): CALL
     770
190  ONERR  GOTO 370
200  PRINT : INPUT "COMMAND? ";A$
210  IF A$ = "A" AND I = L THEN  PRINT
     "YOU ARE AT YOUR LIMIT!!!": GOTO 2
     00
220  IF A$ = "A" THEN 150
230  IF I <  = 0 AND (A$ = "E" OR A$ =
     "P" OR A$ = "H" OR A$ = "N" OR A$ =
     "I" OR A$ = "S") THEN  PRINT "SORR
     Y, NO NOTES":I = 0: GOTO 190
240  IF A$ = "Q" THEN 450
250  IF A$ = "E" THEN 470
260  IF A$ = "P" THEN 390
270  IF A$ = "S" THEN 530
280  IF A$ = "D" THEN 1410
290  IF A$ = "L" THEN 990
300  IF A$ = "R" THEN I = 0
310  IF A$ = "C" THEN  PRINT  CHR$ (4)"
     CATALOG"
320  IF  LEFT$ (A$,1) = "." THEN 1120
330  IF A$ = "H" THEN 1160
340  IF A$ = "N" THEN 1250
350  IF A$ = "I" THEN 1310
360  GOTO 190
370  PRINT "ERROR#" PEEK (222): GOTO 190
380  REM  PLAY ROUTINE
```

```
390  PRINT : INPUT "STARTING NOTE (D=1)
     : ";SN$:SN =  VAL (SN$): IF SN$ =
     "" THEN SN = 1
400  PRINT : INPUT "ENDING NOTE (D=LAST
     ): ";EN$:EN =  VAL (EN$): IF EN$ =
     "" THEN EN = I
410  IF SN < 1 OR SN > I OR EN < 1 OR E
     N > I THEN 390
420  PRINT : INVERSE : PRINT "START OF
     SONG": PRINT : NORMAL : FOR X = SN
      TO EN: POKE 768,N(X): POKE 769,D(
     X): PRINT "NOTE#";X;: HTAB 10: PRINT
     "NOTE=";N(X);: HTAB 19: PRINT "DUR
     ATION=";D(X): CALL 770: NEXT X
430  INVERSE : PRINT : PRINT "END OF SO
     NG": NORMAL
440  GOTO 190
450  TEXT : HOME : PRINT "GOODBYE": END

460  REM  EDIT ROUTINE
470  INPUT "EDIT NOTE# ";NN: IF NN > I OR
     NN < 1 THEN 470
480  PRINT : INVERSE : PRINT "NOTE#"NN;
     : NORMAL : PRINT " OLD: NOTE="N(NN
     );" DUR="D(NN)"
490  INPUT "NOTE,DURATION: ";N$(NN),D$(
     NN):N(NN) =  VAL (N$(NN)):D(NN) =
      VAL (D$(NN)): IF N(NN) > 255 OR N
     (NN) < 0 OR D(NN) > 255 OR D(NN) <
     0 THEN 480
500  POKE 768,N(NN): POKE 769,D(NN): CALL
     770
510  GOTO 190
520  REM  SAVE ROUTINE
530  ONERR  GOTO 860
540  HOME : INPUT "FILENAME? ";FI$: IF
     FI$ = "" THEN 540
550  PRINT : INPUT "STARTING NOTE NUMBE
     R: ";SN: IF SN < 1 OR SN > I THEN
     550
560  PRINT : INPUT "ENDING NOTE NUMBER:
      ";EN: IF EN > I OR EN < 1 THEN 56
     0
570  PRINT : INPUT "FOR FUTURE ADDITION
     ? ";A$: IF A$ <  > "N" AND A$ <  >
     "Y" THEN 570
580  IF A$ = "Y" THEN  POKE 216,0:F2 =
     1: GOTO 640
590 F2 = 0
600  PRINT : INPUT "STARTING LINENUMBER
     : ";SL: IF SL > 63900 OR SL < 0 THEN
     600
610  PRINT : INPUT "FULL LOADER PROGRAM
     ? ";A$:A$ =  LEFT$ (A$,1): IF A$ <
      > "Y" AND A$ <  > "N" THEN 610
620  IF A$ = "Y" THEN FL = 1
630  IF A$ = "N" THEN FL = 0
640  PRINT : INPUT "IS EVERYTHING OK? "
     ;A$: IF  LEFT$ (A$,1) = "Y" AND F2
      = 1 THEN 880
650  IF  LEFT$ (A$,1) = "Y" AND F2 <  >
     1 THEN 670
660  GOTO 190
670 D$ =  CHR$ (4): PRINT D$"OPEN"FI$
680  PRINT D$"DELETE"FI$
690  PRINT D$"OPEN"FI$
700  PRINT D$"WRITE"FI$
710  IF FL <  > 1 THEN  GOTO 740
720  PRINT SL;"FORLOC=770TO790:READBYTE
     :POKELOC,BYTE:NEXT":SL = SL + 2
730  PRINT SL;"DATA173,48,192,136,208,5
     ,206,1,3,240,9,202,208,245,174,0,3
     ,76,2,3,96":SL = SL + 2
740  PRINT SL;"FORR=1TO";EN - SN + 1;":
     READP,D:POKE768,P:POKE769,D:CALL77
     0:NEXTR":SL = SL + 2
750  FOR Z = SN TO EN
760 N = N + 1: IF N = 20 THEN N = 1
770  IF N <  > 1 THEN 810
780  PRINT
790  PRINT SL;"DATA";
800 SL = SL + 2
810  PRINT N(Z);",";D(Z);: IF N <  > 19
      THEN  PRINT ",";
820  NEXT Z
830  PRINT
840  PRINT D$"CLOSE"
850  GOTO 190
860  PRINT : PRINT  CHR$ (7);"ERROR#"; PEEK
     (222): PRINT D$"CLOSE": GOTO 190
870  REM  2ND SAVE ROUTINE
880  ONERR  GOTO 980
890 D$ =  CHR$ (4): PRINT D$"OPEN"FI$
900  PRINT D$"DELETE"FI$
910  PRINT D$"OPEN"FI$
920  PRINT D$"WRITE"FI$
930  FOR S = SN TO EN
940  PRINT N(S): PRINT D(S)
950  NEXT S
960  PRINT D$"CLOSE"
970  GOTO 190
980  REM  LOAD ROUTINE
990  ONERR  GOTO 1090
1000  INPUT "YOU WILL LOSE YOUR DATA, O
     K? ";OK$:OK$ =  LEFT$ (OK$,1): IF
     OK$ <  > "Y" AND OK$ <  > "N" THEN
     1000
1010  IF OK$ = "N" THEN  POKE 216,0: GOTO
     190
1020  PRINT : INPUT "FILENAME: ";FI$: IF
     FI$ = "" THEN 1020
1030 D$ =  CHR$ (4): PRINT D$"VERIFY"FI
     $: PRINT D$"OPEN"FI$
1040  PRINT D$"READ"FI$
1050  FOR Z = 1 TO L
1060  INPUT N(Z): INPUT D(Z)
1070  IF N(Z) <  = 255 AND D(Z) <  = 25
     5 THEN  NEXT Z: POKE 216,0: PRINT
     D$"CLOSE":I = Z - 1: GOTO 190
1080  PRINT : PRINT "INCOMPATIBLE FILE!
     !!": PRINT D$"CLOSE": POKE 216,0: GOTO
     190
1090  PRINT D$"CLOSE": IF  PEEK (222) =
     5 THEN  POKE 216,0:I = Z - 1: GOTO
     190
1100  PRINT : PRINT "ERROR#"; PEEK (222
     ): PRINT D$"CLOSE": GOTO 190
1110  REM  HANDLE DOS COMMANDS
1120  ONERR  GOTO 1140
1130 DC$ =  RIGHT$ (A$, LEN (A$) - 1): PRINT
      CHR$ (4);DC$: POKE 216,0: GOTO 19
     0
1140  PRINT "ERROR#" PEEK (222): PRINT
      CHR$ (4)"CLOSE": POKE 216,0: GOTO
     190
1150  REM  PRINTER ROUTINE
1160  PRINT : INPUT "PRINTER READY? ";A
     $: IF A$ <  > "Y" AND A$ <  > "N" THEN
     1160
1170  IF A$ = "N" THEN 200
1180  PRINT : INPUT "STARTING NOTE TO B
     E PRINTED --  DEFAULT=1: ";ST$: IF
```

```
     ST$ = "" THEN ST$ = "1"
1190  PRINT : INPUT "ENDING NOTE TO BE
     PRINTED --    DEFAULT=ALL: ";EN$: IF
     EN$ = "" THEN EN$ =  STR$ (I)
1200 ST =  VAL (ST$):EN =  VAL (EN$): IF
     ST < 1 OR ST > I OR EN < 1 OR EN >
     I OR EN < ST THEN 1180
1210  PRINT : INPUT "NAME OF SONG: ";FI
     $: IF FI$ = "" THEN 1210
1220  PR# 1: PRINT : PRINT FI$: PRINT :
      FOR X = ST TO EN: PRINT "NOTE#";X
     ;: HTAB 10: PRINT "NOTE=";N(X);: HTAB
     19: PRINT "DURATION=";D(X): NEXT X

1230  PRINT : PRINT "END OF SONG": PR#
     0: GOTO 190
1240  REM  NORMAL PLAY ROUTINE
1250  PRINT : INPUT "STARTING NOTE (D=1
     ): ";SN$:SN =  VAL (SN$): IF SN$ =
     "" THEN SN = 1
1260  PRINT : INPUT "ENDING NOTE (D=LAS
     T): ";EN$:EN =  VAL (EN$): IF EN$ =
     "" THEN EN = I
1270  IF SN < 1 OR SN > I OR EN < 1 OR
     EN > I THEN 1250
1280  FOR Z = SN TO EN: POKE 768,N(Z): POKE
     769,D(Z): CALL 770: NEXT Z
1290  GOTO 190
1300  REM  INSERT ROUTINE
1310  POKE 216,0: PRINT : INPUT "INSERT
      BEFORE WHAT NOTE? ";IB: IF IB < 1
      OR IB > I THEN 1310
1320  PRINT : INPUT "HOW MANY NOTES TO
      INSERT? ";HM: IF HM > L - I OR HM <
      1 THEN 1320
1330  FOR Z = IB TO IB + HM - 1
1340  PRINT : INVERSE : PRINT "NOTE#"Z:
      NORMAL : INPUT "NOTE,DURATION: ";
     NN(Z),ND(Z): IF NN(Z) < 0 OR NN(Z)
      > 255 OR ND(Z) < 0 OR ND(Z) > 255
      THEN 1340
1350  POKE 768,NN(Z): POKE 769,ND(Z): CALL
     770
1360  NEXT Z
1370  FOR Z = I TO IB STEP  - 1:N(Z + H
     M) = N(Z):D(Z + HM) = D(Z): NEXT Z

1380  FOR Z = IB TO IB + HM - 1:N(Z) =
     NN(Z):D(Z) = ND(Z): NEXT Z
1390 I = I + HM
1400  GOTO 190
1410  REM  DELETE ROUTINE
1420  PRINT : INPUT "DELETE FROM NOTE (
     D=LAST): ";DF$: IF DF$ = "" THEN I
      = I - 1: IF I =  - 1 THEN I = 0: GOTO
     190
1430  IF DF$ = "" THEN 190
1440  PRINT : INPUT "TO NOTE: ";DT$:DF =
      VAL (DF$):DT =  VAL (DT$): IF DT <
     1 OR DT > I OR DF < 1 OR DF > I OR
     DF > DT THEN 1420
1450  FOR Z = DT + 1 TO I:N(Z - (DT - D
     F + 1)) = N(Z):D(Z - (DT - DF + 1)
     ) = D(Z): NEXT Z
1460 I = I - (DT - DF + 1): GOTO 190
```

©

INSIGHT: Atari

Bill Wilkinson

Bill concludes last month's column with a program demonstrating the capabilities of the new graphics modes.

If you were a little disconcerted by our discussion last month, here is a little BASIC program which demonstrates the capabilities of the new modes in a crude, but visible, fashion. As usual, I will explain the program line by line.

120. Selects a normal GRAPHICS 2. This is our starting point.

130. Prints a reference line on the screen. This is simply so you can tell where the columns of characters are later, when they get MAPped.

150-180. Print what are now normal characters. Note that the underline denotes inverse video characters (via the Atari key). Did you notice that each set of four characters here will produce the MAP patterns 00, 01, 10, and 11 (in that order) on each line of the displayed area? Remember that the other six bits, then, will select a character from character memory.

190, 290, 320, and **340.** Just messages, to tell you what we are doing.

200-220. We are moving the normal Atari 800 character set from its normal location ($E000) to RAM at address $6000. Note: This requires a 32K machine.

230-250. Here we read the DATA statements from lines 380 to 420 and change the character set for the characters A, B, C, and D.

260-280. A quick and dirty way to arbitrarily select some colors for the various color registers.

300 and **330.** Just some delay loops, so you can actually see it happening.

310. Changes the CHBASE (CHaracter BASE pointer) to point to location $6000, where the new character set pattern is.

350. The magic instruction. Look at your screen. How many different colors do you see?

Do you see the relation between the display and the table? Did you notice that the first character in each line "disappeared"? That's because these characters are using MAP 00, the "all background" map.

I think the only thing left is to explain the bit patterns of the modified characters which are read in by lines 230 to 250.

Character A is changed to a solid block of all "11" bits (thus the pattern is eight $FF bytes).

Character B is changed to a solid block of all "10" bits (eight bytes of $AA). Character C is a solid block of "01" bits (eight bytes of $55).

Finally, character D has a purposely varied pattern. The bit patterns in the byte are as follows:

228	$E4	11	10	01	00
57	$39	00	11	10	01
78	$4E	01	00	11	10
147	$93	10	01	00	11

and then the same bytes in reverse order.

The result of the shifted bit pattern shown is, quite naturally, the "arrows" which you see in the program's display.

Finally, we are finished explaining these new modes. What good are they? Just imagine what Chris Crawford could do with a map which displays seven different colors, instead of only four. But surely there are other uses. How about inventing some and sharing them with us?

```
100 REM DEMO OF THE "NEW" GRAPHICS MODE!
110 REM
120 GRAPHICS 2
130 PRINT #6;"wxyz"
140 PRINT #6;"    "
150 PRINT #6;"AaAa"
160 PRINT #6;"BbBb"
170 PRINT #6;"CcCc"
180 PRINT #6;"DdDd"
190 PRINT "THIS IS IN NORMAL GRAPHICS 2"
200 FOR A=24576 TO 25599
210   POKE A,PEEK(A+32768)
220   NEXT A
230 FOR A=24840 TO 28671
240   READ D:IF D<0 THEN 260
250   POKE A,D:NEXT A
260 FOR A=0 TO 8
270   POKE 704+A,18*A+18
280   NEXT A
290 PRINT "THIS IS WITH COLORS CHANGED"
300 FOR I=1 TO 1000:NEXT I
310 POKE 756,96
320 PRINT "THIS IS THE MODIFIED CHARACTER S
    ET"
330 FOR I=1 TO 1000:NEXT I
340 PRINT "FINALLY, THE NEW AND SPECIAL MOD
    E!"
350 POKE 623,128
360 REM == JUST A LOOP TO KEEP DISPLAYING ==
370 GOTO 360
380 DATA 255,255,255,255,255,255,255,255
390 DATA 170,170,170,170,170,170,170,170
400 DATA 85,85,85,85,85,85,85,85
410 DATA 228,57,78,147,147,78,57,228
420 DATA -1
```

©

Jim Butterfield, Associate Editor

Debugging

When you write a program, you're going to make mistakes. It's hard on the ego, but there it is: none of us are perfect. And we cannot consider our program complete unless we have worked out the errors. Not just the obvious ones – those will be easy to spot since they will often prevent the program from running. We need to go further: to methodically search out less obvious bugs and get them out of there.

When a good programmer completes a program and turns to the testing and debugging phase, he or she needs to have something of a split personality. The natural way for us to respond is to hope that there are no bugs, and sometimes that means that we don't try very hard to find them. The smart programmer switches from a Doctor Jekyll coder to a Mr. Hyde tester, mercilessly trying to find weak spots in the program.

There's no fixed procedure for testing. The programmer will try running "ordinary" data through his program, of course, but should also try probing for weak spots – badly formatted lines, operator errors. Test files should be carefully prepared in advance, and output files closely examined after the run.

Debugging Aids

Today's microcomputers seem somewhat weak on formal debugging aids compared to the "big" computers, but this is partly an illusion. Mainframes can't tolerate programmers playing with the toggle switches – time is money, and the machines have many tasks to do. Because of this, elaborate debugging aids have been developed to allow the programmer to trace down troubles away from the computer.

Microcomputers, on the other hand, are often readily available to the programmer; debugging can take place on-line, and the formal aids are needed less.

We can do many things "on-line" on our micros that must be done "off-line" on big machines. For example, a major debugging aid is the "memory dump" – formerly called a "core dump" when memory was made of small magnetic cores. The big-system programmer would receive dozens of pages of memory printout – often in octal or hexadecimal – and might spend hours studying it. The microcomputer programmer, on the other hand, can simply inspect the contents of memory at the computer itself.

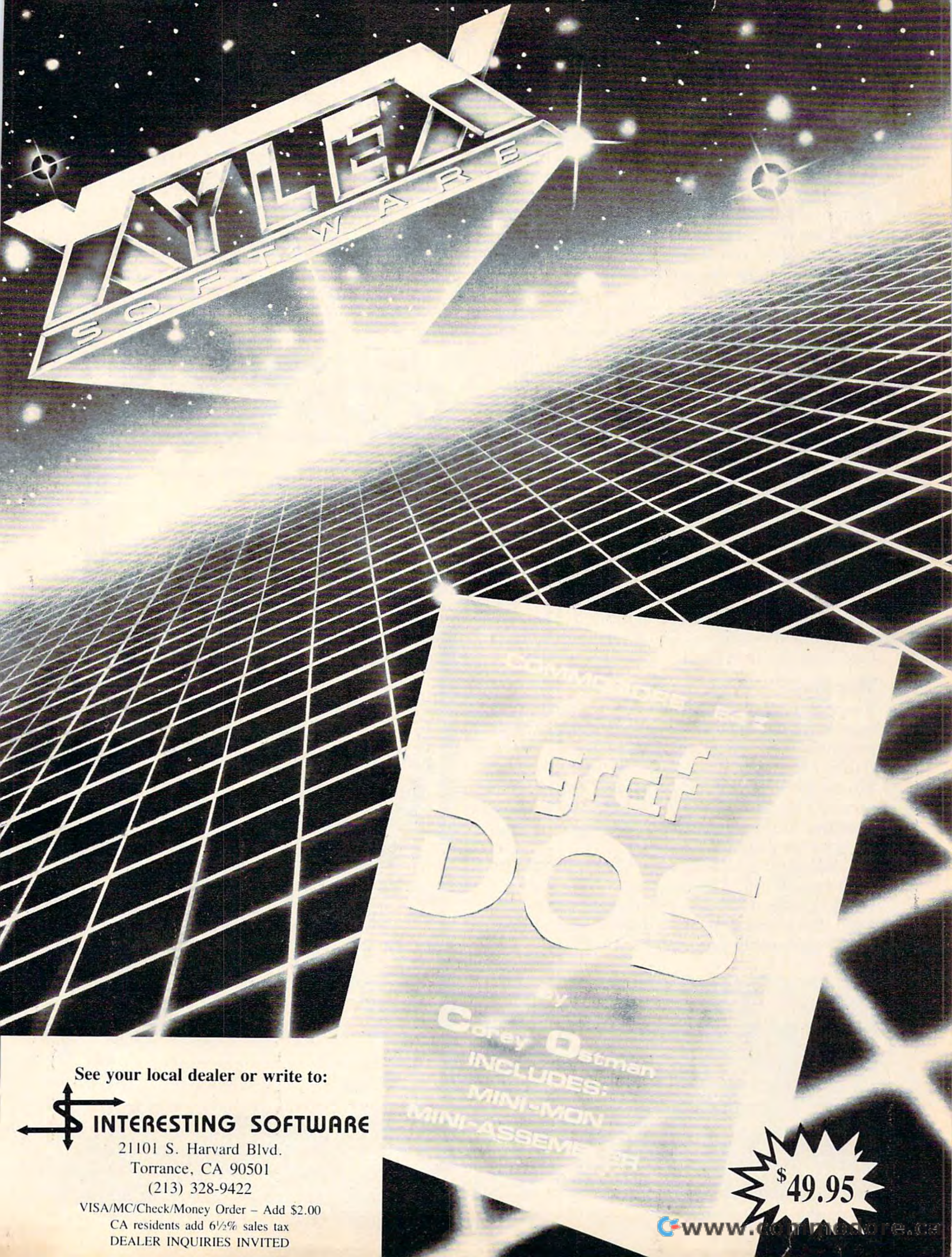
There's a style difference, however. A programmer who sits in a cubicle with printout and a pencil is likely to be less hasty in his analysis. On the other hand, a programmer who sits before a memory display on his machine is likely to shout, "I know what it is!" and immediately type in changes and run again. Sadly, most such changes don't work out. When we're in a rush, we tend to try to fix the symptoms rather than the problem.

Let's discuss some of the formal methods available on big computers that can be used on our micros.

The Memory Dump

A program of any significant size leaves a trail behind it in memory. It accepts input, and puts that input somewhere. It uses work areas, builds tables, and computes statistics. It prepares output. All of these leave traces in memory. In fact, experienced programmers often make sure that these values will be there to aid testing. A work area would be cleared immediately before use, not immediately after, so that its contents will be visible until the next use.

Careful examination of memory can be one of the most powerful tools in the debugging repertoire. Everything is there: your program, your data, your work areas. With enough close study, you'll almost certainly find the problem.



See your local dealer or write to:

INTERESTING SOFTWARE
21101 S. Harvard Blvd.
Torrance, CA 90501
(213) 328-9422

VISA/MC/Check/Money Order – Add $2.00
CA residents add 6½% sales tax
DEALER INQUIRIES INVITED

Traces

A *trace* calls on a computer to report every time it passes a given program point. Full traces cause the computer to report every instruction it executes, which creates a great deal of output. *Branch traces* report only the changes in logic flow – branches that are taken, jumps, subroutines that are called, etc. You can tell roughly where a program was working when it got into trouble, since you'll see the last place that it went to a new location. Similarly, you can see the logic flow so that a program can be checked to verify that it did indeed take a given jump.

We have a comparable facility to Trace on our microcomputer; it's sometimes associated with single step. Instructions are displayed on the screen as they are executed. Specialized tools like "branch traces" are less needed; we can watch the program run and see the branches.

Snapshots

The *snapshot* allows you to see a copy of memory at a given time: say, when a particular instruction is executed. It allows you to watch a work area and see how it is built over time.

We can achieve similar results by putting *breakpoints* into our program. These are often just BRK, "break," instructions. Each time we reach a breakpoint, the program will stop, and we may examine memory locations as desired. Then we may allow the program to continue where it left off – until the next break.

The Wolf

You may hear of the "wolf fence" method of debugging. That's just another way of asking your program to tell you when it passes a given point (crosses the "wolf fence"). In this way, you should be able to tell which section of your program contains the bug (the "wolf"). It's just common sense: dividing your program into ever-smaller pieces and checking out each piece.

Debugging is partly an art: some people are very good at it. It's also a science: you must be methodical in making sure your programs work right. But in any case, it's a duty: find your own bugs before other users fall prey to them. ©

COMPUTE! is looking for good articles, tutorials, and games for the Timex/Sinclair, Commodore 64, and Color Computer.

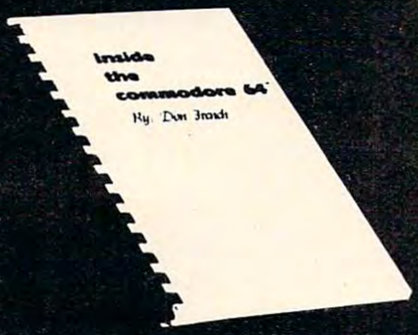
French Silk

DEVELOP-64

Powerful enough for the professional
friendly enough for everyone

An integrated Assembler Editor Loader Decoder Debugger
which comes complete with

Inside the Commodore 64



This step-by-step guide to machine language and assembly language programming will teach you what you don't already know and assist you to use what you do. Combined with the integrated set of software tools of Develop-64, it makes the ideal development system.

A total reference work on the Commodore 64, Inside the Commodore 64 gives you inside information on the inside of the machine: its graphics, its music synthesizer, its built-in software and the techniques for taking advantage of the many powerful features of this computer. A complete memory map is provided with information on how to call the internal programs from your own. Written as a programmer's guide with a machine language programmer's perspective, it will become your primary reference guide. If you are still learning you will find it your best teacher.

Priced at $19.95, Inside the Commodore 64 comes free with Develop-64 ($49.95).

Ask for them at your favorite software outlet.

Software for Commodore Personal Computers.

PO Box 207, Cannon Falls, MN 55009 (507) 263-4821

Commodore 64 and VIC-20

SuperTerm

$99.95

Telecommunications with a difference!

Unexcelled communications power and compatibility, especially for professionals and serious computer users. Look us over; **SuperTerm** isn't just "another" terminal program. Like our famous Terminal-40, **it's the one others will be judged by.**

- **UP/DOWNLOAD FORMATS** –CBM, Xon-Xoff, ACK-NAK, etc.
- **DISPLAY MODES** — 40 column; 80/132 with side-scrolling
- **EMULATION** — 42 popular terminal protocols
- **FUNCTION KEYS** — 8 standard, 52 user-defined
- **BUFFERS** — Receive, Transmit, Program, and Screen
- **EDITING** — Full-screen editing of Receive buffer
- **FILE CONVERSION** — ASCII to PGM, PGM to ASCII
- **PRINTING** — Continuous printing with Smart ASCII and parallel printer; buffer printing with other interfaces or VIC printer
- **DISK SUPPORT** — Directory, Copy, Rename, Scratch, etc.
- **FLEXIBILITY** — Select baud, duplex, parity, stopbits, etc.

Program options are selected by menus and function keys. For maximum convenience, an EXEC file sets options on start-up. SuperTerm may be backed-up for safety. Software on disk or cassette, with special cartridge module.

Write for the full story on SuperTerm; or, if you already want that difference, order today!

Requires: Commodore 64 or VIC-20, disk drive or Datasette, and compatible modem. VIC version requires 16K memory expansion. Please specify VIC or 64 when ordering.

Just need UP/DOWNLOAD?

If you don't yet need SuperTerm's power, perhaps **Terminal-40 Plus** (VIC) or **'64 Terminal Plus** is right for you. We took our top-rated, smooth-scrolling terminal programs, added up/download, disk commands, and even more convenience. Then we put them on disk for fast loading, just like you wanted. Need we say more?

Only $49.95 (VIC version requires 8K mem exp)

P.S. Trade in your original Terminal-40 or '64 Terminal and deduct $10.00.

VIC 20 and Commodore 64 are trademarks of Commodore Electronics, Ltd.

(816) 333-7200

Send for a free brochure.

MAIL ORDER: Add $1.50 shipping and handling ($3.50 for C.O.D.); VISA/Mastercard accepted (card# and exp. date). MO residents add 5.625% sales tax. Foreign orders payable U.S.$, U.S. Bank ONLY; add $5 shp/hndlg.

MIDWEST MICRO inc.

311 WEST 72nd ST. • KANSAS CITY • MO • 64114

Phone Directory And Dialer For The TI

Ken McCann

This useful program will work as a phone number file as well as an automatic dialer. It will run on the TI with or without Extended BASIC.

Your computer, cassette recorder, and TV are all you need to run this program. Although it is written in standard TI BASIC, it will run faster if you have Extended BASIC.

DATA statements are included in the program, so only one load is required. Also, with the data files at the beginning of the program, data entry is simple and straightforward. The line numbers and the file numbers are the same, to make updating and deleting less complicated.

DATA Line Format

Enter the name and phone number information for your personal directory in the form shown below (also see line 1 in program for same example):

Line No.	File No.	Name	Phone Number
1 DATA	1,	MCCANN.K,	1,2,3,4,5,6,7
variables (in program)	A	B$	C D E F G H I

The last DATA entry must be followed by an END line. For example:

140 DATA 140,END

Auto Dialing

In most cases, the accuracy of the frequencies generated with the CALL SOUND statement is not close enough to use as "touch tones" to dial the phone. Therefore, I executed CALL SOUND statements with a frequency counter hooked up to the TV audio output, and added or subtracted until I got the proper frequency.

To use the Auto Dial feature, hold the phone up to the speaker of your TV and press C. Note that you must have a Touch-Tone type phone to use the Auto Dial feature. Two tones were used, and the frequencies for each digit of the phone dial are as follows:

1 697,1209	6 770,1447
2 697,1336	7 852,1209
3 697,1447	8 852,1336
4 770,1209	9 852,1447
5 770,1336	0 941,1336

Program Operation

Search Name and Dial: Type RUN, then press N to enter the Search mode. You are prompted to enter the Name *exactly* as it is in the files and then to press ENTER. The computer will display the DATA item called for. Press C, and the computer will PRINT the number and sound the dial tones for the number.

List: Type RUN, then press L to enter the List mode. You will see DATA as it is in the files. Press C to look at the entire list.

Letter Index List: Type RUN, then press I to enter the Index Mode. Then simply enter the first letter of the last name and press ENTER. The computer will display all the entries beginning with that letter.

Program Explanation

Lines 1-299 can be used for DATA statements. Remember to put the END statement last, after all files are listed.

Lines 300-480 set up the menu.

Lines 510-740 list all DATA items in the order they appear.

Lines 810-990 search DATA for a particular name.

Lines 1000-1820 dial the number and print the file number.

Lines 1830-2030 search for and print all names beginning with a given letter.

Phone Directory

```
1 DATA 1,MCCANN.K,2,1,2,4,4,4,4
2 DATA 2,CLAUSS.S,5,5,5,1,2,1,2
3 DATA 3,NIXON.R,3,3,3,4,5,4,5
4 DATA 4,END
300 CALL CLEAR
```

```
310 PRINT "PHONE DIRECTORY"
320 PRINT "{3 SPACES}& DIALER":::
330 PRINT "SELECT MODE DESIRED"
340 PRINT
350 PRINT
360 PRINT
370 PRINT "LIST ALL ENTRIES
    {6 SPACES}(L)"
380 PRINT
390 PRINT "SEARCH NAME & DIAL
    {4 SPACES}(N)"
400 PRINT
410 PRINT "LETTER INDEX LIST
    {5 SPACES}(I)"
420 CALL KEY(0,KEY,STATUS)
430 CALL SOUND(50,4000,8)
440 IF STATUS=0 THEN 420
450 IF KEY=76 THEN 500
460 IF KEY=78 THEN 780
470 IF KEY=73 THEN 1820
480 GOTO 1260
490 REM
500 REM
510 CALL CLEAR
520 CALL SCREEN(16)
530 PRINT "PHONE DIRECTORY LIST"
540 PRINT
550 FOR Z=1 TO 200
560 PRINT
570 READ A
580 READ B$
590 IF B$="END" THEN 1770
600 READ C,D,E,F,G,H,I
610 PRINT "FILE NUMBER  >";A
620 PRINT
630 PRINT "NAME  >";B$
640 PRINT
650 PRINT "NUMBER>";C;D;E;"-";F;G;H
    ;I
660 PRINT
670 PRINT "PRESS <C> TO PROCEED WIT
    H"
680 PRINT "LIST"
690 REM
700 CALL SOUND(100,1000,2)
710 CALL SOUND(75,675,2)
720 CALL KEY(0,KEY,STATUS)
730 IF STATUS=0 THEN 720
740 IF KEY=67 THEN 760
750 PRINT
760 NEXT Z
770 REM
780 REM{3 SPACES}
790 CALL CLEAR
800 CALL SCREEN(12)
810 REM
820 PRINT "NAME SEARCH"
830 PRINT
840 PRINT "ENTER NAME TO SEARCH FOR
    "
850 INPUT I$
860 PRINT
870 PRINT
880 PRINT
890 FOR S=1 TO 2000
900 REM
910 IF B$="END" THEN 1780
920 READ B$
930 IF B$<>I$ THEN 1530
940 PRINT "INDEX LETTER >";SEG$(B$,
    1,1)
950 PRINT
960 PRINT "NAME >";B$
970 PRINT
980 PRINT "READY TO DIAL"
990 PRINT
1000 PRINT "PRESS >C< TO DIAL NUMBE
     R"
1010 PRINT
1020 CALL SOUND(100,1000,2)
1030 CALL SOUND(75,675,2)
1040 CALL KEY(0,KEY,STATUS)
1050 CALL SOUND(50,2000,6)
1060 IF STATUS=0 THEN 1040
1070 IF KEY=67 THEN 1090
1080 REM
1090 REM
1100 READ C
1110 PRINT C;
1120 N=C
1130 GOSUB 1430
1140 READ D
1150 PRINT D;
1160 N=D
1170 GOSUB 1430
1180 READ E
1190 PRINT E;
1200 N=E
1210 GOSUB 1430
1220 READ F
1230 PRINT F;
1240 N=F
1250 GOSUB 1430
1260 READ G
1270 PRINT G;
1280 N=G
1290 GOSUB 1430
1300 READ H
1310 PRINT H;
1320 N=H
1330 GOSUB 1430
1340 READ I
1350 PRINT I;
1360 N=I
1370 GOSUB 1430
1380 READ A
1390 A=A-1
1400 PRINT "{5 SPACES}"::
1410 PRINT "FILE NUMBER >";A
1420 GOTO 1740
1430 IF N=1 THEN 1540
1440 IF N=2 THEN 1560
1450 IF N=3 THEN 1580
1460 IF N=4 THEN 1600
1470 IF N=5 THEN 1620
1480 IF N=6 THEN 1640
1490 IF N=7 THEN 1660
1500 IF N=8 THEN 1680
1510 IF N=9 THEN 1700
1520 IF N=0 THEN 1720
1530 NEXT S
1540 CALL SOUND(100,1209,0,697,0)
1550 RETURN
1560 CALL SOUND(100,1336,0,697,0)
1570 RETURN
1580 CALL SOUND(100,1447,0,697,0)
1590 RETURN
1600 CALL SOUND(100,1209,0,770,0)
```

```
1610 RETURN
1620 CALL SOUND(100,1336,0,770,0)
1630 RETURN
1640 CALL SOUND(100,1447,0,770,0)
1650 RETURN
1660 CALL SOUND(100,1209,0,852,0)
1670 RETURN
1680 CALL SOUND(100,1336,0,852,0)
1690 RETURN
1700 CALL SOUND(100,1447,0,852,0)
1710 RETURN
1720 CALL SOUND(100,1336,0,941,0)
1730 RETURN
1740 PRINT
1750 PRINT "type >RUN< to start pro
     gram again"
1760 END
1770 PRINT
1780 PRINT "end of list"
1790 PRINT
1800 PRINT "type >RUN< to start pro
     gram over"
1810 END
1820 CALL CLEAR
1830 CALL SCREEN(15)
1840 PRINT "LETTER INDEX LIST"
1850 PRINT
1860 PRINT "ENTER FIRST LETTER IN L
     AST NAME TO SEARCH FOR"
1870 PRINT
1880 INPUT H$
1890 PRINT
1900 FOR Z=1 TO 2000
1910 READ B$
1920 IF B$="END" THEN 1990
1930 IF SEG$(B$,1,1)<>H$ THEN 1980
1940 READ C,D,E,F,G,H,I
1950 PRINT B$
1960 PRINT C;D;E;"-";F;G;H;I
1970 PRINT
1980 NEXT Z
1990 PRINT
2000 PRINT "END OF LETTER INDEX FIL
     E"
2010 PRINT
2020 PRINT "TYPE >RUN< TO START AGA
     IN"
2030 END
```

©

Modem Save And Download For The VIC-20

Dennis Colombo

When used together, this series of programs will let you save downloaded VICmodem files and listings. Also included is a program which will convert a downloaded listing to a tokenized BASIC program that you can run. Requires expansion memory.

After I bought my VICmodem, it soon became evident that I needed a way to store all the information which was coming over the line and relentlessly scrolling off the top of the screen. Without a disk drive, I needed a way to store the data on tape so I could later read the information off the screen or edit and make hard copies of selected data such as specific files, airline schedules, or encyclopedia information. Connect time could be appreciably reduced if I didn't have to stop and absorb the information as I received it. With these considerations in mind, I wrote Programs 1, 2, and 3.

Also, there is another type of data which may be retrieved: program listings. While the capability to store and read these listings is useful, they probably won't be in a form your VIC will understand, and they will not run. So I wrote Program 4, which will convert a BASIC listing back into a "tokenized" BASIC program which *will* run.

Machine Language Is More Effective

Program 2 will run the terminal and allow the information to be saved to tape and printed on the VIC printer. It uses 2K of memory; the remaining RAM is utilized as a buffer to hold the information until it is sent to tape or printer. The program was written with a 16K expander, but will work with an 8K or 24K expander. Although it could easily be modified to run without memory expansion, the small amount of memory left for the text buffer would fill up too quickly to be practical.

Program 1 loads VIC to ASCII and ASCII to VIC lookup tables, a machine language subroutine to send and receive data, and a machine language interrupt service routine into memory. I had originally attempted to write the terminal program entirely in BASIC. I found, however, that for receiving large amounts of text, BASIC was too slow. This resulted in buffer overflow and frequent loss of data. The machine language subroutine in Program 1 which handles data reception and transmission is far more effective.

Program 3 will allow you to search through the text in RAM and select portions to be displayed on the screen and sent to the printer.

Program 4 will download a BASIC listing utilizing the "dynamic keyboard" technique. Each line of the BASIC listing in memory is entered as though it is typed directly from the keyboard.

Creating The Modem Save Program

Program 1 should be typed in first. SAVE a copy on tape or disk, since the program destroys part of itself as it is RUN. When you RUN the program, checksums will be calculated to help detect any typing errors you may have made in the DATA statements. If no errors are reported, type NEW, but *do not* reset the computer. It is important that the data which Program 1 POKEs into memory still be there when you type in Program 2. Remember that the NEW command does not clear memory, but only resets pointers.

Program 2 should be typed in exactly as shown; otherwise, when it is RUN, the BASIC pointers may cut off the end of the program. After you finish typing it in, PEEK locations 45 and 46. They should contain the values 49 and 20 respectively. If not, you have either added to or left something out of the program. SAVE a temporary backup copy at this point in case you have problems with the next step.

Now that you have Program 2 typed in, you must attach the tables and machine language programs which were loaded into memory by Program 1. You can do this by typing POKE 45,1:POKE 46,26 and RETURN. This fools BASIC into thinking that Program 2 includes all the memory up to location 6655, the end of the memory loaded by Program 1. You should now SAVE the new Program 2. All further references to Program 2 are to this version of the program which includes the tables and machine language routines.

To test whether all the data from Program 1 was in fact included in Program 2, type SYS 64802 to clear memory, then LOAD Program 2. Next, type in the following direct mode line and hit RETURN:

```
FOR A=6144 TO 6655: SUM=SUM+PEEK(A): NEXT
  : PRINT SUM
```

If the value reported is 62616, then the data has been included. If not, you'll have to LOAD and RUN Program 1 again, type NEW, LOAD the temporary backup copy of Program 2 you made earlier, then try the POKEs to 45 and 46 and the SAVE again. Once your Program 2 passes this test, you won't need Program 1 again.

You're Ready To Dial

When Program 2 is run, the screen will clear and a cursor (__) will appear in the upper-left corner of the screen. You are now ready to dial up CompuServe or another telecommunications service and connect to the modem. Use the F1 key for Control-C; F3 for Control-P (Break); F5 for Control-Q (resume sending); and F7 for Control-S (stop sending). The interrupt service routine signals with an audible alarm when your text buffer is within 256 bytes of being filled.

You then have the option of logging off and saving your text to tape or printer, or remaining on-line while your text is being saved (about five minutes for 16K) and refilling your buffer with more text. These options are implemented by logging off or by depressing F7 (stop sending) and then pushing the British pound sign key. You will then be asked to select from a menu whether you wish to send the text to tape, printer, or both.

After the SAVE, you will be asked if you are still logged on. If not, the program ends. If you respond that you are still logged on, the program resets the text buffer, clears the screen, and you are ready to continue by depressing F5 (resume sending).

After you have saved your text to tape, you can load it back in at any time and use Program 3 to search and select portions for display and printing. When loading the text back into your VIC, you may occasionally get a ?LOAD ERROR message. Don't be concerned – your data should still be intact. After loading the text tape, type NEW and LOAD Program 3.

When RUN, Program 3 will display a set of instructions. Press the SPACE BAR to start and stop the printing of text to screen as many times as desired. When the text is stopped, the S or E key can be used to mark the start or end of selected text. The starting or ending point will be the last character to appear on the screen before the text is stopped. The memory location of that character will appear on the screen, and you will be given the opportunity to change the start (or end) of selected text by changing this number. Once the end of the text has been marked, press the D key to display the selected text. You will then be asked if you would like a hard copy. Pressing Y will send the selected text to the printer.

I would have preferred that the start of selected text be marked from the top of the screen rather than from the bottom, but I couldn't find a reliable way to correlate the location of a particular portion of text in screen memory to its location in the text buffer due to the inherent inconsistency of the text. The capability to adjust the start and end of selected text after it is marked should help.

If you find a BASIC listing in the text which you would like to download, find the starting address of the listing with Program 3. Type NEW and load Program 4. When this program is run, you will be prompted for the starting address of the listing. Enter the address and RETURN. Each of the listings will then appear briefly, one at a time, near the top of the screen. It may take awhile, depending on the length of the listing, but when it is done, your listing has become a program. You can now delete Program 4 from the end of your downloaded program and SAVE or RUN it.

When Program 4 reaches the end of a program listing, it terminates with an error message. This is due to the absence of a line number or valid BASIC statement. Although this does end the program at the proper point, maybe you can come up with a more elegant way to terminate execution.

Program 2

Line 2 conditions the interrupt service routine to sound an audible alarm when the text buffer is within 256 bytes of being filled. It also changes the IRQ Vector to point to the interrupt routine and sets upper/lowercase mode.

Line 3 moves the top of memory down to protect the machine language routines, lookup tables, and text buffers.

Line 4 resets the BASIC pointers which were moved when the programs were saved. This must

SJB DISTRIBUTORS. THE MOST COMPETITIVE PRICES ON COMMODORE.

commodore

NEW COMMODORE PRODUCTS

The Executive 64	Call
CBM B128-80	$ 825
CBM B256-80	1095
CBM BX700	2990
B Series Software	Call
CBM 1520 Plotter	169
CBM 1526 Printer	349

SOFTWARE FOR CBM 64

BUSINESS

WordPro 3+/64 w/Spell Right Plus	$ 79
Spell Right Plus	55
Calc Result (Advanced)	125
Calc Result (Easy)	75
Busicalc II	95
Mirage Concepts (Powerful Data Base)	95
M File (merge with WordPro)	89
Home Utilities	49
64 Mailing List (Galactic)	28
The Manager	50
Home Accountant (continental)	75
Code Writer (Writes Basic Programs)	95
Stock (investment analysis)	80
Agricultural Management	Call
General Ledger, A/R, A/P, P/R, Inv	Call

RECREATION

Assembler Package (cassette or disk, compiled, includes editor, loader, disassembler)	39
Sprite Master (access)	30
Neutral Zone (access)	35
Space Belt	19
Pet Emulator	30
Coco II (build your own games)	40
Vic Tree (programmers utilities)	75
Micro-Term (save to printer,disk)	39
Hesmon	35
Synthesound	45
Gothmogs Lair	30
Road Toad	15
Commodore Games	Call

INTERFACES & ACCESSORIES

80 Column Expander	$159
VIC 1600 Modem	95
VIC 1650 (auto answer, auto dial)	150
VIC 1525 Graphic Printer	225
VIC 1530 Datasette Recorder	65
VIC 1541 Disk Drive	249
VIC Switch (connect 8 64's or Vics to printer, dd)	149
PET-IEEE cable	33
IEEE-IEEE cable (2m)	49
5 Slot Expander for 64	65
Parallel Interface (Epson, Okidata, IDS, NEC)	70
Programmers Reference Guide	18
Verbatim Diskettes (10 per box)	26
Hes Modem	75
ADA 1450	149
ADA 1800 (new)	129
Numeric Keypad	35

VIC PRODUCTS & ACCESSORIES

8K RAM Memory Expansion Cartridge	$ 40
16K RAM	70
24K RAM	105
VIC 3 Slot Expander	27
VIC 6 Slot Expander	70
Gorf (64 also)	30
Omega Race	30
Arcade Joystick - Heavy duty w/2 firing buttons! Great for the VIC or 64	25
Auto Clock	125

MONITORS - GREAT RESOLUTION (64 OR VIC)

CBM 1701 Color Monitor	$ 249
Amdek Color Plus	299
Panasonic TR-120 (w/speaker)	155
Panasonic CT-160	279
BMC (green screen)	95
Video/Audio Cable	15

PRINTERS - LETTER QUALITY

CBM 6400, 40 cps	$1450
Diablo 620, 25 cps	949
Transtar 140 (serial)	1395
Transtar 130, 16 cps (auto load, wp features!)	769
NEC 3500 Series	1600
NEC 7700 Series	2350

PRINTERS - DOT MATRIX

CBM 8023, 150 cps/graphics	$ 545
CBM 4023 Printer	395
Epson FX Printer, 160 cps	549
Epson MX-80 FT w/graftrax	Call
Epson FX-100	859
Okidata 82A, 120 cps (serial and parallel)	429
NEC 8023A (parallel)	429
Okidata 92	559
Star Gemini, 10X	329
Star Gemini, 15	499
Transtar 315 (hi-res, color)	575

COMMODORE BUSINESS SERIES

SuperPet (5 languages, 2 processors)	$1059
CBM 8032 Computer, 80 Column	625
CBM Memory Expansion, 64K	259
CBM 8050, 1 mg. Dual Drive	995
CBM 8250, 2 mg. Dual Drive	1295
CBM D9060, 5 mg. Hard Disk	1995
CBM D9090, 7.5 mg. Hard Disk	2250
CBM 2031, 170K Single Drive (New)	295
DC Hayes Smart Modem	220

BUSINESS SOFTWARE-8032

WordPro 4+ or 5+	$ 309
InfoPro	219
Administrator	489
VisiCalc (expanded)	199
BPI A/R, G/L, Job Cost, Inventory, Payroll	ea.325

MasterCard, Visa, Money Order, Bank Check

COD (add $5) accepted.
Add 3% surcharge for credit cards.
In stock items shipped within 48 hours.
F.O.B, Dallas, Texas (Texas Res. add 5% tax)
All products shipped with manufacturer's warranty.
Prices are subject to change without notice.

TO ORDER CALL TOLL FREE
800-527-4893
800-442-1048
(Within Texas)
Business Hours
Mon.- Fri. 8 to 6, Sat. 10-2
Write for free catalog.

SOFTWARE OF THE MONTH

Mirage Concepts
Machine Language Data Base
2000 Characters PL Record
200 Fields Per Record
Multiple Files Per Disk

PRODUCT OF THE MONTH

INTERPOD (intelligent IEEE RS232, serial interface for VIC or C64) .... $ 179

SJB DISTRIBUTORS INC.
10520 Plano Road, Suite 206
Dallas, Texas 75238
(214) 343-1328

be done before editing or running any part of the program which contains variable names.

Line 5 opens a file to the RS-232 device and sets the baud rate at 300.

Line 20 sets up a zero page pointer to the start of the text buffer. The machine language subroutine then stores text in the buffer beginning at this location and increments once for each character up to the end of available RAM.

Line 40 calls the machine language subroutine loaded by Program 1 which receives data, prints it to the screen, stores it in memory, and sends data from the keyboard to the RS-232 channel. This loop will continue until the British pound sign key is pressed.

Line 125 turns off the out-of-memory alarm.

Lines 130 through 165 display a menu and call an appropriate subroutine, depending upon which menu item is selected.

Lines 170 through 200 are the subroutine which sends data to the printer.

Lines 220 through 240 are the subroutine which saves to tape all memory from the start of BASIC to the last location of data. This is accomplished by moving the start of variables pointer to the location pointed to by memory locations 1 and 2. This subroutine also ascertains whether or not the user is still logged on, continuing the program if he is and ending it if he isn't.

Program 3

Lines 10 through 28 display the instructions on the screen.

Line 30 sets the beginning of text memory and initializes the start of selected text.

Lines 40 through 130 mark the start and end of selected text.

Lines 140 through 150 print the selected text on the screen.

Lines 160 through 190 print out a hard copy of the selected text on the VIC printer.

Lines 200 through 220 are a subroutine which allows adjusting the start of selected text.

Lines 250 through 280 are a subroutine which allows adjusting the end of selected text.

Program 4

Line 60000 inputs the starting address in memory of the program listing.

Lines 60005 and 60010 clear the screen, print the characters which form the listing one at a time, and look for a carriage return character, signaling the end of a line of BASIC.

Line 60020 increments text memory by one and loops back to print the next character.

Line 60030 is executed when the end of a line of BASIC is detected. POKEing a carriage return into the keyboard buffer at this time causes the text on the screen to be centered as if input were from the keyboard.

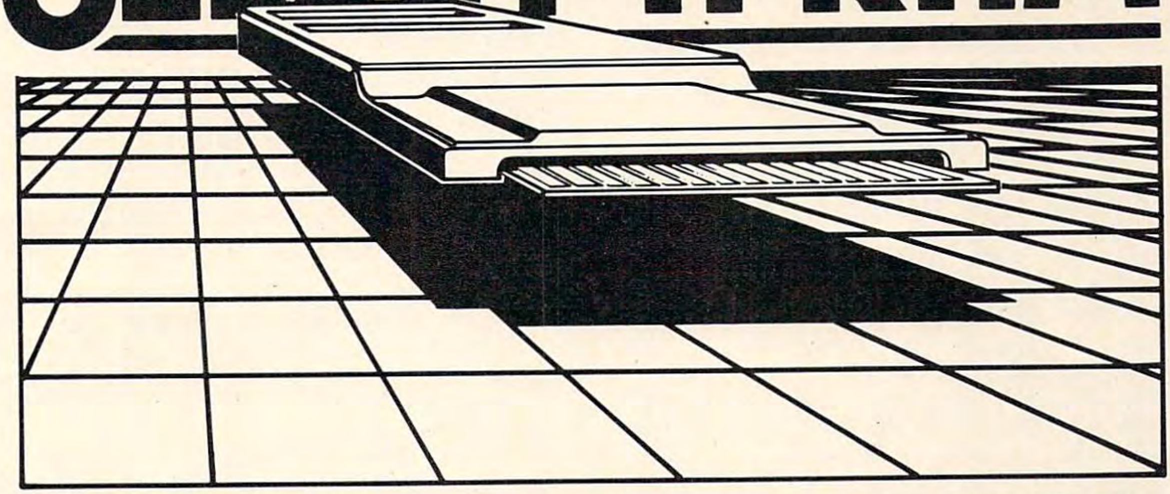
Program 1:

Data Tables And Machine Language

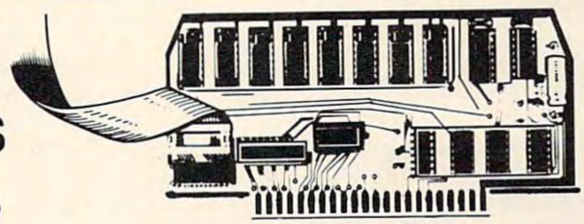
```
10 CK=0: FOR A=10240 TO 10495: READ D: CK
   =CK+D: POKE A,D: NEXT A
20 IF CK<>32753 THEN PRINT "DATA ERROR IN
   ": PRINT "LINES 100-410": STOP
30 PRINT "DATA LINES 100-410 OK"
40 CK=0: FOR A=7936 TO 8191: READ D: CK=C
   K+D :POKE A,D: NEXT A
50 IF CK<>29863 THEN PRINT "DATA ERROR IN
   ": PRINT "LINES 420-730": STOP
60 PRINT "DATA LINES 420-730 OK"
70 CK=0: FOR A=6144 TO 6655: D=PEEK(A+409
   6): CK=CK+D: POKE A,D: NEXT A
80 IF CK<>62616 THEN PRINT "DATA RELOCATI
   ON ERROR": STOP
90 PRINT "DATA RELOCATION OK": END
100 DATA 170, 170, 170, 170, 170, 170, 17
    0, 170
110 DATA 170, 170, 170, 170, 170, 13, 170
    , 170
120 DATA 170, 170, 170, 170, 8, 170, 170,
     170
130 DATA 170, 170, 170, 170, 170, 170, 17
    0, 170
140 DATA 32, 33, 34, 35, 36, 37, 38, 39
150 DATA 40, 41, 42, 43, 44, 45, 46, 47
160 DATA 48, 49, 50, 51, 52, 53, 54, 55
170 DATA 56, 57, 58, 59, 60, 61, 62, 63
180 DATA 64, 97, 98, 99, 100, 101, 102, 1
    03
190 DATA 104, 105, 106, 107, 108, 109, 11
    0, 111
200 DATA 112, 113, 114, 115, 116, 117, 11
    8, 119
210 DATA 120, 121, 122, 91, 92, 93, 94, 9
    5
220 DATA 170, 170, 170, 170, 170, 170, 17
    0, 170
230 DATA 170, 170, 170, 170, 170, 170, 17
    0, 170
240 DATA 170, 170, 170, 170, 170, 170, 17
    0, 170
250 DATA 170, 170, 170, 170, 170, 170, 17
    0, 170
260 DATA 170, 170, 170, 170, 170, 3, 16,
    {SPACE}17
270 DATA 19, 170, 170, 170, 170, 170, 170
    , 170
280 DATA 170, 170, 16, 170, 170, 170, 170
    , 170
290 DATA 170, 170, 170, 170, 170, 170, 17
    0, 170
300 DATA 170, 170, 170, 170, 170, 170, 17
    0, 170
310 DATA 170, 170, 170, 170, 170, 170, 17
    0, 170
320 DATA 170, 170, 170, 170, 170, 170, 17
    0, 170
330 DATA 170, 170, 170, 170, 170, 170, 17
    0, 170
340 DATA 170, 65, 66, 67, 68, 69, 70, 71
350 DATA 72, 73, 74, 75, 76, 77, 78, 79
360 DATA 80, 81, 82, 83, 84, 85, 86, 87
370 DATA 88, 89, 90, 170, 170, 170, 170,
    {SPACE}170
380 DATA 165, 2, 201, 127, 144, 13, 169,
    {SPACE}240
```

64K *for* VIC 20™

SELECT·A·RAM™



SELECT·A·RAM



STANDARD FEATURES

- 8K BLOCKS SELECTABLE FROM THE KEYBOARD OR BY SOFTWARE COMMAND
- TWO EXPANSION SLOTS
- WRITE PROTECTION
- RESET SWITCH
- EXPANDABLE TO 192K WITH ADDITION OF 64K EXPANSION MODULES
- COMPATIBLE WITH ROM CARTRIDGES
- ONE YEAR WARRANTY ON PARTS AND LABOR
- 15 DAY MONEY BACK GUARANTEE

SELECT-A-RAM ....$169.
64K EXPANSION MODULE ....$149.

TRADE-INS ACCEPTED
3K $5 8K$10 16K-$20

VIC 20 IS A TRADEMARK OF COMMODORE ELECTRONICS LIMITED

APS-52A

52K MEMORY FOR ATARI 400/800 .... $119.00

The APS-52A memory expansion boards come assembled and tested with a 90 day warranty covering materials and workmanship. Boards come with complete documentation including operation and installation instructions and a memory test program.

16K TRADE-INS ACCEPTED $15.00

ATARI 400 AND 800 ARE TRADEMARKS OF ATARI,INC.

512-441-3202 PO BOX 43006 Austin, Tx. 78745-0001

APS

ADVANCED·PROCESSOR·SYSTEMS

MasterCard VISA

```
390 DATA 141, 11, 144, 169, 15, 141, 14,
    {SPACE}144
400 DATA 76, 191, 234, 169, Ø, 141, 14, 1
    44
410 DATA 76, 191, 234, 170, 170, 170, 170
    , 170
420 DATA 170, 170, 170, 133, 170, 170, 17
    Ø, 170
430 DATA 20, 170, Ø, 170, 170, 13, 170, 1
    70
440 DATA 134, 135, 170, 136, 170, 170, 17
    Ø, 170
450 DATA 170, 170, 170, 170, 170, 170, 17
    Ø, 170
460 DATA 32, 33, 34, 35, 36, 37, 38, 39
470 DATA 40, 41, 42, 43, 44, 45, 46, 47
480 DATA 48, 49, 50, 51, 52, 53, 54, 55
490 DATA 56, 57, 58, 59, 60, 61, 62, 63
500 DATA 64, 193, 194, 195, 196, 197, 198
    , 199
510 DATA 200, 201, 202, 203, 204, 205, 20
    6, 207
520 DATA 208, 209, 210, 211, 212, 213, 21
    4, 215
530 DATA 216, 217, 218, 91, 92, 93, 94, 95
540 DATA 32, 65, 66, 67, 68, 69, 70, 71
550 DATA 72, 73, 74, 75, 76, 77, 78, 79
560 DATA 80, 81, 82, 83, 84, 85, 86, 87
570 DATA 88, 89, 90, 59, 60, 61, 62, 20
580 DATA 162, 2, 32, 198, 255, 32, 228, 2
    55
590 DATA 170, 240, 36, 166, 144, 208, 32,
     41
600 DATA 127, 170, 189, Ø, 25, 201, 13, 2
    Ø8
610 DATA 3, 32, 218, 25, 32, 210, 255, 160
620 DATA Ø, 132, 212, 145, 1, 230, 1, 208
630 DATA 2, 230, 2, 234, 24, 144, 209, 32
640 DATA 204, 255, 169, 164, 32, 210, 255
    , 169
650 DATA 157, 32, 210, 255, 162, 2, 32, 2
    Ø1
660 DATA 255, 32, 228, 255, 240, 7, 170,
    {SPACE}189
670 DATA Ø, 24, 32, 210, 255, 32, 204, 255
680 DATA 165, 197, 201, 6, 240, 3, 24, 144
690 DATA 167, 96, 168, 169, 32, 32, 210,
    {SPACE}255
700 DATA 152, 96, 66, 67, 68, 69, 70, 71
710 DATA 72, 73, 74, 75, 76, 77, 78, 79
720 DATA 80, 81, 82, 83, 84, 85, 86, 87
730 DATA 88, 89, 90, 59, 60, 61, 62, 20
```

Program 2: Modem Save

```
1 PRINT"{CLR}{5 SPACES}MODEM SAVE":PRINT:
  FORI=1TO3000:NEXT
2 POKE6371,PEEK(56)-1:POKE788,224:POKE789
  ,24:POKE36869,194
3 POKE644,24:POKE52,24:POKE56,24
4 POKE45,72:POKE46,20:POKE47,72:POKE48,20
  :POKE49,72:POKE50,20
5 OPEN2,2,3,CHR$(6)
20 POKE1,Ø:POKE2,26
30 PRINT" {CLR}"
40 SYS6528
125 POKEØ,PEEK(2):POKE2,26
130 PRINT"{CLR}":PRINT"1. SAVE TO TAPE":P
    RINT"2. PRINTER":PRINT"3. TAPE & PRIN
    TER"
140 PRINT:PRINT"MAKE A SELECTION"
150 GETS$:IFVAL(S$)<1ORVAL(S$)>3THEN150
160 ONVAL(S$)GOSUB220,170,170:IFA$="Y"THE
    N20
165 END
170 OPEN4,4:FORK=8192TOPEEK(Ø)*256+PEEK(1
    ):PRINT#4,CHR$(PEEK(K)AND127);:NEXT
190 IFS$="3"THEN210
200 RETURN
220 POKE45,PEEK(1):POKE46,PEEK(Ø):SAVE"MO
    DEM DATA":POKE45,49:POKE46,20
225 PRINT"STILL LOGGED ON?(Y/N)"
230 GETA$:IFA$=""THEN230
240 RETURN
```

Program 3:
Text Search For Display Or Printout

```
5 PRINT"{CLR}":POKE56,26
10 PRINT"THE TEXT RESIDES IN":PRINT"MEMOR
   Y STARTING AT":PRINT"LOCATION 6656":PR
   INT
20 PRINT"PUSH {RVS}SPACE{OFF} TO START":P
   RINT"AND STOP TEXT SEARCH":PRINT
24 PRINT
25 PRINT"PUSH {RVS}S{OFF} AND {RVS}E{OFF}
    TO":PRINT"MARK START AND END":PRINT"O
   F TEXT FOR DISPLAY":PRINT
28 PRINT"{RVS}D{OFF} DISPLAYS SELECTED":P
   RINT"TEXT"
30 J=6656:SA=J
35 GETA$:IFA$<>" "THEN35
40 PRINTCHR$(PEEK(J)AND127);:POKE212,Ø
50 GETA$:IFA$<>" "THENJ=J+1:GOTO40
60 GETA$:IFA$=""THEN60
110 IFA$="S"THENSA=J:PRINT:PRINT:PRINT"ST
    ART ADDRESS=";SA;"OK?":GOSUB200
120 IFA$="E"THENEA=J:PRINT:PRINT:PRINT"EN
    D ADDRESS=";EA;"OK?":GOSUB250
130 IFA$=" "THENJ=J+1:GOTO40
140 IFA$<>"D"THEN60
150 PRINT:PRINT:PRINTSPC(4)"SELECTED TEXT
    ":FORI=SATOEA:PRINTCHR$(PEEK(I)AND127
    );:NEXT
160 PRINT:PRINT:PRINT"HARD COPY?(Y/N)"
170 GETA$:IFA$=""THEN170
175 IFA$<>"Y"THENEND
180 OPEN4,4:FORI=SATOEA:PRINT#4,CHR$(PEEK
    (I)AND127);:NEXT
190 PRINT#4:CLOSE4:END
200 GETB$:IFB$=""THEN200
210 IFB$<>"Y"THENINPUT"START ADDR.";SA:PR
    INT"{RVS}SPACE{OFF} TO CONTINUE":RETU
    RN
220 PRINT"{RVS}SPACE{OFF} TO CONTINUE":RE
    TURN
250 GETB$:IFB$=""THEN250
260 IFB$<>"Y"THENINPUT"END ADDR.";EA:PRIN
    T"PRESS {RVS}D{OFF} TO DISPLAY":RETUR
    N
280 PRINT"PRESS {RVS}D{OFF} TO DISPLAY":R
    ETURN
```

Program 4: BASIC Download

```
60000 PRINT"START ADDRESS OF":INPUT"PROGR
      AM";I
60005 PRINT"{CLR}{2 DOWN}"
60010 PRINTCHR$(PEEK(I)AND127);:IFPEEK(I)
      =13THEN60030
60020 I=I+1:GOTO60010
60030 PRINT"I="I+2":GOTO60005{HOME}":POKE
      198,2:POKE631,13:POKE632,13
```
©

Termulator For The 64

Gordon C. Lyman

"Termulator" is a speedy, machine language program which emulates a terminal program. It thus gives you an alternative if you find BASIC terminal programs too slow or if you cannot find suitable programs available commercially. You don't need to know machine language to type in and use this program. Termulator is limited to full duplex operation.

After buying a Commodore 64 computer and a VICmodem, I soon discovered that the terminal program supplied with the VICmodem would not run on the 64. I tried using a terminal program written in BASIC, but found it too slow for my purposes. Also, I could not find a terminal program offered for sale for the 64, so I wrote "Termulator" (*term*inal em*ulator*), a machine language program which is quite simple in operation.

Basically, the program gets a character from the keyboard, sends the character via modem, receives a character from the modem, and finally displays it on screen. This simple logic limits the program's ability to full duplex operation; however, I have never required anything but full duplex operation. The program utilizes RAM in the range $0900-$8500 as a receive buffer, storing the text displayed on the screen into memory. Termulator consists of three basic sections: initialization, main loop, and cursor subroutine. Let's look at each one in some detail.

Initialization ($C000-$C048)

Termulator uses the Kernal routine "CLALL" ($FFE7) to close all files, just in case any have been left open. Next, the value $00 is stored in the RS-232 command register ($0294) and the value $06 is stored in the RS-232 control register ($293).

The next instructions set up a filename for the modem file. The location of the filename is loaded into the X and Y registers, and the length of the name is loaded into the accumulator. Now the important part: the first two bytes of the modem filename must be the RS-232 control and command registers. Then, by using the Kernal routine "SETNAM" ($FFBD), the RS-232 interface is instructed to operate according to the RS-232 control and command registers. In this case, the RS-232 interface will operate at 300 baud, with no parity checking, one stop bit, and an eight-bit word length. In order to change these, you must change the values that are loaded into these registers. For further explanation, see the *Commodore 64 Programmer's Reference Guide*.

A pointer in the zero page of memory is initialized to the start of the receive buffer. This buffer starts at $0900 in order to leave a cushion between the start of BASIC at $0800 and the buffer area. The pointer will be used by the main routine to store the text received into this buffer for future manipulations. The limit of memory pointer is reset in order to protect the file buffers which will be allocated when opening a file for the modem. The limit of memory pointer is also set low enough to protect a monitor or other program stored within the top 6656 bytes of RAM.

The program next sets up the logical first and secondary addresses and opens the modem file. This automatically allocates 512 bytes at the top of free RAM for input and output buffers. The accumulator is loaded with the file number, the X register is loaded with the device number, and the Y register is loaded with the secondary address, which would be a command to the modem. The value $FF loaded into the Y register means no command to the device. Then the "SETLFS" ($FFBA) and "OPEN" ($FFC0) Kernal routines are called.

The ASCII data from $C0F4 is displayed until a zero value is found. This includes the character codes to change to upper/lowercase and display white characters, as well as a title message.

The Main Loop ($C04A-$C0BF)

The Kernal routine "STOP" ($FFE1) is called, which will return a $00 in the accumulator if the stop key is pressed. If the stop key is pressed, all files are closed and the program stops; otherwise, the program branches to set the input device to device number 0 (the keyboard).

The Kernal routine "GETIN" ($FFE4) is used to return one byte from the keyboard buffer as an ASCII value in the accumulator. If the keyboard buffer was empty, a $00 is returned and the program will branch to the modem input routine. Otherwise, the ASCII value from the keyboard is stored in a zero page location ($6A) for later processing. The ASCII value from the keyboard is translated into standard ASCII by selecting the corresponding value from a list, 256 bytes long, starting at $C226. This is required because Commodore ASCII is not the same as standard ASCII. Also in this list of data are the ASCII values for the Control A through Control Z. When you wish to send a control character while using the program, type the appropriate letter key while holding down the Commodore key. Another list, starting at $C126, contains the Commodore ASCII for the reverse translation. The Kernal routine "CHROUT" ($FFD2) is used to send the byte, now in the accumulator, over the modem.

The Kernal routines "CHKIN" ($FFC6) and "CHRIN" ($FFE4) are used to input a byte from the modem. Then this byte, which is standard ASCII, is translated to Commodore ASCII and stored in zero page (at $6A).

If the value returned from the modem was null ($00), the program will branch back to the beginning of the main routine.

To erase the cursor before outputting to the screen, a space and cursor-left are displayed. Then the byte that was received from the modem is printed on the screen.

A check is made to see if the character received, now in the accumulator, is a delete. If it is, the receive buffer pointer is decremented and the program returns to the start of the main loop; if not, the receive buffer pointer is incremented. If the pointer has reached the limit of memory pointer, it is reset to $0900. The character is stored in the receive buffer, and the program returns to the start of the main loop.

The Cursor Subroutine ($C0C0-$C0F3)

The least significant byte of the Commodore 64's jiffy clock is used as a timer for the cursor. This byte is compared to the value $15, which is the length of time the cursor takes to flash on or off. By changing this value in location $C0C3, you can change the speed at which the cursor flashes. If the timer has not expired, then the RTS instruction at $C0C6 will return to the main routine.

If the timer has expired, it is reset and a flag stored at $6B is checked. This flag will be either $00 or $FF. If the flag is set to $FF, it will be cleared to $00 and a space which turns the cursor off will be displayed.

If the flag was clear, then the program branches to set the flag to $FF and displays a reversed space which turns the cursor on. After the cursor is turned either on or off, a cursor-left is displayed. This is done so that the next thing displayed will be in the right position. The program then returns to the main routine.

How To Use Termulator

Type in and RUN Program 1, which is a BASIC program that will load the machine language for Termulator into RAM. If any errors are detected, it will display the message ERROR IN BLOCK # x. You will need to check from $C300 to $C337 by hand. If no errors are found, the program is ready to run. Just type SYS 49152.

Once you've got a working version of Termulator, you can eliminate the trouble of having to run the BASIC loader program again by making a copy of the machine language on tape or disk. To do this, you'll need either a monitor program or a program like "Machine Language Saver" (COMPUTE!, June 1983). SAVE the contents of memory from 49152-49976 ($C000-$C338). When you reload the machine language, you start the program just as before, by typing SYS 49152.

Alternatively, you could use Program 2. This POKEs in a short routine which creates a tape copy of the Termulator machine language. Type in and RUN Program 2, insert a blank tape in the datassette, and type SYS 52736. You should see on your screen the prompt PRESS PLAY & RECORD ON TAPE, at which point you are ready to make the copy. You can reload Termulator from this tape by typing LOAD " ",1,1.

Program 1: Termulator – BASIC Loader

```
100 FOR M=49152 TO 49975
110 READ D:POKE M,D:NEXT
120 FOR L=0 TO 11
130 LN=L*70+49152
140 FOR C=0 TO 69
150 IF LN+C>49975 THEN 180
160 T=T+PEEK(LN+C)
170 NEXT C
180 READ CS
190 IF CS<>T THEN PRINT"ERROR IN LINES";L
    N;" -";LN+63:STOP
200 T=0:NEXT L
210 PRINT"{3 DOWN}TERMULATOR LOADED SUCCE
    SSFULLY{2 DOWN}"
220 PRINT"TYPE SYS 49152 TO START"
230 END
49152 DATA 32,231,255,169,0,141,148
49159 DATA 2,169,6,141,147,2,169
49166 DATA 0,133,97,133,99,169,9
49173 DATA 133,98,169,133,133,56,169
49180 DATA 2,162,147,160,2,32,189
49187 DATA 255,169,128,162,2,160,255
49194 DATA 32,186,255,32,192,255,169
49201 DATA 240,141,32,208,169,240,141
49208 DATA 33,208,162,0,189,244,192
49215 DATA 240,9,32,210,255,232,76
49222 DATA 60,192,234,234,32,225,255
49229 DATA 208,4,32,231,255,0,162
49236 DATA 0,134,153,32,228,255,240
```

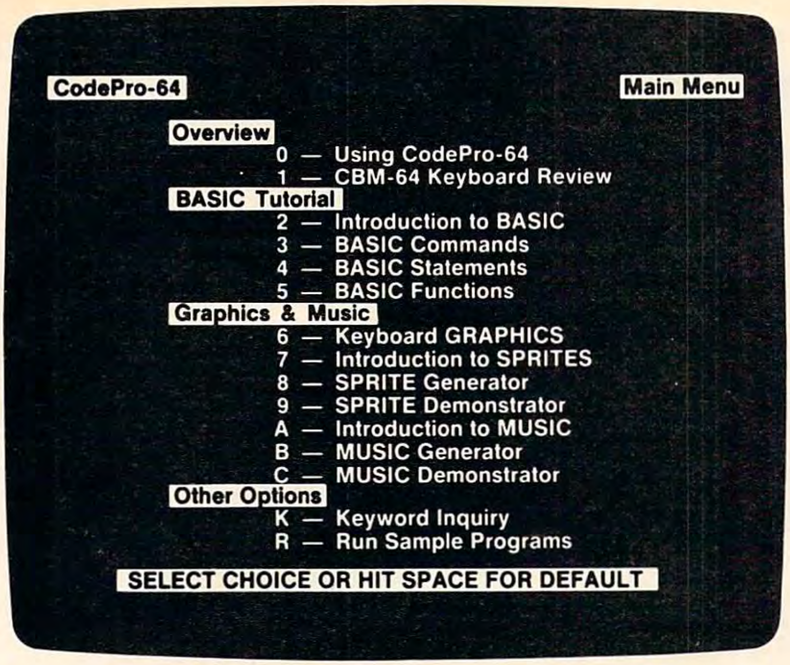

READ THE LABEL
THE FILING ASSISTANT
DATA BASE MANAGEMENT for the COMMODORE 64
This data management package allows you to create record formats, sort information, and produce reports. It includes the following features:
Design records up to 256 characters, up to 25 fields, (up to 35 characters per field)
Designate up to 10 two-level sorts
"Step through" records while displaying selected fields to screen
Add, copy, change, or delete records
"Find" a record by key field, or "locate" by record number
Set function keys to insert a frequently used word, phrase, zip code, etc.
No programming or code necessary. On-screen prompts lead to full operation in one to two hours.
THE FILING ASSISTANT contains a built-in mail label program that allows you to print a 1 to 6 line label. You can include more than one field per line and print up to four labels across the page. Labels to be printed can be selected on one or more fields.
The program also allows you to design your own report formats. Your reports may include:
Automatic record counting
Computation between fields
Totals and/or sub-totals per column
Record selection by one or more fields
TECHNICAL NOTE
This program supports the serial or the user port —you can configure just about any printer. It can be used with more than one 1541 disk drive or with the "C-64 Link-IEEE Interface" which allows you to work with the Commodore 2031, 4040, and 8050 disk drives.
THE FILING ASSISTANT is part of a series. It will integrate with THE WRITER'S ASSISTANT (word processing) and THE SPREADSHEET ASSISTANT (forecasting and financial analysis).
Even if you are a beginner THE FILING ASSISTANT will allow you to manage personal and business records. The ASSISTANT SERIES provides you with the most complete and practical information management package available for the Commodore 64.
RAINBOW Computer Corporation
490 Lancaster Pike
Frazer, Pennsylvania 19355
www.commodore.ca

```
49243 DATA 22,133,106,201,133,208,3
49250 DATA 32,55,195,162,128,32,201
49257 DATA 255,166,106,189,55,194,32
49264 DATA 210,255,162,128,32,198,255
49271 DATA 32,228,255,170,189,55,193
49278 DATA 133,106,162,3,134,154,32
49285 DATA 192,192,165,106,240,191,169
49292 DATA 32,32,210,255,169,20,32
49299 DATA 210,255,165,106,32,210,255
49306 DATA 201,20,208,11,164,99,208
49313 DATA 2,198,98,198,99,76,74
49320 DATA 192,230,99,208,12,230,98
49327 DATA 164,98,196,56,208,4,160
49334 DATA 9,132,98,164,99,145,97
49341 DATA 76,74,192,165,162,201,21
49348 DATA 16,1,96,160,0,132,162
49355 DATA 164,107,240,12,160,0,132
49362 DATA 107,169,32,32,210,255,24
49369 DATA 144,19,160,255,132,107,169
49376 DATA 18,32,210,255,169,32,32
49383 DATA 210,255,169,146,32,210,255
49390 DATA 169,157,32,210,255,96,5
49397 DATA 14,147,17,17,17,17,17
49404 DATA 17,32,32,32,32,32,32
49411 DATA 32,212,197,210,205,45,213
49418 DATA 45,204,193,212,207,210,32
49425 DATA 32,157,157,13,13,32,32
49432 DATA 32,32,32,32,32,32,32
49439 DATA 32,66,89,32,199,46,32
49446 DATA 204,89,77,65,78,13,13
49453 DATA 13,0,234,0,0,0,0
49460 DATA 0,0,0,0,0,0,137
49467 DATA 0,0,0,0,20,0,0
49474 DATA 0,0,13,0,0,146,134
49481 DATA 0,138,0,0,0,0,0
49488 DATA 0,0,0,0,0,0,0
49495 DATA 32,33,39,35,36,37,38
49502 DATA 39,40,41,42,43,44,45
49509 DATA 46,47,48,49,50,51,52
49516 DATA 53,54,55,56,57,58,59
49523 DATA 60,61,62,63,64,193,194
49530 DATA 195,196,197,198,199,200,201
49537 DATA 202,203,204,205,206,207,208
49544 DATA 209,210,211,212,213,214,215
49551 DATA 216,217,218,91,92,93,94
49558 DATA 95,0,65,66,67,68,69
49565 DATA 70,71,72,73,74,75,76
49572 DATA 77,78,79,80,81,82,83
49579 DATA 84,85,86,87,88,89,90
49586 DATA 0,0,0,0,0,0,0
49593 DATA 0,137,0,0,0,0,20
49600 DATA 0,0,0,0,13,0,0
49607 DATA 146,134,0,138,0,0,0
49614 DATA 0,0,0,0,0,0,0
49621 DATA 0,0,32,33,34,35,36
49628 DATA 37,38,39,40,41,42,43
49635 DATA 44,45,46,47,48,49,50
49642 DATA 51,52,53,54,55,56,57
49649 DATA 58,59,60,61,62,63,64
49656 DATA 193,194,195,196,197,198,199
49663 DATA 200,201,202,203,204,205,206
49670 DATA 207,208,209,210,211,212,213
49677 DATA 214,215,216,217,218,91,92
49684 DATA 93,94,95,0,65,66,67
49691 DATA 68,69,70,71,72,73,74
49698 DATA 75,76,77,78,79,80,81
49705 DATA 82,83,84,85,86,87,88
49712 DATA 89,90,0,0,0,0,0
49719 DATA 0,0,0,0,0,0,0
49726 DATA 0,0,0,0,0,0,13
49733 DATA 0,0,0,0,0,0,8
49740 DATA 0,0,0,0,0,0,0
49747 DATA 0,0,0,0,32,33,34
49754 DATA 35,36,37,38,39,40,41
49761 DATA 42,43,44,45,46,47,48
49768 DATA 49,50,51,52,53,54,55
49775 DATA 56,57,58,59,60,61,62
49782 DATA 63,64,97,98,99,100,101
49789 DATA 102,103,104,105,106,107,108
49796 DATA 109,110,111,112,113,114,115
49803 DATA 116,117,118,119,120,121,122
49810 DATA 91,92,93,94,95,0,0
49817 DATA 0,0,0,0,0,0,0
49824 DATA 0,0,0,0,0,0,0
49831 DATA 0,0,0,0,0,0,0
49838 DATA 0,0,0,0,0,0,0
49845 DATA 0,0,0,0,0,0,0
49852 DATA 3,17,19,0,3,19,0
49859 DATA 0,0,0,0,0,0,16
49866 DATA 0,0,0,0,0,0,0
49873 DATA 0,0,0,0,0,0,0
49880 DATA 11,9,20,0,7,0,13
49887 DATA 0,0,14,17,4,26,19
49894 DATA 16,1,5,18,23,8,10
49901 DATA 12,25,21,15,0,6,3
49908 DATA 24,22,2,0,65,66,67
49915 DATA 68,69,70,71,72,73,74
49922 DATA 75,76,77,78,79,80,81
49929 DATA 82,83,84,85,86,87,88
49936 DATA 89,90,0,0,0,0,0
49943 DATA 0,0,0,0,0,0,0
49950 DATA 0,0,0,0,0,0,0
49957 DATA 0,0,0,0,0,0,0
49964 DATA 0,0,0,0,0,0,0
49971 DATA 0,0,0,0,96
50000 DATA 9342,10115,8611,7872,2344,7993
50010 DATA 2536,8449,1657,3439,1123,1416
```

Program 2: Termulator – Tape Copy

```
100 FOR M=52736 TO 52760
110 READ D:POKE M,D:CK=CK+D:NEXT
120 IF CK<>3515 THEN PRINT"ERROR IN DATA
    {SPACE}STATEMENTS":STOP
130 PRINT"{2 DOWN}INSERT TAPE AND TYPE"
140 PRINT"{DOWN}{5 RIGHT}SYS 52736"
150 END
160 DATA 162, 1, 32, 186, 255, 169, 0
170 DATA 133, 106, 32, 219, 255, 169, 192
180 DATA 133, 107, 169, 106, 162, 69, 160
190 DATA 195, 32, 216, 255
```

©



NEW! For the Commodore 64™ ANNOUNCING

CodePro-64™

A new concept in interactive visual learning . . .

Now you can learn to code in BASIC and develop advanced programming skills with graphics, sprites and music—**visually.** You learn by interacting with CodePro-64, a new concept in interactive visual learning.

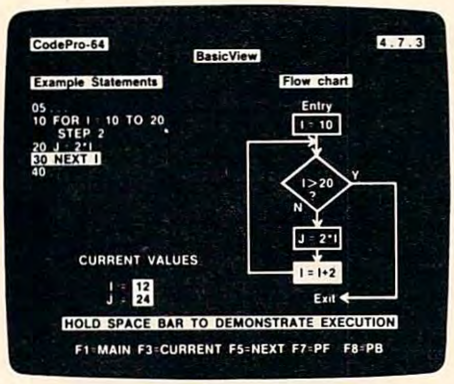
SEE PROGRAM EXECUTION

Imagine actually seeing BASIC statements execute. CodePro-64 guides you through structured examples of BASIC program segments. You enter the requested data or let CodePro-64 do the typing for you. (It will not let you make a mistake.)

After entering an example you invoke our exclusive **BasicView™** which **shows you how the BASIC program example executes.**

You step through and actually see the execution of sample program statements by simply pressing the space bar. CodePro-64 does the rest.

You see statements with corresponding **flow chart graphics** and variable value displays. You learn by visual examples.



EXTENSIVE TUTORIAL

CodePro-64's extensive tutorial guides you through each BASIC command, program statement, and function. You get clear explanations. Then you enter program statements as interactive examples. Where appropriate, you invoke BasicView to see examples execute and watch their flow charts and variables change.

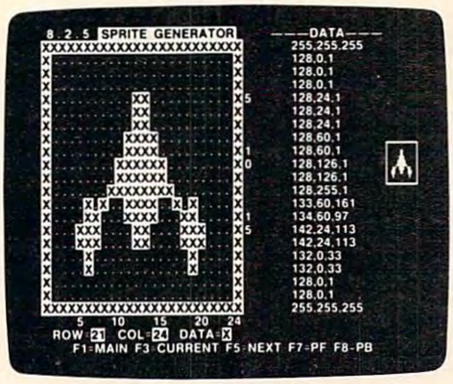
By seeing graphic displays of program segment execution you learn by visual example. You **learn faster and grasp programming concepts easier** with CodePro-64 because you immediately see the results of your input.

You control your learning. You can go through the tutorial sequentially, or return to the main menu and select different topics, or **use keywords** to select language elements to study. You can page back and forth between screens within a topic at the touch of a function key.

CodePro-64 **lets you follow your interests** and practice with interactive examples. But you can never get "lost". F1 will always return you to the main menu. Once you have practiced and mastered the BASIC language elements you move on to more advanced concepts. You learn about sprite and music programming.

SPRITE GENERATOR & DEMONSTRATOR

CodePro-64's sprite generator lets you **define your own sprites** on the screen. You learn how to define sprites and what data values correspond to your sprite definitions. (You can then use these values to write your own programs.) You can **easily experiment** with different definitions and make changes to immediately see the effects.

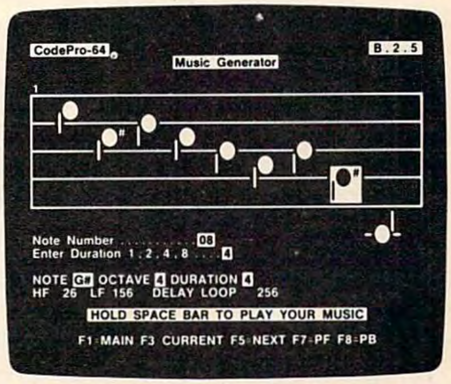


We also help you learn to program with sprites by giving you a **sprite demonstrator** so you can see the effect of changing register values. You can experiment by moving your sprite around in a screen segment, change its color or priority, and see the effects of your changes. You learn by visual examples.

MUSIC GENERATOR & DEMONSTRATOR

To teach you music programming CodePro-64 gives you an interactive music generator and demonstrator. First we help you set all your SID parameters (attack/decay, sustain/release, waveform, etc.). Then you enter notes to play and **we show your tune graphically as it plays,** note by note, on the scale. You learn by seeing and hearing the results of your input.

> **OUR GUARANTEE**
>
> **We guarantee your satisfaction.** You must be satisfied with CodePro-64 for the Commodore-64. Try it for 10 days and if for any reason you are not satisfied return it to us (undamaged) for a full refund. **No risk.**



Our music demonstrator **lets you experiment** with various combinations of music programming parameters and hear the results. You can **quickly modify any of the SID register values** to hear the effects of the change. For example, you could easily change waveform and attack/decay values while holding all other SID values constant. By seeing your input and hearing the result you quickly learn how to create new musical sounds and special sound effects.

AND MORE . . .

We don't have enough space to tell you everything CodePro-64 offers. You need to see for yourself. BASIC tutorials, graphics, sprites, music, keyboard review, sample programs—the main menu shown above gives you just a summary of the contents of this powerful educational product.

Whether you're a beginning programmer or an experienced professional, CodePro-64 will help you improve your Commodore 64 programming skills. We're sure because CodePro-64 was developed by a team of two professionals with **over 25 years** of software development experience.

CodePro-64 is a professional quality educational program for the serious student of personal computing. And it's **fully guaranteed.** Order yours today.

HOW TO ORDER

Order your copy of CodePro-64 today by mail or phone. Send only $59.95 plus $3.00 shipping and handling to:

SYSTEMS MANAGEMENT ASSOCIATES
3700 Computer Drive, Dept. C
Raleigh, N.C. 27609

Please **specify tape or diskette** version. MasterCard/VISA accepted. For faster service on credit card orders call (919)-787-7703.

Commodore 64 is a trademark of Commodore Business Machines, Inc.

Dealer inquiries invited.

C. Regena

Answers To Common Questions

I have appreciated your comments and feedback. Your letters help me in several ways to write a better column. I thought this month I would try to answer some general questions that I frequently see. Most of the questions concern peripherals or debugging, so I'll discuss these two main topics this month.

Do You Really Need Peripherals?

Peripherals are anything extra that you add on to your computer. To use your TI-99/4A, all you really need is the computer itself, a television or monitor so you can see what you're doing, the cord to connect the television and the computer, and the power cord (these cords are included with the computer). If you are writing your own programs, purchasing programs on cassette, or typing programs from magazines, you will need a cassette recorder and a cassette cable. You can use just about any kind of cassette recorder, but the TI Program Recorder is more reliable. The TI-99/4A console seems quite sensitive to the setting of the volume control. Your recorder does need to have a volume control and a tone control. Your *User's Reference Guide* tells how to use the cassette recorder.

To save a program you've written or typed in, use the command SAVE CS1 then press ENTER and follow the cassette instructions. To load a purchased program or a previously saved program, use OLD CS1 and follow the instructions. After you have pressed STOP on the cassette and ENTER on the keyboard, wait for the cursor to return (it may take a few seconds on longer programs), then type RUN to start the program.

By the way, as you are typing in a program, it's a good idea to SAVE your program every 20 minutes or so. It's a disaster to have a program all typed in after hours of effort, then have a sudden power failure that wipes out your program. I always use two cassettes and alternate them during the SAVE procedure just in case the power fails while I'm saving the program.

Most of my writing is for unexpanded computers with no peripherals other than the cassette. TI computers are very powerful machines just as they are, and I like to show readers how much they can do without investing any more money. The TI has many nice features and a very powerful built-in BASIC. The average household or educational user will not need any peripherals to enjoy and use the TI.

What Do You Buy First?

Many computer users soon want to do even more with their computers and begin to add peripherals. Many readers ask what peripherals to buy, and I can't really answer that because it depends on what *you* want to do with the computer. I added a printer first because I needed (wanted) listings of the programs I was writing. Other people can't live without a disk drive, so that's their first purchase.

There are many, many brands of printers available. To use a printer with the TI you need the RS-232 Interface; just make sure your printer is RS-232 compatible. My first printer was an old teletype. If you need to make a cable to connect the printer to the RS-232, the RS-232 manual has the pin connections and all the configuration information. My next printer included the cable – so work with your dealer to make sure you have everything you need.

It is still possible to use just the RS-232 without the peripheral box (known as the "old-style" peripheral system). If the only peripheral you will need is the RS-232, it is unnecessary to buy the Peripheral Expansion Box plus the RS-232 Interface Card. In fact, if you have the Peripheral Box, you can use the old-style RS-232 or the RS-232

card. If you have the old-style RS-232, just plug the Peripheral Box flex cable into the side of the RS-232.

Using the Peripheral Expansion Box is the present method of adding on peripherals to the TI. Inside the box are slots for various cards. There is also room for one disk drive to be inside the box. You may add cards as they become available (or as you can afford them or need them).

A disk drive can be used in many ways. To add a disk drive you also need the Disk Controller box or the Disk Controller card and the Peripheral Expansion Box. The main advantage of a disk system over a cassette system is speed. You may SAVE programs on disk just as on cassette. A full-memory program may take about 3 minutes to load with cassette but only about 20 seconds with the disk system. The disk system is also much faster on any file processing, and thus practically a necessity for business programs. Many business programs require two disk drives. One disk controller can control up to three disk drives. Disk systems are possibly undesirable for some home use or for use in elementary schools because the cassette system is easier for children to use, less expensive, and not as fragile.

The 32K Memory Expansion is available either as a separate box or as a card to go in the Peripheral Expansion Box. One irate reader wrote that in my January column I did not mention that to add the Memory Expansion you also have to buy the Peripheral Expansion Box. The answer is that the Memory Expansion is still available separately in a box that attaches to the side of the computer. The Peripheral Expansion Box is the best way to go if you are adding several peripherals, but if you need only one unit the "old-style" still works. The 32K Memory Expansion does require a command module that can access it. You cannot use the Memory Expansion with the built-in BASIC. TI Extended BASIC and Logo are examples of two of the modules that can use the Memory Expansion.

Computer Enhancements

A modem allows telecommunication – you can connect your computer through telephone lines to another computer such as a large data base or a mainframe "host" computer. Your home computer thus acts as a terminal. To use a modem you'll need the RS-232 Interface and the Terminal Emulator command module. The RS-232 has two ports so you can interface with both a printer and a modem. There are two terminal emulator command modules, and either one will work. They contain the software necessary to set up the communications. Terminal Emulator II also contains speech capabilities, so it is a dual-purpose module. There are many brands of modems and telephone couplers; you just need to make sure the one you use is RS-232 compatible.

The Speech Synthesizer is the peripheral that makes the computer talk. With the "free speech" offer (buy six command modules and get the Speech Synthesizer free), every home with young children should get one. The speech feature adds an extra touch to educational programs. To enable the computer to talk, you need a command module that has speech capabilities.

To program your own speech, you'll need Terminal Emulator II. Words are pronounced phonetically, or you can use numbered allophones, so programming speech takes some experimentation. You can also use the TI Extended BASIC module, but this module has limited speech – only a certain vocabulary (and variations of those words) can be used.

Wired Remote Controllers are available for games or for educational programs. The TI version comes as a pair of joysticks for two-player games.

With a Hex-Bus adapter you can save a program with the Texas Instruments Compact Computer 40 (CC-40) onto wafertape, then load it onto the TI-99/4A.

Alternatives To BASIC

Several languages are available for the TI-99/4A. TI Extended BASIC is probably the first one I would get for someone who likes to program. Extended BASIC comes as a command module, and no extra peripherals are required. Extended BASIC allows multistatement lines, actual subprograms, and complex IF-THEN-ELSE logic. If you like to convert programs from other versions of BASIC, Extended BASIC makes it a little easier. Another feature of Extended BASIC is the DISPLAY AT command to print at a specific location on the screen – and the PRINT USING command allows formatting, which makes it easier to print reports or line up numbers in a column of numbers.

Another main feature of Extended BASIC is sprite capability. You may custom design your own objects just like in TI BASIC, but then you can place the sprite on the screen, designate a color, and put the sprite in motion (all in one statement). For people who like to design games, Extended BASIC is a must. The sprites are a lot of fun to work with.

Logo and Logo II are command modules which are popular in introducing children to programming. The TI version of Logo allows all the common turtle commands, and you can define your own characters and choose colors. The Logo II version has music capabilities. To use Logo or Logo II, you need the memory expansion. If you are a teacher using Logo, be sure to get the *Logo Curriculum Guide.* It is a manual of excellent ideas for using Logo in the classroom. It also includes sample programs.

For machine language, you can get the Editor/Assembler cartridge. The 32K Memory Expansion, Disk Drive, and Disk Controller are required. A less expensive way to try machine language is to use the Mini-Memory module (no peripherals required).

Another language available to TI users is Pascal. Peripherals required are the P-Code Card, 32K Memory Expansion, Disk Drive, Disk Controller, and Peripheral Expansion Box.

I'm sorry I cannot answer your questions about machine language or Pascal. My programming so far has been in BASIC (for several computers) and TI Extended BASIC.

Why Won't The Program Run?

Now to the second main topic – debugging. *Debugging* is a computer term which means finding what's wrong with a program that doesn't work correctly. This month I'd like to give you some tips on how you might pinpoint errors in a program you've typed in but won't run correctly.

Syntax errors are the easiest to find and correct. If you RUN the program, it will stop at any syntax error and tell you exactly what's wrong and in which line. *Syntax* usually refers to a typing error such as a word spelled incorrectly, a comma in the wrong place, unmatched parentheses or quotes, or the wrong number of parameters in a CALL command. The TI catches a lot of typing errors as you are typing in the lines. Others are detected as the program is RUN. Remember that you can type the line number then the down arrow (with FCTN on the TI-99/4A and SHIFT on the TI-99/4) to edit a particular line, then use the arrow keys to move the cursor to the error.

Check line numbers in program transfer statements – GOTO, GOSUB, ON-GOTO, ON-GOSUB, and IF-THEN-ELSE statements. One digit can make a difference in the proper program control. For example, my coordinate geometry program in the February 1983 issue had a typesetting error. Line 760 should have been GOSUB 1860 instead of GOSUB 1850. That one digit caused an error. Several people wrote in very complex solutions to a problem I didn't know existed until I compared line numbers and noticed that one digit. This was one case where there really was a printing error. Now COMPUTE! has the listings printed directly from the computer to avoid such errors.

Check Your DATA Statements

Check to make sure DATA statements are typed correctly. If your program has DATA statements and doesn't run properly, the most likely place for a typing error is in a DATA statement. You may want to review the description of DATA statements in your *User's Reference Manual* or my August 1983 column on DATA and READ statements so you can follow the logic of the READ statements and corresponding DATA statements.

If you get a DATA ERROR, you may not have enough data items to fulfill the READ requirements. The line number given in the error message is the READ statement, so you'll have to find the corresponding DATA statement. Check the DATA statements for the proper placement of commas. It is possible there are commas together with nothing between them – this indicates a null string or " ", and every comma is necessary. Also, make sure you do *not* have a comma at the end of a DATA statement.

Another type of DATA error is that the computer is trying to read a numeric value but gets a string (letters). Again, check the commas. Also make sure you haven't mistyped the number zero and the letter O.

If you have a lot of DATA statements, your eyes may get tired trying to compare printed statements with your typed statements. To try to pinpoint the trouble spot, LIST the lines around the READ statement referred to in the DATA error message. Remember you can list specific lines, such as LIST 640-660. Now PRINT the variables you are reading to find the last good values that were accepted. If you are reading within a FOR-NEXT loop, you can PRINT the index counter to see how far along the loop you are.

Anytime the program stops (BREAK), in this case with an error message, and as long as you don't do any editing, you may PRINT the value of any variable. For example, you may type PRINT B and press ENTER, and the present value of B will be printed. You can then look in the DATA statements to see where that particular value is. The value printed will be the last acceptable value for B, so the next couple of items may contain the error.

An error in a DATA statement may actually cause a problem in a statement other than a READ statement. For example, suppose you have this section of a listing:

```
650 FOR I=1 TO N
660 READ X,Y,G
670 CALL HCHAR(X,Y,G)
680 NEXT I
```

You could get the error message BAD VALUE IN 670. This means X, Y, or G is not acceptable. X must be a number from 1 to 24 for the row number, Y must be from 1 to 32 for the column number, and G must be an ASCII code number. You can PRINT X;Y;G to see what the values for X, Y, and G are. The next step is to see how you got the bad value. In this case, line 660 READs the values from DATA, so you can search through the DATA statements to find a sequence of the three numbers the computer printed. The error will probably be a typing error just before those numbers.

Other Common Errors

There are also errors unrelated to DATA statements. FOR-NEXT errors are usually not difficult to find. Every FOR statement must have a corresponding NEXT statement. Once in a while, however, you can search and search and everything seems matched up correctly. The most likely cause for the error is that a line just before a FOR statement or just before a NEXT statement has 28 characters (or a multiple of 28), so the cursor goes to the next line. You need to press ENTER, but the cursor makes you think you have already pressed ENTER, and you may go ahead and type the next line. The result is a run-together line. If you list that line among several others, they all look right because the numbers line up properly.

To see if this is the problem, LIST only the line containing the FOR or the NEXT to see if it's really there. Warning: The FOR-NEXT error message may list a line number that is really OK; the run-together FOR or NEXT statement may occur before the one listed in the error message. If you use the automatic numbering feature as you type in programs, this problem is less likely to occur.

The run-together line problem may occur anytime you are typing lines that have 28 characters and could cause other problems.

A "glitch" type problem may occur in ON-GOTO and ON-GOSUB statements. The line should be typed in the following example form:

```
200 ON A GOTO 340,550,760,800
```

where there are no spaces between the line numbers. If you happen to type a space in between line numbers then later LIST the line, the space will not be there but it could still be causing an error. If you suspect you are having trouble with an ON-GOTO or ON-GOSUB statement, retype the whole statement. By the way, don't try to second-guess the author. The line numbers do not have to be in numerical order, and you can use the same line number in several of the positions.

These are answers to the most common questions I've been asked. If you still have problems getting a program to run, you may write to me. Be sure to tell me which program you are typing, which computer you are using, the exact error message with the line number, and what happens plus whatever other conditions may contribute to the problem. I want you to be able to use and enjoy these programs; however, it is difficult for me to help you debug if I don't know the exact conditions and line numbers. Please do not ask me how to solve problems with programs written by other authors.

Also, please do not ask me to debug one of your programs or to write a program (or convert a program) for you. It isn't your project if I do it for you, and the joy of programming is accomplishing your own goals. I also am not set up to review programs for you. You may submit them to COMPUTE! directly. I *do* welcome comments or suggestions for future columns that will interest the general TI user.

Since I haven't written about a specific programming technique this month, here's a short graphics display program to try this time. I'll try to have a Christmas present for you in my December column.

```
100 DEF R=INT(16*RND+1)
110 RANDOMIZE
120 FOR I=1 TO 16
130 CALL SOUND(-50,R*110,4)
140 CALL COLOR(I,R,R)
150 CALL SCREEN(R)
160 CALL HCHAR(R+4,R*2,R*R/2,R*R)
170 CALL VCHAR(R+4,R*2,R*R/2,R*R)
180 NEXT I
190 GOTO 110
200 END
```

Line 100 defines a function R to be a random integer from 1 to 16. Every time R is used in later lines, R will be a random integer from 1 to 16 – a lot less typing by using the DEFinition function. ©

Micros With The Handicapped

Susan Semancik & C. Marshall Curtis

Developing A Communications Program

This is the final column in the series on developing a communications program for the handicapped. The final version of the program, written for the VIC with at least 8K RAM, has the following features:

1. Multiple menu selection, with menus stored within the program.

2. A joystick button used as a one-movement, alternative input device.

3. A faster scanning algorithm used in a passive selection process.

4. Input options:
- **a.** Changing the rate of scanning and the time in which to make a selection.
- **b.** Changing the menu being displayed.
- **c.** Storing messages within the program, which can be retrieved in any order.
- **c.** Audible signal when message is ready for viewing.
- **e.** Erasing characters, words, or sentences within the current message, or erasing a stored message.
- **f.** Saving the program with its menus and messages.

5. Word and character selection from the same menu.

6. Automatic spacing after words and prevention of word-splitting in messages.

User Options

The top line of the screen displays the input options in green. If one is selected, its choices are presented in cyan on the second screen line. The next 13 screen lines are used for menus, which are displayed in blue. Any messages are reverse-fielded in red in the bottom 8 screen lines.

Instead of every menu entry being scanned until the user responds, the columns of the menu and options are alternately scanned until one is selected. If a menu column is chosen, its rows are scanned until no more selections are made from that column.

A "debouncer" line has been added to the input routine so that the selection timing isn't started until the joystick button is released. This should help avoid unexpected and unwanted multiple selections.

When the message area is filled, the user cannot add to it until after selecting the message erase option. This allows the user to first save the message or signal someone to read it. A saved message is erased by referring to it by a positive number under the ERASE option. Choosing a zero instead will clear the message display area for further message formation.

The message review option allows multiple messages to be displayed in any order. Any current message is preserved, and then restored after the review mode is done.

To make the program easier to use by people with limited computer experience, menus and messages are stored in the program within DATA statements. The program adds new messages to DATA statements by means of the "dynamic keyboard" method. The first DATA statement contains information that the program updates when the user changes response time or the number of stored messages. Then enough DATA line numbers have been reserved for nine messages, each taking up four DATA lines. The remaining DATA lines contain menu parameters and menu entries by rows.

Suggestions

The complete program with all of its features will not fit in an unexpanded VIC. We suggest at least an 8K system to make this a useful communication tool. If you eliminate multiple menus and some of the input options, and limit the number of messages to be stored even further, you can make the program fit in the unexpanded VIC.

One helpful addition would be the ability to change an entry of a menu or add a new menu. This would allow the user even greater independence in creating a tailored communication tool.

We hope that this series has been helpful to you, or at least given you a new perspective on designing specialized communication programs for the handicapped. There are many other ways to approach the problem, but most of these require a larger and more expensive system than we have considered here. For those wishing to further explore this topic, we recommend the June 1983 issue of *IEEE MICRO* magazine, which contains an excellent article and references on computerized, anticipatory letter-selection programs.

There are also many new computer interface devices being developed for the handicapped user. For instance, an Atari compatible joystick that is mouth operated is being advertised for $65 by KY Enterprises, 195 Claremont, Suite 288, Long Beach, CA 90803, (213)433-5244.

If anyone knows of other computer programs or devices to help the handicapped, or if anyone develops enhancements to our final program, please let us know and we will share the information with the rest of our readers in this column. You can contact us through Jean Trafford, Secretary, The Delmarva Computer Club, P.O. Box 36, Wallops Island, VA 23337.

VIC Communications Program

```
10 W=22:PM=8*W:DIMS(W),L(W),O$(11):SP=409
   6:CP=37888-SP:XJ=37137
14 POKE37139,Ø:SM=SP+W*23-PM:S=36876:P$="
   .?!"
20 GOSUB95Ø:PRINT"{CLR}{GRN}";I$;"{BLU}":
   FORI=ØTOPM-1:POKESM+I+CP,2:NEXTI
50 GOSUB1Ø1Ø:IFTTHENGOSUB197Ø:T=Ø
60 P1=SP:P2=P1+21:A=128:GOSUB11ØØ:GOSUB12
   ØØ:A=-128:GOSUB11ØØ
70 IFX=ØTHEN14Ø
80 P1=SP:P2=SP+LEN(I$)-1:GOSUB13ØØ:IFX=ØT
   HEN14Ø
85 X1=Y:X2=E-2
90 N=Z:PRINT"{HOME}{DOWN}{CYN}";:A$=O$(N)
   :IFN=2THENA$=LEFT$(O$(2),2*NM)+"D"
100 PRINTA$:P1=SP+W:P2=P1+LEN(A$)-1:GOSUB
    13ØØ
110 IFX=ØTHENGOSUB13ØØ:GOTO11Ø
120 ONNGOSUB14ØØ,15ØØ,16ØØ,18ØØ,2ØØØ
130 GOSUB115Ø:IFCL=ØTHENFORI=X1TOX2:POKEI
    ,PEEK(I)-128:NEXTI
135 CL=Ø:GOTO6Ø
140 IFFTHEN6Ø
145 C=1:PRINT"{BLU}";
148 FORR=1TORM:A=128:GOSUB1Ø9Ø:GOSUB11ØØ:
    NEXTR
150 GOSUB12ØØ:FORR=1TORM:A=-128:GOSUB1Ø9Ø
    :GOSUB11ØØ:NEXTR
160 IFX=ØTHEN22Ø
170 R=1
175 A=128:GOSUB1Ø9Ø:E=P2:FORI=P2TOP1STEP-
    1:IFPEEK(I)<>32THENE=I:I=P1
180 NEXTI:X=Ø:P=Ø
190 P2=E:A=128:GOSUB11ØØ:GOSUB12ØØ
200 P2=E:A=-128:GOSUB11ØØ
210 IFXTHEN23Ø
215 R=R+1:IFR<=RMTHEN175
220 C=C+1:IFC<=CMTHEN148
225 GOTO6Ø
230 IFC<CM-CC+1THEN3ØØ
240 I=P1
245 IFL(C)=1THEN28Ø
250 POKEI,PEEK(I)+128:GOSUB12ØØ
260 POKEI,PEEK(I)-128:IFXTHEN28Ø
270 I=I+1:IFI<=ETHEN25Ø
275 GOTO215
280 POKESM+J,PEEK(I)+128:J=J+1:IFJ=PMTHEN
    F=1:GOTO6Ø
290 GOTO148
300 Y=W-(J-W*INT(J/W)+1):IFE-P1>=YTHENGOS
    UB36Ø:J=J+Y+1:IFJ>=PMTHENF=1:GOTO6Ø
310 I=P1
315 POKESM+J,PEEK(I)+128
320 J=J+1:IFJ=PMTHENF=1:GOTO6Ø
330 I=I+1:IFI<=ETHEN315
340 POKESM+J,16Ø:J=J+1:IFJ=PMTHENF=1:GOTO
    6Ø
350 GOTO148
360 FORI=ØTOY:POKESM+J+I,16Ø:NEXTI:RETURN
950 READNM,M,DE:IFM=ØTHEN97Ø
960 FORI=1TOM:READA$,A$,A$,A$:NEXTI
970 READNI,I$:FORI=1TONI:READO$(I):NEXTI:
    RETURN
1000 READRM,CM,BR,BC,SC,CC:FORI=1TOCM:REA
     DL(I):NEXTI:RETURN
1010 GOSUB1ØØØ:S(1)=SC:IFCM=1THEN1Ø2Ø
1015 FORI=2TOCM:S(I)=S(I-1)+L(I-1)+BC:NEX
     TI
1020 PRINT"{HOME}{2 DOWN}{BLU}";:FORR=1TO
     RM:READM$:IFLEN(M$)>WTHENPRINT"{RVS}
     ERROR ROW{OFF}"R:GOTO1Ø4Ø
1030 PRINTM$;:IFLEN(M$)<WTHENPRINT
1040 IFBRTHENFORB=1TOBR:PRINT:NEXTB
1050 NEXTR:RETURN
1090 P1=SP+2*W+(R-1)*W+(R-1)*BR*W+S(C)-1:
     P2=P1+L(C)-1:RETURN
1100 FORI=P1TOP2:POKEI,PEEK(I)+A:NEXTI:RE
     TURN
1150 PRINT"{HOME}{DOWN}";:FORI=1 TO W:PRI
     NT" ";:NEXTI:PRINT"{HOME}";:RETURN
1200 X=Ø:P=Ø
1205 IF(PEEK(XJ)AND32)=ØTHEN12Ø5
1210 IF(PEEK(XJ)AND32)=ØTHENX=1:P=DE
1220 P=P+1:IFP<DETHEN121Ø
1230 RETURN
1300 Z=Ø:Y=P1
1305 E=Ø:FORI=YTOP2:IFPEEK(I)=32THENE=I+1
     :I=P2:Z=Z+1:GOTO132Ø
1310 POKEI,PEEK(I)+128
1320 NEXTI:IFE=ØTHENE=P2+2:Z=Z+1
1330 GOSUB12ØØ:IFXTHENRETURN
1340 FORI=YTOE-2:POKEI,PEEK(I)-128:NEXTI
1350 IFE=<P2THENY=E:GOTO13Ø5
1360 RETURN
1400 ONZGOTO141Ø,142Ø:RETURN
1410 DE=DE-.5*DE:GOTO143Ø
1420 DE=DE+.5*DE
1430 GOSUB1975
1440 RESTORE:READA,A,A:LS=PEEK(63)+256*PE
```

```
     EK(64):PRINT"{CLR}{3 DOWN}"LS"DATA";
     NM","M","DE
1450 PRINT"CLR:T=1:GOTO10{HOME}":POKE198,
     3:FORI=0TO2:POKE631+I,13:NEXTI:END
1500 IFZ>NMTHENRETURN
1510 RESTORE:GOSUB950
1520 FORH=1TONM:IFH=ZTHENH=NM:GOTO1540
1530 GOSUB1000:FORK=1TORM:READA$:NEXTK
1540 NEXTH:GOSUB1975:PRINT"{CLR}{GRN}";I$
     :FORI=0TOPM-1:POKESM+I+CP,2:NEXTI
1550 GOSUB1010:GOSUB1970:CL=1:RETURN
1600 ONZGOTO1610,1700:RETURN
1610 IFM=0THENPRINT"{HOME}{DOWN}{RVS}NO M
     ESSAGES STORED.{OFF}{3 SPACES}":GOSU
     B1200:RETURN
1620 GOSUB1975
1630 GOSUB1150:A$=LEFT$(O$(2),2*M)+"D":PR
     INT"{DOWN}";A$
1640 P1=SP+W:P2=P1+LEN(A$)-1:GOSUB1300
1650 IFX=0THENGOSUB1300:GOTO1650
1655 IFZ>MTHENGOSUB1970:RETURN
1660 RESTORE:READNM,M,DE
1670 FORI=1TOM:IFI=ZTHENI=M:GOTO1680
1675 READA$,A$,A$,A$
1680 NEXTI
1685 K=INT(PM/4+.5):FORI=1TO4:READA$:PRIN
     T"{HOME}"A$:FORB=0TOK-1
1688 POKESM+B+(I-1)*K,PEEK(SP+B):NEXTB,I
1690 PRINT"{HOME}";:FORI=1TOW:PRINT" ";:N
     EXTI:PRINT"{HOME}{GRN}"I$"{CYN}":CL=
     1:GOTO1630
1700 POKES+2,15:FORI=1TO50:POKES,220:FORK
     =1TO5
1710 NEXTK:POKES,0:NEXTI:GOSUB1200:IFX=0T
     HEN1700
1720 POKES+2,0:RETURN
1800 ONZGOTO1810,1830,1870,1900:RETURN
1810 IFJ=0THENRETURN
1820 J=J-1:POKESM+J,32:F=0:RETURN
1830 X=0
1835 IFJ=0THENRETURN
1840 J=J-1:IFPEEK(SM+J)<>160THENPOKESM+J,
     32:F=0:X=1:GOTO1835
1850 IFXTHENJ=J+1:RETURN
1860 POKESM+J,32:GOTO1835
1870 X=0
1873 IFJ=0THENRETURN
1875 J=J-1
1880 FORI=1TO3:K=ASC(MID$(P$,I,1))+128:IF
     PEEK(SM+J)=KTHENX=X+1
1885 NEXTI:IFX<2THENPOKESM+J,32:F=0:GOTO1
     873
1890 J=J+1:RETURN
1900 GOSUB1150:A$="0 "+LEFT$(O$(2),2*M)+"
     D":PRINT"{DOWN}";A$
1905 P1=SP+W:P2=P1+LEN(A$)-1:GOSUB1300
1910 IFX=0THENGOSUB1300:GOTO1910
1915 Z=Z-1:IFZ>MTHENRETURN
1920 IFZTHEN1930
1925 FORI=0TOPM-1:POKESM+I,32:NEXTI:J=0:F
     =0:RETURN
1930 RESTORE:READNM,M,DE:LS=PEEK(63)+256*
     PEEK(64)
1935 FORI=1TOM:IFI=ZTHENI=M:GOTO1945
1940 READA$,A$,A$,A$
1945 NEXTI:M=M-1:GOSUB1975
1950 PRINT"{CLR}{3 DOWN}";LS;"DATA";NM;",
     ";M;",";DE
1955 FORI=1TO4:READA$:L=PEEK(63)+256*PEEK
     (64):PRINTL:NEXTI
1960 PRINT"CLR:F=";F;":";
1965 PRINT"T=1:GOTO10{HOME}":POKE198,7:FO
     RI=0TO6:POKE631+I,13:NEXTI:END
1970 FORI=0TOPM-1:POKESM+I,PEEK(828+I):NE
     XTI:J=PEEK(828+PM):RETURN
1975 FORI=0TOPM-1:POKE828+I,PEEK(SM+I):PO
     KESM+I,32:NEXTI:POKE828+PM,J:RETURN
2000 ONZGOTO2005,2070:RETURN
2005 GOSUB1975
2010 IFM=9THENGOSUB1150:PRINT"{DOWN}{RVS}
     MAXIMUM=9 MESSAGES.{OFF}":GOSUB1200:
     RETURN
2020 RESTORE:READNM,M,DE:LS=PEEK(63)+256*
     PEEK(64):L=LS
2025 IFM=0THENL=LS+10:GOTO2045
2030 READA$,A$,A$,A$:L1=PEEK(63)+256*PEEK
     (64)
2035 IFL1-L>40THENL=L+10:GOTO2045
2040 L=L1:GOTO2030
2045 M=M+1:PRINT"{CLR}{3 DOWN}";LS;"DATA"
     ;NM;",";M;",";DE
2047 K=INT(PM/4+.5):FORI=1TO4
2050 PRINTL"DATA"CHR$(34)CHR$(18);:LP=PEE
     K(209)+256*PEEK(210)+PEEK(211)
2051 FORB=0TOK:PRINT" ";:NEXTB:PRINTCHR$(
     34):CR=0:E=0:FORB=0TOK-1
2052 X=PEEK(828+B+(I-1)*K):IF(X<128)ANDCR
     =0THENPOKELP+E,210:CR=1:E=E+1
2053 POKELP+E,X+128*(X>127):IFCRTHENB=K-1
2054 E=E+1:NEXTB:IFCR=0THENPOKELP+E,210
2055 L=L+10:NEXTI:GOTO1960
2070 PRINT"{CLR}{3 DOWN}SAVE";CHR$(34)"CO
     MMUNICATIONS"CHR$(34)":CLR:GOTO10
     {HOME}"
2075 POKE198,1:POKE631,13:END
5000 DATA 3 , 0 , 100
6000 DATA5,"+- MENU CALL ERASE NEW"
6010 DATA"FASTER SLOWER D"
6020 DATA"1 2 3 4 5 6 7 8 9 D"
6030 DATA"RECALL SIGNAL D"
6040 DATA"CHAR. WORD SNT. MSG. D"
6050 DATA"MSG. PRGM. D"
7000 DATA6,4,1,1,1,1,3,3,5,7
7010 DATA"DR. IS{2 SPACES}COLD{2 SPACES}I
     NGEDS1"
7020 DATA"I{3 SPACES}AM{2 SPACES}WHEN
     {3 SPACES}AOTFR3"
7030 DATA"YOU ARE DRINK .ULHCP5"
7040 DATA"MOM EAT WANT{2 SPACES}?MYWKB7"
7050 DATA"DAD NO{2 SPACES}TIME{2 SPACES},
     VJQZX9"
7060 DATA"HOT YES SLEEP ;$02468"
8000 DATA11,4,0,1,1,1,4,5,9,1
8010 DATA"SUN. EARLY TOMORROW{2 SPACES}0"
8020 DATA"MON. LATE{2 SPACES}YESTERDAY 1"
8030 DATA"TUE. NIGHT WEEKEND{3 SPACES}2"
8040 DATA"WED. WEEK{2 SPACES}SPRING
     {4 SPACES}3"
8050 DATA"THU. MONTH SUMMER{4 SPACES}4"
8060 DATA"FRI. YEAR{2 SPACES}WINTER
     {4 SPACES}5"
8070 DATA"SAT. DAY{3 SPACES}FALL
     {6 SPACES}6"
8080 DATA"A.M. TIME{2 SPACES}MORNING
     {3 SPACES}7"
8090 DATA"P.M. NOON{2 SPACES}AFTERNOON 8"
8100 DATA"NOW{2 SPACES}DATE{2 SPACES}EVEN
     ING{3 SPACES}9"
8110 DATA"HOUR NEXT{2 SPACES}MIDNIGHT
     {2 SPACES}:"
9000 DATA9,3,0,1,2,0,5,7,6
9010 DATA" FISH{2 SPACES}LETTUCE SOUP"
```

```
9020 DATA" PORK{2 SPACES}CARROT{2 SPACES}
     BREAD"
9030 DATA" LAMB{2 SPACES}CELERY{2 SPACES}
     BUTTER"
9040 DATA" HT DG TOMATO{2 SPACES}CHEESE"
9050 DATA" HM BG CORN{4 SPACES}SALAD"
9060 DATA" LIVER POTATO{2 SPACES}DRESSG"
9070 DATA" STEAK FRIES{3 SPACES}CHIPS"
9080 DATA" CHKN{2 SPACES}SPINACH MUFFIN"
9090 DATA" TRKY{2 SPACES}PEAS{4 SPACES}GR
     AVY"
```
©

Larry Isaacs

The STATUS Variable

Part 2

In this conclusion of the two-part article on the STATUS variable, we examine possible ST bit values and the resulting indicators.

Before we continue our investigation of the STATUS variable (ST), let's briefly review what we discussed in Part 1.

ST is a *reserved* variable (which means that, like TI for TIME, we can't use ST as a variable in our programs) whose value indicates if anything unusual happened during the last I/O (Input/Output) operation. This status value signals a number of different conditions, using a separate bit "on" for each. However, last month we avoided a direct discussion of bits. Instead, we decided to dissect the ST value by representing the value as a sum of the numbers in the group 1, 2, 4, 8, 16, 32, 64, and -128. If you allow each number to be used only once, only one combination of these numbers can represent the value returned by ST. For example, -118 can be represented by the sum -128+8+2.

Last month we began investigating what the ST variable tells us with respect to the Datassette unit, specifically by examining the End-Of-File (EOF) indicator. The EOF condition is indicated by the presence of 64 in the sum equivalent to the ST value. After a sequence of simple test programs, we discovered that the End-Of-File indicator really meant that the next byte in the file has a value of 0. If perhaps some 0 bytes got mixed into your data, the EOF indicator would sometimes be on when reading the data.

We also found that data is written to the cassette in blocks of 191 bytes, and that a 0 byte is added automatically to the end of the data when the file is closed. Chances are very good, however, that the last block written to the file will contain something less than 191 bytes. One of the tests performed last month showed that if you accidentally read past the end of the data, you could continue receiving data with no apparent ill effects, except that the data isn't valid. We even found that you could continue reading right into an End-Of-Tape block which could follow the data file.

Testing With INPUT#

It is obviously essential, then, to pay attention to the EOF indicator and not allow any 0 bytes to be written to the data file. All the tests in last month's column used the GET# statement to read the data. Before leaving our discussion of the EOF indicator, we should see if there are any problems when using the INPUT# statement. Essentially, we need to find out how the INPUT# statement reacts to the presence of 0 bytes, and what EOF conditions they cause. Run this simple program:

```
100 OPEN 1,1,2,"TEST"
110 PRINT#1,"A";CHR$(0);"B"
120 PRINT#1,"C":CLOSE 1
200 PRINT "REWIND THE CASSETTE."
210 PRINT "PRESS RETURN WHEN READY."
220 INPUT Z$
300 OPEN 1,1,0,"TEST"
310 INPUT#1,Z$:PRINT Z$,LEN(Z$),ST
320 INPUT#1,Z$:PRINT Z$,LEN(Z$),ST
330 CLOSE 1
```

As you can see, lines 100-110 write two lines to a file, lines 200-220 ask you to rewind the cassette, and lines 300-330 read the file. Running this program results in the following display:

```
A   1   64
C   1   64
```

Something a little strange happened. The second string read began with the letter C, which implies that the B was previously read as part of the first string – yet it does not appear as part of the first string. Also, an EOF indicator was given with the first string even though the 0 byte occurred in the middle of the string.

The missing B is fairly easy to explain. The INPUT# statement first reads its data into an input buffer. When the statement stores the data into a string variable, the data must be moved to another area of memory where string characters are stored. In this process, a 0 byte is used to mark the end of the string data to be moved. The presence of the 0 byte in the middle of the data being moved causes the process to be terminated prematurely. Thus the B is left in the input buffer, but isn't stored as part of the string.

As for the EOF, once it is set, it will remain set even though additional bytes are read as part of the input. This also illustrates another case where the EOF indicator does not halt the input of data. This means you must make sure the last data written is properly terminated if you intend to read that data with an INPUT# statement. In our program above, the PRINT# statement causes a carriage return to be written after the C. With this carriage return as the terminator, we get a valid EOF condition when reading the last of the data.

Block Length Errors

A SHORT BLOCK is indicated by the presence of the number 4 in the sum equivalent to the ST value. A LONG BLOCK is indicated by the presence of the number 8 in the sum. These status indicators mean that a block has been read from cassette which contains something other than the expected 191 bytes. This naturally indicates an error: the data read from this block is probably not what we want. These errors occur if something goes wrong while LOADing the cassette, or if something went wrong while SAVEing to the cassette.

Another way to receive these error indicators is to read a program file as if it were a data file. A program file, written by the SAVE command, differs from a data file in that the program is written as a single block. Actually two copies of the program are written (that is, two blocks), with the second copy being used to check for errors in the first block. It is highly unlikely that a program would contain exactly 191 bytes, so you will probably get an error if you try to read a program file as data. For example:

```
100 SAVE "TEST"
200 PRINT "REWIND THE CASSETTE."
210 PRINT "PRESS RETURN WHEN READY."
220 INPUT Z$
300 OPEN 1,1,0,"TEST"
310 GET#1,Z$:PRINT LEN(Z$),ST
320 CLOSE 1
```

The result displayed is:

```
1     4
```

The 4 printed for ST shows that our test program is less than 191 bytes long. You might also note that Z$ still received a byte of data although there was an error when the block was read from the cassette. We can try to force a LONG BLOCK error by adding the following two lines to the example above:

```
110 REM MAKE THE PROGRAM LONGER
120 REM MAKE THE PROGRAM LONGER
```

These lines make the program slightly longer than 191 bytes. Running the program now displays:

```
1     32
```

This wasn't quite what we were expecting. The 32 for ST indicates a CHECKSUM ERROR. With each 191-byte block written, a sum of all the bytes in the block is written along with the data. This sum, called the CHECKSUM, is used to help make sure the data is later read correctly. If at least 191 bytes are received for the block, the CHECKSUM is checked first. Even if the CHECKSUM accidentally matched, we would still get a LONG BLOCK error.

End-Of-Tape Condition

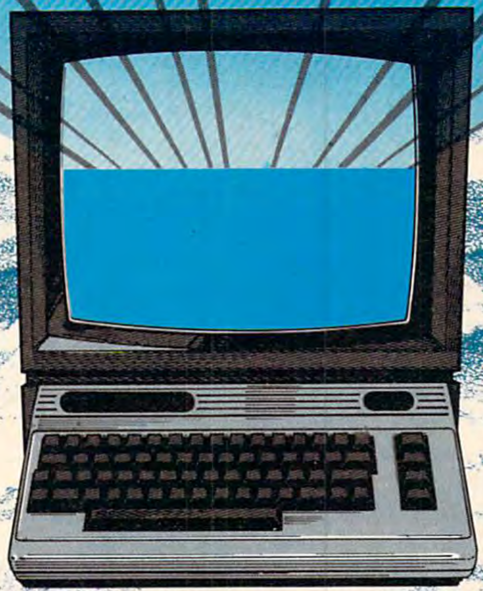
EOT is indicated by a -128 in the sum equivalent to the ST value. The program below provides a simple demonstration:

```
100 OPEN 1,1,2,"TEST"
110 PRINT#1,"A":CLOSE 1
200 PRINT "REWIND THE CASSETTE."
210 PRINT "PRESS RETURN WHEN READY."
220 INPUT Z$
300 OPEN 1,1,0,"NOFILE"
310 PRINT ST:CLOSE 1
```

However, when we RUN this program, the result isn't at all what we expected. We never get a chance to look for a value of -128 in the ST variable because the program quits with the message ?DEVICE NOT PRESENT ERROR IN 300. Later in this article we'll see that when using the serial bus, a -128 for ST indicates that an attempt was made to send data to a device not connected to the computer. When BASIC detects a -128 in the ST value while a file is open for reading, it aborts the program with the "device not present" message without checking to see if the device was the Datassette (in which case the -128 was due to an EOT marker being detected) or a serial bus peripheral such as a disk drive or modem (in which case the -128 indicates a true "device not present" condition). Since the error message throws us out of our program, there's no way to check for the EOT indicator while reading a cassette file in

WORD PROCESSOR FOR COMMODORE 64™

EXPAND YOUR HORIZONS



80 CHARACTER DISPLAY USING ONLY SOFTWARE!

Mirage Concepts' Word Processor does not require costly add-on modules to deliver an 80 column display from your standard Commodore 64™. 200 pages of helpful documentation includes Tutorial, Reference and Index sections. Mirage's Word processor is all you need to expand your Commodore 64™ into an easy-to-use and powerful problem solver with the capabilities of far costlier systems.

MIRAGE'S WORD PROCESSOR FEATURES:

- 40 or 80 Column Screen Display
- 100% Machine Language
- Over 70 Single Keystroke Commands
- Printed Page / Line / Character Counters
- True Word Wrap
- Search, Replace and Block Operations
- 200 Pages of Documentation in Hard Vinyl Binder

Ask About Mirage Concepts'
COMPATIBLE DATABASE MANAGER
Discover How Easy Information Management Can Be!

Mirage Concepts' Word Processor is only... **$99.95**

Available At Quality Computer & Software Dealers Nationwide

MIRAGE CONCEPTS, INC.

2519 W. Shaw Ave., #106 • Fresno, CA 93711 • (209) 227-8369
TELEPHONE ORDERS: (800) 641-1441 • In California (800) 641-1442

COMPUTER MAIL ORDER

TeleVideo

TERMINALS

910	$559.00
912	$689.00
920	$739.00
925	$719.00
950	$929.00
970	CALL

COMPUTERS

800A	$1099.00
802	$2699.00
803	$1949.00
802H	$4695.00
806/20	$4999.00
816/40	$9199.00
1602	$3399.00
1603	CALL

EAGLE

IIE-1	$1369.00
IIE-2	$1649.00
IIE-3	$2399.00
IIE-4	$3199.00
PC-E	$1579.00
PC-1	$2399.00
PC-2	$2799.00
PC-XL	$3599.00
1620	$3599.00
1630	$5499.00
1640	$6499.00
Cyma Software	CALL

IBM

NEC 3550 PRINTER... $1799

PERCOM/TANDON DRIVE

5¼" 320K Floppy	$279.00
5 meg hard w/controller	$1495.00
10 meg hard w/controller	$1795.00
15 meg hard w/controller	$2195.00
20 meg hard w/controller	$2595.00

AMDEK

310A Amber Monitor	$169.00
DXY 100 Plotter	$599.00
Color II	$399.00

AST RESEARCH, INC

SIXPAKPLUS,64K-	
clock,serial,parallel	$299.00
clock,serial,parallel,gameport	$339.00
MEGAPLUAII,64K-	
clock,serial,parallel	$339.00
clock,serial,parallel,gameport	$379.00
clock,2 serial parallel,gameport	$419.00
COMBOPLUSII,64K-	
clock,serial,parallel	$299.00
I/OPLUSII-	
clock,serial,parallel	$145.00
clock,serial,parallel,gameport	$179.00

PROFESSIONAL SOFTWARE

PC Plus Word Processing	$319.00

MICRO PRO

Word Star/Mail Merge	$319.00
InfoStar	$299.00
Spell Star	$159.00
CallStar	$159.00

MICROSTUF

Crosstalk	$139.00

MICROSOFT

Multiplan	$199.00

ASHTON-TATE

D-Base II	$419.00

IUS

EasyWriter II	$209.00
EasySpeller	$129.00
EasyFiler	$129.00

CONTINENTAL SOFTWARE

1st Class Mail/Form Letter	$89.00
The Home Accountant Plus	$109.00

SYNAPSE

File Manager	$119.00

LOTUS

123	$369.00

PFS

	APPLE	IBM
File	$89.00	$99.00
Report	$89.00	$89.00
Graph	$89.00	$99.00
Write	n/a	$99.00

KRAFT

IBM Joystick	$55.00
IBM Paddles	$39.00

QUAD RAM .... CALL

SANYO

MBC-555PC	$795.00
MBC 1100	$1599.00
MBC 1150	$2099.00
MBC 1200	$1999.00
MBC 1250	$2399.00
FDD 3200-320K Drive	$399.00
FDD 6400-64K Drive	$499.00
PR 5500 Printer	$699.00

MODEMS

HAYES

Smart	$219.00
Smart 1200 (1200 Baud)	$519.00
Chronograph	$199.00
Micromodem 100	$309.00
Micromodem II	$279.00
Micromodem II (with term)	$299.00
Smart Com II	$99.00
Smart 1200B	$469.00

NOVATION

J-Cat	$119.00
Cat	$144.00
D-Cat	$159.00
103 Smart Cat	$189.00
Apple Cat II	$279.00
103/212 Smart Cat	$439.00
212 Apple Cat II	$609.00
Apple Cat II 212 Upgrade	$309.00

ANCHOR

Mark I (RS-232)	$79.00
Mark II (Atari)	$79.00
Mark III (T.I. 99)	$109.00
Mark IV (CBM-PET)	$125.00
Mark V (Osborne)	$95.00
Mark VI (IBM-PC)	$179.00
Mark VII (Auto Ans/Auto Dial)	$119.00
Mark VIII	$269.00
TRS-80 Color Computer	$99.00
9 Volt Power Supply	$9.00

ZENITH

ZT1 Terminal	$369.00

MONITORS

AMDEK

300G	$149.00
300A	$159.00
310A	$169.00
Color I	$279.00
Color I plus	$299.00
Color II	$399.00
Color III	$349.00
Color IV	$999.00

USI

Pi 1, 9" G	$99.00
Pi 2, 12" G	$119.00
Pi 3, 12" A	$159.00
Pi 4, 9" A	$139.00
1400 Color	$299.00

ZENITH

ZVM 122A	$119.00
ZVM 123G	$109.00

BMC

12" Green	$85.00
9191 13" Color	$299.00

TAXAN

12 N Green	$129.00
12 A Amber	$139.00

PANASONIC

TR 120 Hi-res. Green	$149.00
CT 160 Dual Mode Color	$279.00

NEC

JB 1260	$119.00
JB 1201	$149.00
JC 1212	$299.00
JC 12-202	$299.00
JC 1203	$469.00

GORILLA

12" Green	$89.00

PRINTERS

EPSON

MX80 FT, MX100, RX80, FX80, FX100	CALL

OKIDATA

82, 83, 84, 92, 93	CALL

STAR

STX 80	$169.00
Gemini 10X	$299.00
Gemini 15	$479.00
Serial Board	$75.00

SMITH CORONA

TP-1	$469.00
TP-2	CALL
Tractor Feed	$119.00

C.ITOH

Gorilla	$209.00
Prowriter 8510P	$379.00
Prowriter 1550P	$689.00
Starwriter F10-40P	$1149.00
Printmaster F10-55P	$1569.00
Tractor Feed	$199.00

DAISYWRITER

2000 Letter Quality	$1149.00
2500...NEW	CALL
Tractor Feed	$109.00

DIABLO

620	$949.00
630	$1769.00

IDS

Call for ALL Configurations on IDS PRISM PRINTERS.

NEC

8023	$399.00
8025	$729.00
3510	$1449.00
3530	$1499.00
3550	$1799.00
7710/7730	$1999.00

BMC

401 Letter Quality	$699.00

BROTHER

Comriter II Printer	$499.00

CABLES & CONNECTIONS

Atari to Parallel	$29.00
Atari to Serial	$29.00
Apple to Parallel	$69.00
Apple to Parallel/Graphics	$99.00
Apple to Serial	$89.00
IBM to Parallel	$35.00
IBM to Serial	$29.00
Parallel to Parallel	$29.00
Serial to Serial	$29.00
Grappler Plus	$129.00
PKASO	$139.00
Atari to Modem Cable	$29.00
CBM 64 to IEEE Board	$79.00
Apple 80 Column Card	$159.00
CBM Pet to Parallel	$99.00
CBM Pet to Serial	$89.00

PAPER SUPPLIES

1or2" Address Labels (Tract. Feed)	$9.95
15" Report Paper (Tract Feed)	$24.95
8½" Blnk Wht Paper (Tract. Feed)	$19.95

HEWLETT PACKARD

HP41CV... $209.00

HP 75........... $749.00

HP 41C	$146.00
HP 10C	$52.00
HP 11C	$69.00
HP 12C	$92.00
HP 15C	$92.00
HP 16C	$92.00

For HP41/41CV

HPIL Module	$99.00
HPIL Cassette or Printer	$359.00
Card Reader	$144.00
Extended Functions Module	$64.00
Time Module	$64.00

EPSON COMPUTERS

QX-10	CALL
HX-20	CALL

TI-40 COMPACT COMPUTER

$209

TIMEX SINCLAIR 1000 $49.95

16K Memory	$44.95
2040 Printer	$99.95
Vu-Calc	$17.95
Mindware Printer	$99.00

SHARP

PC-1500A $169.00

Pocket Computer

PC-1250A........ $89.00

125 Printer/Micro Cassette	$129.00
150 Printer/Plotter/Cassette	$172.00
152 Cassette Recorder	$62.00
155 8K RAM	$94.00
158 8K RAM Battery	$129.00
161 16K RAM	$139.00

WEST

1-800-648-3311

In NV call (702)588-5654 Order Status Number: 588-5654

Dept. 1106, P.O. Box 6689, Stateline NV 89449

No risk, no deposit on C.O.D. orders. Pre-paid orders receive free shipping within the UPS continental United States with no waiting period for certified checks or money orders. Add 3% (minimum $5.00) shipping and handling on all C.O.D. and credit card orders. Larger shipments may require additional charges. NV and PA residents add sales tax. All items subject to availability and price change. We stock manufacturer's and third party software for most all computers on the market. Call today for our catalog.

BASIC. We'll just have to be careful not to attempt to read past the last file on the tape.

While BASIC won't let us check for the EOT marker, it doesn't place the same restrictions on itself. BASIC checks the ST variable for a -128 while LOADing or VERIFYing from cassette to determine if it has read the last program from a tape.

There's one remaining possible condition for the ST variable in cassette operations. A value of 16 in the sum indicates an UNRECOVERABLE READ ERROR. This means that a byte could not be read from the tape. However, as with the LONG BLOCK error, it is unlikely that you will detect this condition since the bad byte will also cause a CHECKSUM error, which is what the ST variable will report. As with the EOT indicator, this is a value which BASIC uses for its own testing during LOADs and VERIFYs.

We said earlier that when you SAVE a program to tape, two copies of the program are actually written out. When the program is read back in, BASIC checks for a value of 16 in the ST variable as the first copy is being read. If too many bad bytes are found, BASIC uses the second copy of the program. It is this feature which makes Commodore Datassettes such reliable data storage devices. People who have used tape storage for other home computers may have trouble believing how rarely the ?LOAD ERROR message is seen on the 64.

In our discussion of the ST variable as it relates to the cassette unit, we found several cases where its actual operation wasn't quite what we were told in the documentation (which didn't say very much on the ST variable anyway). This information will prove useful should you try to write your own program using the cassette for data storage. Also, when information provided by books or manuals isn't sufficient to deal with your specific questions or difficulties, using small test programs is often the best way to find out how something really works.

The Serial Bus

The serial bus is involved when connecting various devices, such as the 1541 disk drive, to the 64. Let's take a look at what the ST variable tells us when used with the serial bus. According to the *Commodore 64 Programmer's Reference Guide*, the status indicators are as follows:

VALUE	MEANING
1	READ TIME OUT
2	WRITE TIME OUT
4	not defined
8	not defined
16	not defined
32	not defined
64	EOI
-128	DEVICE NOT PRESENT

We'll begin with the EOI indicator, which, like EOF for the cassette, indicates when the end of the data has been reached when reading. Again, the important question is whether the EOI indicator accompanies the last byte of data, or comes on when you try to read past the last byte. A simple test program would show that the EOI indicator accompanies the last byte of data, like the EOF does with the cassette. However, a little more investigation shows that the 1541 disk drive, unlike the Datassette, is able to really know when the last byte is sent. This means your data can have all the 0 bytes you want without causing multiple EOI indications.

This also implies that the disk does something different from the cassette with respect to reading past the end of the data. A simple test here shows that the EOI indicator remains on as you continue to read past the end of the data. In addition, the READ TIME OUT indicator comes on (that is, the ST value is 66, the sum of 64+2). Thus, for any given read operation, a read routine is able to determine if the operation occurred normally, read the last byte, or has already passed the end of the data. This is a substantial improvement over what the ST variable tells us when we're working with the tape unit.

The DEVICE NOT PRESENT Indicator

The DEVICE NOT PRESENT condition is shown by the presence of -128 in the sum equivalent to the ST value. This indicator shows that an attempted communication with a particular device was not successful. The error is obvious if the selected device is not connected or not turned on. In addition, if you try to write to an existing file or read a nonexistent file on the 1541, you will get this error message.

It is important to remember that this condition can't occur until an attempt is made to transfer data. A statement like OPEN 1,13 doesn't transfer any data, so the status bit doesn't get a chance to get set. Should you execute such a statement in a program and later execute a PRINT#1 statement, the ST variable would return a value of -128, as you would expect. If you execute a GET#1 or INPUT#1 instead and have no devices connected to the serial bus, you will also get -128 for ST.

However, if at least one device is connected, then the 64 will hang up if you do an INPUT# or GET# from the nonexistent device. The only way to recover is to press the STOP and RESTORE keys simultaneously. The 64 can tell when the desired device isn't there to receive data, but not when the device isn't there to send data. The 64 will patiently wait forever if you let it. This doesn't happen, however, when no serial devices are connected or turned on. The 64 must always out-

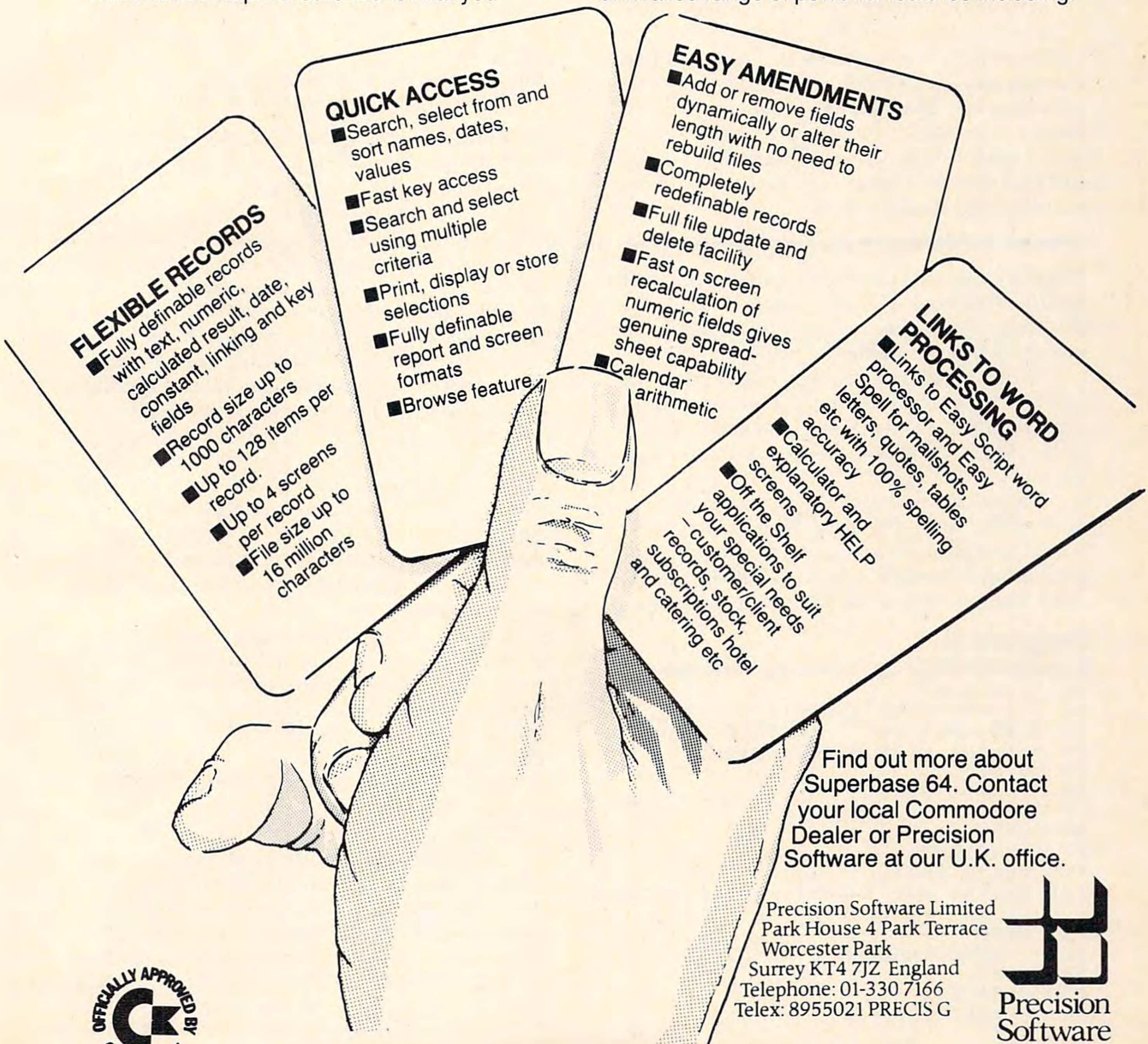
put some command bytes to identify the device with which it wants to communicate. With no devices to receive the command bytes, the DEVICE NOT PRESENT condition is detected before the computer begins waiting to receive data from the nonexistent device.

Since writing to a nonexistent device either hangs up the 64 or gives a DEVICE NOT PRESENT error, it leaves me wondering what situations cause the WRITE TIME OUT. We have already seen the READ TIME OUT, but that was in conjunction with the EOI indicator. Again, there isn't much in the user's manual or reference guide on this topic. I assume that these manuals indicate a data transfer operation was unsuccessful or failed to occur within some time limit.

What we've seen here for status indicators on the 64 may not be typical for other computers. You will find that the EOF indicators will typically come on after the last byte is read, rather than in conjunction with the last byte. So if you are using another computer in addition to the 64, be prepared to find some differences with respect to status indicators for I/O operations. ©

Strengthen your hand with *Superbase* 64

The complete information control system for the Commodore 64. Ideal for any home, business or professional environment where records are kept. Create the format you need and enter your records. If the layout or data field sizes are not quite right, correct them and carry on. Superbase gives you an unrivalled range of powerful features including:



Find out more about Superbase 64. Contact your local Commodore Dealer or Precision Software at our U.K. office.

Precision Software Limited
Park House 4 Park Terrace
Worcester Park
Surrey KT4 7JZ England
Telephone: 01-330 7166
Telex: 8955021 PRECIS G

Precision Software

OFFICIALLY APPROVED BY COMMODORE

Automatic Variables For Atari PILOT

Marvin Roberts

Self-definition allows a variable to be mentioned once, and only once, in a PILOT program. By using a few techniques not discussed in the documentation, it is possible to write a PILOT "story compiler" which: scans a simple text program; asks for values for $VARIABLES; stores these values; and runs the simple text program using the defined values.

The computer can help you create a game using *substitution stories*. PILOT is a good language for this type of interaction, especially for stories which younger children create themselves.

While preparing this type of material, you quickly recognize two things: similar words are repeated in story after story, and each variable occurs twice, once in a question and once again in the story.

To cut down on the typing and to allow a more elaborate response to the more common words and phrases, some sort of general programming technique is required. Program 1 is a short example which illustrates the technique.

Program 1: Example Of PILOT Self-defining Variables

```
10 T:{CLEAR}
20 T:========== example #1 ===
30 A:$A=MARY had a $LITTLE LAMB
40 R:later READ D:$FILENAME,$F
50 MS:$,
60 R:split at $
70 T:
80 DUMP
90 A:=$RIGHT
100 MS:{RIGHT} ,
110 R:skip pad {RIGHT} split at blan
    k
120 DUMP
130 T:please enter $LEFT > \
140 A:$A
150 C:$$LEFT=$A
160 R:self-defined variable
170 T:
180 DUMP
190 T:MARY had a $LITTLE LAMB
200 T:=== end of example #1 ===\
```

The Match String command allows you to split strings into sections. There are several subtle points which need to be noted.

First, strings in the Accept buffer are padded with a leading and a trailing blank, internal blanks are collapsed to only one character, and any lowercase characters are switched to uppercase. You will never get a match on a lowercase string even though it is not rejected by the Match and Match String statements. $VARIABLE names are also restricted to uppercase alphanumerics, but the Accept and Calculate commands generate a warning message, as should the Match statements.

Second, trailing blanks are important. Although the documentation states that an underscore character (_) is used to match on trailing blanks, the actual behavior is more direct. You simply include a blank before the comma (,) or vertical bar (|) which terminates the Match String. This feature works very nicely.

A consequence of the active trailing blanks, however, is that you must use the trailing comma if you have a comment field on the same line as a Match, or if you want to select on a trailing blank. A more subtle consequence of the same logic is that you must avoid comments on an Accept: $VARIABLE line, as the extra blanks would be appended to your entry.

Third, if you use a cursor right (RIGHT) string entry, the leading character positions may be ignored on the match. A similar technique, concatenating an escape (ESC) character and then Matching with a right cursor escape (RIGHT) (ESC), would allow you to trim both the leading and trailing blanks. Take a close look at Program 1 and try various other combinations.

The final technique in Program 1 is the use of an indirect reference to bootstrap a $VARIABLE. This technique, in "TALES Composer" (Program 2), will allow us to read "MARY" (Program 3) as if it were data, pick off and define the variables, and then auto-load and run with the self-defined variables.

Integrating The Programs

The entire body of the text program, MARY (Program 3), is very simple. The only distinction is that any word eligible for substitution must have a leading $ and be in uppercase.

The last three lines of Program 3 are special:

```
100 T:   the end\    (or more\)
110 A:
120 LOAD D:TALES
```

When the user finishes reading the story and presses RETURN, the LOAD statement leaves all strings intact, and the called program runs immediately.

In the composer program, TALES (Program 2), the variables are cleared and the screen is cleared. This takes the housekeeping responsibility away from Program 3.

```
10 VNEW:
20 GR:QUIT
```

A title page is displayed, and the user is asked to enter a $FILENAME, for example, MARY. The program then remains active until a keystroke is detected. This important technique is used again in each of the user modules.

```
240 *FLASH
    (overwrites in inverse video)

280 J(@B764=255):*FLASH
290 A:$FILENAME
```

Now for the magic part. It is possible to read the MARY program as if it were data. The records are placed in $F, which is then examined for $VARIABLES. When an end of file is detected, the TALES program forces the designated text program to load and run. This time, however, the $VARIABLES have been defined, and the substitutions will be made.

```
330 *NEXTOLD READ:$FILENAME,$F
340 J(@B228=136):*ENDOLD
```

One of the stated program objectives was to allow an elaborate response to certain common words and phrases. This is achieved by matching against several keywords and jumping to the appropriate user-developed module.

```
450 M:NOUN,ADJECTIVE,
460 JM:*NOUN,*ADJECTIVE,
470 J:*MODEL
480 *RETURN A:$A
```

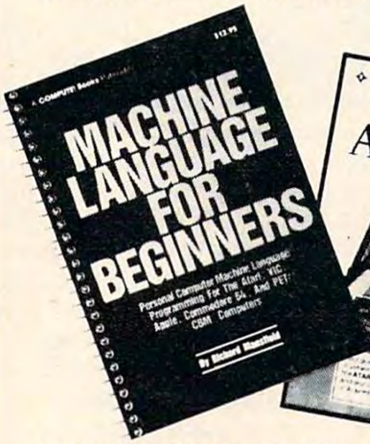
By following the pattern provided by *MODEL and by using *RETURN as an exit when a keystroke is detected, you can add many special modules to the composer program.

Program 2: TALES Composer

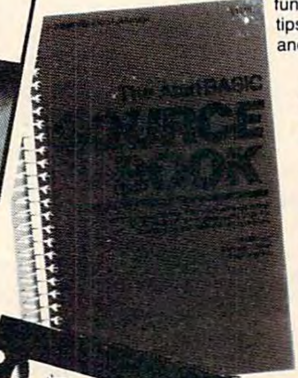
```
10 VNEW:{5 SPACES}[ needed for autor
   eload
20 GR:QUIT
30 C:@B752=1{9 SPACES}[ cursor off
40 T:{CLEAR}{7 SPACES} The Teller of
   Tales
50 T:
60 T: This program will scan other P
   ILOT programs on disk and will as
   k you to provide values for all $
   VARIABLES.
70 T:Your text programs can be very
   simple as they only need T: state
   ments.
80 T:
90 T: 30 T:Mary had a $ADJECTIVE $NO
   UN.
100 T:
110 T:The Teller of Tales will be re
    loaded if the last few lines of
    the text program are:
120 T:
130 T: 100 T:{5 SPACES}more\ or th
    e end\{3 SPACES}\
140 T:
150 T: 110 A:{5 SPACES}( when ready
    )
160 T: 120 LOAD D:TALES
170 T:
180 T:==============================
    ======
190 T:
200 T:
240 *FLASH POS:6,19  [ overprint
250 , T:text old FILENAME > \
260 , POS:6,19
270 , T:text old FILENAME > \
280 J(@B764=255):*FLASH  [ keystroke
290 A:$FILENAME
300 M:D:,{4 SPACES}[ add D: if neede
    d
310 , CN:$FILENAME=D:$FILENAME
320 GR:CLEAR
330 *NEXTOLD READ:$FILENAME,$F
340 J(@B228=136):*ENDOLD
350 *PARSE A:=$F
360 , MS:$,{10 SPACES}[ find variable
    s
370 JN:*NEXTOLD
380 *STRIP A:=$RIGHT
390 , MS:{RIGHT} ,{9 SPACES}[ skip l
    eft pad
400 , C:$F=$RIGHT{4 SPACES}[ split a
    t blank
410 , A:=$LEFT
420 , MS:!.!,!"!)!)!<!>!
430 ,{3 SPACES}CY:$RIGHT=$LEFT $MATC
    H$RIGHT$F
440 JY:*STRIP{8 SPACES}[ find illega
    l
450 , M:NOUN,ADJECTIVE, [ future
460 , JM:*NOUN,*ADJECTIVE, [ modules
470 , J:*MODEL{10 SPACES}[ no match
480 , R:jump back to read user entry
490 *RETURN A:$A
```

THE PROGRAM STORE

YOUR HOME COMPUTER JUST GOT A WHOLE LOT SMARTER WITH THESE SELECTED BOOKS FROM THE PROGRAM STORE'S STOCK OF MORE THAN 500 BOOKS



Everyone, even the most accomplished BASIC programmer needs a good reference manual to get the most out of their computer. ***YOUR ATARI COMPUTER*** (#10629 $17.95) provides a comprehensive, all-in-one guide for any user, beginner or expert. We recommend it over other texts because of its handy alphabetical glossary of statements and functions; its coverage of advanced BASIC and graphics; tips on hardware, peripherals and compatible software; and more!

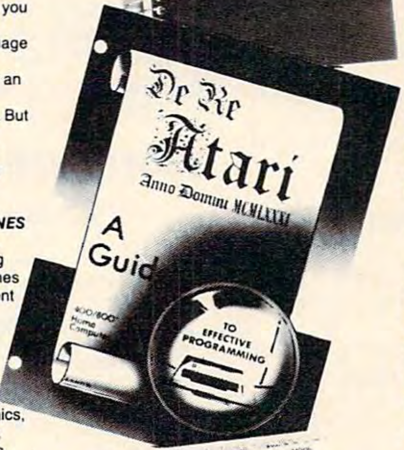


LEARNING MACHINE LANGUAGE

We find that many Atari owners become interested in learning about machine language. ***MACHINE LANGUAGE FOR BEGINNERS*** (#29802 $12.95) will help you make the transition from BASIC to machine language with relative ease. The book assumes you know BASIC and shows you how to write machine language programs. Includes a BASIC program developed from writing machine language to disassembling it.

To start in machine or assembly language, you need an assembler and most people start with the Atari ***ASSEMBLER/EDITOR CARTRIDGE*** (#14308 $56.95). But its operating manual assumes that you already know assembly language. ***THE ATARI ASSEMBLER BOOK*** (#11002 $14.95) by Don and Kurt Inman will guide you through the rudiments in clear, easy steps.

As you become more interested in using assembly language, ***6502 ASSEMBLY LANGUAGE SUBROUTINES*** (#18605 $17.95) is ideal. From Osborne/McGraw-Hill, Lance Leventhal describes general 6502 programming methods and provides code for more than 40 subroutines which you can learn from and use. This book is excellent for those who learn best by examples.



INSIDE THE ATARI

If you are interested in how to get animation, scrolling screens, alternate character sets, player-missile graphics, new sounds, and other features the Atari is capable of, then you will need ***DE RE ATARI*** (#11798 $19.95). This book, ready for a three ring binder, shows you how to exploit the many hardware and operating system features that make the Atari so tremendously versatile. The authors do this in an easy to read text with examples. If you want a more technical style, get Atari's ***TECHNICAL REFERENCE NOTES MANUAL*** (#31318 $19.95). We strongly recommend this book to anyone doing assembly language programming.

The power is inside your Atari but ***MAPPING THE ATARI*** (#29821 $14.95) shows how to use it. Ian Chadwich's extensive research for Compute! Books resulted in this comprehensive resource, memory guide and learning aid. Completely cross-referenced with detailed tutorial commentary on all major memory locations, this book should be part of every machine language programmer's library.

SOURCE OF ALL PROGRAMS

Source listings provide you with the assembly language code of the origirial programmer. These are very useful to learn how a program works (e.g., BASIC), to learn how a professional programs, and to learn how to use the routines in your programs.

THE ATARI BASIC SOURCE BOOK (#19606 $12.95) gives you not only a source listing, but also tells you everything you always wanted to know about the making of a computer language. Even BASIC programmers will enjoy reading about the details of how Atari BASIC works.

Similarly, ***INSIDE ATARI DOS*** (#25973 $19.95) is the comprehensive manual on the disk file manager, commonly known as Atari DOS 2. OS.

ATARI GRAPHIC MAGIC

Almost every Atari owner is interested in programming more graphics. Graphics are one of the best features of the marvelous Atari and we have found three excellent books for almost any programmer.

NEW! Tom Rowley, in ***DESIGNS FROM YOUR MIND*** (#38584 $12.95), introduces shapes, colors, 3-D and screen composition with many sample programs. In the second part of this excellent tutorial he covers the advanced features of player missile graphics, collision registers, display list interrupts, character sets and animation. We highly recommend it.

COMPUTE!'S FIRST BOOK OF ATARI GRAPHICS (#23746 $12.95) has games, tutorials, programs and more collected together for the first time. You'll enjoy the many how-to articles and learn the graphic tricks of many of the leading authors.

ATARI SOUND AND GRAPHICS (#20125 $10.95) published by John Wiley is an excellent self-teaching guide for those learning BASIC. You'll compose and play melodies, draw cartoons, create sound effects and simple games while learning BASIC. A perfect gift for your favorite Atari user!

VISICALC USERS

If you're using ***ATARI VISICALC*** (#15938 $199.95) or thinking about buying it, you'll want to learn more about its expanded uses in ***THE VISICALC BOOK, ATARI EDITION*** (#38360 $14.95). Donald Bell understands that the power of VisiCalc is designing good models so he shows you how to build a model, enter data and explore all aspects! The larger number of practice problems will increase your skills and understanding. Make VisiCalc work for you!

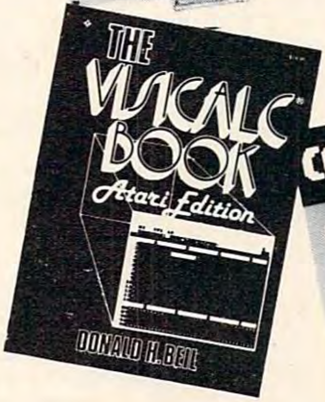
And if you are looking for more examples, including home management, personal finance, general business and more, then you should get ***VISICALC HOME AND OFFICE COMPANION*** (#10719 $15.95). It's one of our best sellers!

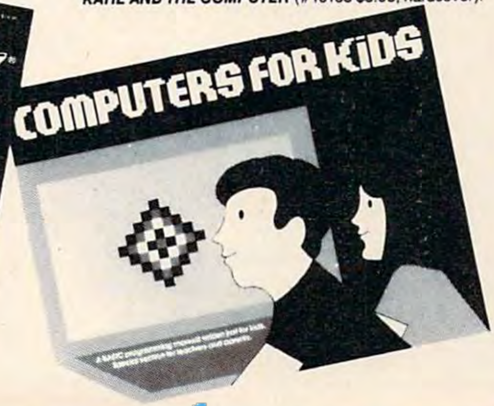
COMPUTER KIDS

For holiday gifts for the kids, you won't want to pass up ***KIDS AND THE ATARI*** (#32050 $19.95) for the 10 to 14 year old who wants to become a computer wizard.

The younger kids will like Creative's new edition of ***COMPUTERS FOR KIDS, ATARI VERSION*** (#10179 $5.95). This BASIC programming manual includes the sure-to-please program, "Scare Mom with an Elephant." Detailed instructions and sketches plus a glossary of statements and commands, lesson plans, and tips for parents all included.

For the preschoolers, we suggest ***COMPUTER PARADE*** by D'Ignazio and richly illustrated by Gilliam ($9.95, hardcover). Katie and her brother arrive in Cybernia just in time to learn how music is made from Colonel Byte. This is the second in the extremely popular series of ***KATIE AND THE COMPUTER*** (#10168 $8.95, hardcover).





THE PROGRAM STORE®

```
500 , C:$$LEFT=$A
510 , R:auto-define the text variabl
    e
520 J:*PARSE
530 *ENDOLD GR:QUIT
540 T:{CLEAR}\
550 LOAD:$FILENAME
560 E:
570 *NOUN R:dummy module
580 *ADJECTIVE R:module
590 *MODEL GR:CLEAR
600 T:{CLEAR}
610 T:this could be a special title
    {7 SPACES}
620 T:enter $LEFT > \
630 *FMODEL GR:PEN YELLOW
640 , GR(#D<90):PEN RED
650 , GR(#D<40):PEN BLUE
660 , C:#D=?\120
670 , SO:#D
680 , GR:GOTO #D-90,?\40-15;TURNTO 4
    5
690 , GR:4(DRAW40;TURN90)
700 J(@B764=255):*FMODEL   [ keystrok
    e
710 SO:0
720 J:*RETURN
```

Program 3: MARY Text Program

```
10 R: D:MARY{3 SPACES}old text PILOT
   program
20 T:{CLEAR}
30 T:$MARY HAD A $LITTLE $LAMB
40 T:
50 T:$MARY had a $LITTLE $LAMB.
60 T:It's $FLEECE was $WHITE as $SNO
   W.
70 T:And everywhere that $MARY went,
80 T:The $LAMB was sure to $GO.
90 T:
100 T:{9 SPACES} the end\
110 A:
120 LOAD D:TALES
130 E:
```

©

THE COMMODORE CHARACTER SET

Dan Carmichael, Assistant Editor

The character chart on the following pages contains all the character information you will need while programming your VIC-20 or your Commodore 64. Keep it by your computer for handy reference.

Here's a handy and essential chart, including information on decimal, hexadecimal, BASIC, 6502, ASCII, and screen POKE codes. Please note the following conventions:

BASIC – lists the BASIC (language) keyword tokens.

6502 – contains the numeric representation of the 6502 instruction set. Zero page operations are listed with ZZ. Immediate operations are listed with II. Others are listed with NNNN.

ASCII – contains the Commodore ASCII control and character codes.

Screen Display – lists the character codes for POKEing to the screen. Set 1 and Set 2 correspond to uppercase and lowercase, respectively. Both sets cannot be displayed on the screen at the same time. Switch to Set 2 by simultaneously holding down the SHIFT and COMMODORE keys or POKEing 53272,23. To return to uppercase again, press the SHIFT/COMMODORE keys or POKE 53272,21.

Screen display characters may also be displayed in reverse character mode by adding 128 to the values that are listed.

Standard ASCII – gives values for standard ASCII codes. These codes can be used for controlling printers or when standard ASCII values are needed, as in telecommunications applications.

STANDARD ASCII

1	start of heading
2	start of text
3	end of text
4	end of transmission
5	enquiry
6	acknowledge
7	bell
8	backspace
9	horizontal tabulation
10	line feed
11	vertical tabulation
12	form feed
13	carriage return
14	shift out
15	shift in
16	data link escape
17	device control 1 (X-on)
18	device control 2
19	device control 3 (X-off)
20	device control 4
21	negative acknowledge
22	synchronous idle
23	end of transmission block
24	cancel
25	end of medium
26	substitute
27	escape
28	file separator
29	group separator
30	record separator
31	unit operator

Alspa Computer, Inc.

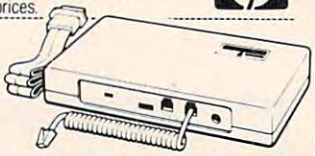
Price-performance leader. Includes Z80A, 8" ds/dd drives, 3 serial + 1 parallel port, winchester port, networking. Prices start below $1500. DEALER / OEM inquiries invited.

SPECIALS on INTREGATED CIRCUITS

6502	7.45	10/ 6.95	50/ 6.55	100/ 6.15
6502A/6512A	8.40	10/ 7.95	50/ 7.35	100/ 6.90
6520 PIA	5.15	10/ 4.90	50/ 4.45	100/ 4.15
6522 VIA	6.45	10/ 6.10	50/ 5.75	100/ 5.45
6532	7.90	10/ 7.40	50/ 7.00	100/ 6.60
2114-L200		2.45	25/ 2.30	100/ 2.15
2716 EPROM		4.90	5/ 4.50	10/ 4.00
2532 EPROM		6.90	5/ 6.75	10/ 6.45
6116 2KX8 CMOS RAM		6.90	5/ 6.75	10/ 6.45
4116 RAM				8 for 14
4164 RAM				6.50
Zero Insertion Force 24 pin Socket (Scanbe)				2.00

Hewlett Packard

Write or call for prices.



Anchor Automation Signalman Modems

FREE SOURCE MEMBERSHIP WITH SIGNALMAN

All Signalman Modems are Direct Connect, and include cables to connect to your computer and to the telephone. Signalman Modems provide the best price-performance values.

Dealer and OEM inquiries invited

Volksmodem	(79)	55
RS232 or Atari Cable for Volksmodem		9
Mark I RS232	(99)	59
Mark VI for IBM Personal Computer	(279)	179
Mark VII Auto Dial/Auto Answer	(159)	89
Mark XII Smart Model 1200/300	(399)	295

DC HAYES Smartmodem	219
DC Hayes Smartmodem 1200/300	529

PROM QUEEN for VIC	170
Apple Emulator for Commodore 64	Call
STAT Statistics Package for C64	95
Solid Oak 2 Level Stand for C64 or VIC	29
C64/VIC Switch (networking)	125
BACKUP V1.0 tape copier for C64 or VIC	20
CARDBOARD/6 Motherboard - VIC	64
CARDBOARD/5 Motherboard - C64	56
CARDPRINT Printer Interface - C64/VIC	64
CARDBOARD/3s Motherboard - VIC	32
CARDCO C64/VIC Calculator Keypad	32
CARDRAM/16 RAM Expansion - VIC	64
Complete CARDCO Line in stock	
CIE and VIE IEEE Interfaces in stock	
BASM Compiler/Assembler for C64	89

APPLE—FRANKLIN ITEMS

KRAFT Apple Joystick	40
16K RAM Card for Apple	59
Solid Oak 2 Level Stand for Apple	29
Serial Card for Apple	99
MPC RAM/80 column card for IIE (AP/TXT)	139
Z80 Softcard and CP/M (Microsoft)	235
RANA Elite I with Controller	389
Parallel Printer Interface/Cable	79
Apple Dumpling (Microtek) Printer Interface	115
Apple Dumpling with 16K Buffer	160
Grappler + Interface	129
Kraft Products for Apple in stock	
DC Hayes Micromodem II	299
PFS: File	100
PFS: Report	100
Videx 80 Column Card	209
Hayden Software for Apple 20% OFF	
Apple Blue Book	19

commodore

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

PETSCAN I $245 base price

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

COMPACK/STCP $115

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

VE-2 IEEE to Parallel Interface 110

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

VIDEO ENHANCER for Commodore 64 69
Realize video quality equal or better than composite monitor using standard color TV.

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

GENISIS Computer Corp

VIController (for C64 as well)	50
combine with BSR modules for home or business control	
COMSENSE Remote Sensing Adapter for C64 or VIC	35
COM VOICE Synthesizer for C64 or VIC	139
includes software for test to speech, pitch, etc.	
COM CLOCK Real Time Clock with battery backup	45

VIC 20 Products and Software in stock

Thorn EMI Software		UMI Software	
ABACUS Software		HES Software	
16K RAM for VIC	64	Vanilla Pilot	27

VICTORY Software for VIC and C64

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

PAPER CLIP Word Processor	109
ORACLE Data Base from Batteries Included	125
Commodore 64 Programmers Reference Guide	16
MicroChess for C64—8 levels of play	19
Compute!'s First Book of PET/CBM	11
C64 or VIC SWITCH	125
POWER ROM Utilities for PET/CBM	78
WordPro 3+/64	69
WordPro 4+ - 8032, disk, printer	295
SPELLMASTER spelling checker for WordPro	
VISICALC for PET, ATARI, or Apple	189
PET-TRAX PET to Epson Graphics Software	40
SM-KIT enhanced PET/CBM ROM Utilities	40
Programmers Toolkit - PET ROM Utilities	35
EASY CALC for C64	65
PET Spacemaker II ROM Switch	36
COPYWRITER Word Processor for C64	69
2 Meter PET to IEEE or IEEE to IEEE Cable	40
Dust Cover for PET, CBM, 4040, or 8050	8
CmC Interfaces (ADA1800, ADA1450, SADI in stock)	
Programming the PET/CBM (Compute!) — R. West	20
Compute! First Book of VIC	11
HES MODEM with Software	65
HES Software and Hardware in stock	
UMI products in stock	

GRIDRUNNER (HES) VIC or C64			29
COCO (HES) Tutorial for C64			39
Public Domain Disks for C64			65
Agressor (HES)			29
HES Turtle Graphics	VIC 29	C64	49
HES Writer	VIC 30	C64	34
Grand Master Chess (UMI) C64 or VIC			27
Renaissance (UMI)	VIC 29	C64	24
Synthesound (HES)			44
HES MON	VIC or C64		29
6502 Professional Developement System (HES)			24
Robot Panic 29	Pirate's Peril 29	Retro Ball	29

DISK SPECIALS

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

We stock VERBATIM DISKS

Write for Dealer and OEM prices.

Sentinal 5" ss/dd	10/ 1.90	50/ 1.85	100/ 1.80
Sentinal 5" ds/dd	10/ 2.55	50/ 2.50	100/ 2.45

We stock Dysan disks

Wabash 5" ss/sd	10/ 1.60	50/ 1.55	100/ 1.45
Wabash 5" ss/dd	10/ 1.90	50/ 1.85	100/ 1.75
Wabash 8" ss/sd	10/ 2.00	50/ 1.95	100/ 1.85

We stock MAXELL DISKS

Write for dealer and OEM prices.

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

CASSETTE TAPES—AGFA PE-611 PREMIUM

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

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

Zenith ZVM-121 Green Phosphor Monitor	95
Zenith new color and monochrome monitors in stock	
MultiPlan—IBM or Apple	185
Quadboard for IBM available	
Peachtext 5000 Software Package	219
PFS Software for IBM and Apple in stock	
VOTRAX Personal Speech System	280
VOICE BOX Speech Synthesizer (Apple or Atari)	
CompuServe Subscription (5 hours free)	35
Dynax (Brother) DX-15 Daisy Wheel Printer	469
Itoh Prowriter Parallel Printer	379
Panasonic 1090 Printer with Correspondence Mode	319
USI CompuMOD 4 R F Modulator	39
Daisywriter 2000 with 48K buffer	1050
Many printers available (Gemini-Star, Brother, OKI, etc.)	
We Stock AMDEK Monitors	
Amdek DXY-100 Plotter	590
A P Products	15% OFF
Watanabe Intelligent Plotter 990 6-pen	1290
BROOKS 6 Outlet Surge Suppressor/Noise Filter	54
We stock Electrohome Monitors	
Synertek SYM-1 Microcomputer	189

ALL BOOK and SOFTWARE PRICES DISCOUNTED

Panasonic 12" Monitor (20 MHz) with audio	139
Panasonic CT-160 Dual Mode Color Monitor	259

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

ZENITH data systems

HERO 1 Robot (factory assembled)	2145
Z29 Terminal (DEC and ADM compatible)	680
ZT-10 Intel. Terminal with Serial Port	340
Z100 16-bit/8-bit Systems in stock	CALL
We stock entire Zenith line.	

ATARI SPECIALS

WE STOCK ENTIRE LINE—write for prices.
Software for Atari in stock.

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

A B Computers

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

OAK STAND-C64, VIC, Apple, Atari — 29

Beautiful natural solid oak two-level stand. Rests on table above computer. Holds disk drives/cassette deck, as well as your monitor/TV.

KMMM Pascal IV.1 for PET/CBM/C64 — $95

Full-featured Pascal for Commodore Computers.

Now suitable for all Advanced Placement Courses.
Includes all features of full Jensen-Wirth Pascal except WIDTH, SETS, RECORD VARIANCE (plus has STRING extension). Includes machine language Pascal Source Editor (with Syntax checking), machine language P-Code Compiler, P-Code to machine language Translator or optimized object code. Run-time package, User Manual, and sample programs.
Requires 32K — Please specify configuration.

EARL for PET (disk file based) — $65

Editor, Assembler, Relocater, Linker

Generates relocatable object code using MOS Technology mnemonics. Disk file input (can edit files larger than memory).

Paper Clip (Batteries Included)

Extremely comprehensive word processor for Commodore and Commodore 64 computers. Has features of WordPro, plus advanced functions like horizontal scroll, column move, column arithmetic, column sort, and comprehensive printer support.

Delphi's ORACLE (Batteries Included)

Comprehensive Data Base, Report Writer, Mail Label system allowing large record size (over 8000 characters) with the number of records in a file limited only by disk capacity (7.5 MB on 9090 drive). Fast machine language routines, including full multilevel sorts.

Super BusCard (Batteries Included)

- full buffered IEEE488 bus for speed.
- cartridge extension slot.
- parallel printer port.
- DOS "wedge" commands included.
- machine language monitor included.
- room for 24K ROM BASIC 4.0 (optional).

RAM/ROM for PET/CBM

4K or 8K bytes of soft ROM optional battery backup.

Use RAM/ROM as a software development tool to store data or machine code beyond the normal BASIC range, or to load a ROM image to avoid ROM socket conflicts. Possible applications include machine language sort (such as SUPER-SORT), universal wedge, Extramon, etc.

RAM/ROM — 4K $75 — RAM/ROM — 8K 90
Battery Backup Option — 20

PORTMAKER DUAL RS232 SERIAL PORT — $63

Two ports with full bipolar RS232 buffering. Baud rates from 300 to 4800. For PET/CBM, AIM, SYM.

SuperGraphics 2.0

NEW Version with TURTLE GRAPHICS

SuperGraphics, by John Fluharty, provides a 4K machine language extension which adds 35 commands to Commodore BASIC to allow fast and easy plotting and manipulation of graphics and shapes on the PET/CBM video display.

SOUND commands allow you to initiate notes or songs from BASIC, and then play them in the background mode without interfering with your BASIC program.

Additionally, seven new TURTLE commands open up a whole new dimension in graphics.

Specify machine model (and size), ROM type.

SuperGraphics in ROM ($A000 or $9000) — $45

Volume discounts available for schools.

FLEX-FILE for PET/CBM Computers — NEW VERSION 2 now for C64

FLEX-FILE is a set of flexible, friendly programs to allow you to set up and maintain a data base. Includes versatile Report Writer and Mail Label routines, and documentation for programmers to use Data Base routines as part of other programs.

RANDOM ACCESS DATA BASE

Record size limit is 256 characters. The number of records per disk is limited only by record size and free space on the disk. File maintenance lets you step forward or backward through a file, add, delete, or change a record, go to a numbered record, or find a record by specified field (or partial field). Field lengths may vary to allow maximum information packing. Both subtotals and sorting may be nested up to 5 fields deep. Any field may be specified as a key. Sequential file input and output, as well as file output in WordPro and PaperMate format is supported. Record size, fields per record, and order of fields may be changed easily.

MAILING LABELS

Typical mail records may be packed 3000 per disk on 8050 (1400 in 4040). Labels may be printed any number wide, and may begin in any column position. There is no limit on the number or order of fields on a label, and complete record selection via type code or field condition is supported.

REPORT WRITER

Flexible printing format, including field placement, decimal justification and rounding. Define any column as a series of math or trig functions performed on other columns, and pass results such as running total from row to row. Totals, nested subtotals, and averages supported. Complete record selection, including field within range, pattern match, and logical functions can be specified.

FLEX-FILE 2 by Michael Riley — $110

CBM64, PET/CBM/VIC—32K, Disk. Specify configuration.

SCREEN MAKER (cgrs microtech) — $149

80 Column Adapter for Commodore 64

Expand your computer for business applications. Provides 80 column × 24 line display in a 2K video RAM. Linking software provided.

Copy-Writer Word Processor — $69

Full-featured professional word processor with over 800 lines of text per memory load on C64. Has features not available in many word processors such as double column printing, built in graphic capability, shorthand notations, and ability to support all printer codes.

SPECIAL COMBINATION PACKAGE — $200

Includes SCREEN MAKER AND Copy Writer for C64

PROGRAM YOUR OWN EPROMS — $75

Branding Iron EPROM Programmer for PET/CBM software for all ROM versions. Includes all hardware and software to program or copy 2716 and 2532 EPROMs.

DISK I.C.U. — $40

Intensive Care Unit by L.C. Cargile

COMPLETE DISK RECOVERY SYSTEM FOR CBM DRIVES

Edit disk blocks with ease; duplicate disks, skipping over bad blocks; un-scratch scratched files; check and correct scrambled files; recover improperly closed files.

Includes complete diagnostic facilities, extensive treatment of relative files, optional output to IEEE488 printer, and comprehensive user manual (an excellent tutorial on disk operation and theory).

Furnished on copy-protected disk with manual.
Backup disk available, $10 additional.

CBM Software

SUBSORT for PET/CBM — 35
—excellent general purpose machine language sort routine.
COMAL Package for CBM — 25
—includes software on disk, and Comal Handbook
BASIC INTERPRETER for CBM 8096 — $95
PEDISK II Systems from cgrs Microtech available.
FILEX IBM 3741/2 Data Exchange Software available.
JINSAM Data Base Management System for CBM.

CASH MANAGEMENT SYSTEM for CBM	$45
Petspeed BASIC Compiler	120
Integer BASIC Compiler	120
BPI Accounting Modules	280
UCSD Pascal (without board)	135
Wordcraft 80 or 8096	265

FORTH for PET — $50

BY L.C. Cargile and Michael Riley

Features include:

- full FIG FORTH model.
- all FORTH 79 STANDARD extensions.
- structured 6502 Assembler with nested decision macros.
- full screen editing (just as in BASIC)
- auto repeat key.
- sample programs.
- standard size screens (16 lines by 64 characters).
- 150 screens per diskette on 4040, 480 screens on 8050.
- ability to read and write BASIC sequential files.
- introductory manual and reference manual.

For Commodore 64, or any 16K/32K PET/CBM with ROM 3 or 4, and CBM disk drive. Please specify configuration when ordering.

Metacompiler for FORTH — $30
Simple metacompiler for creating compacted object code which can be run independently (without FORTH system).

PageMate 60 COMMAND WORD PROCESSOR

by Michael Riley

Paper-Mate is a full-featured word processor for Commodore computers. Page-Mate incorporates 60 commands to give you full screen editing with graphics for all 16K or 32K machines (including 8032), all printers, and disk or tape drives. Many additional features are available (including most capabilities of WordPro 3).

Page-Mate functions with all Commodore machines with at least 16K, with any printer, and either cassette or disk.

To order Page-Mate, please specify machine and ROM type.

Page-Mate (disk or tape) for PET, CBM, VIC, C64 — $40

SM-KIT for PET/CBM — $40

Enhanced ROM based utilities for BASIC 4. Includes both programming aids and disk handling commands.

STAT for PET/CBM and C64 — 95

Comprehensive Statistical Analysis Routines

Includes complete disk-based handling routines. Features: normal, T, Chi-Square, F, binomial, Poisson, and exponential distributions; one way ANOVA; two way ANOVA; contingency analysis; linear regression; data transformation; histogram, curve, and scatter plotting; and random sample data generation.

Specify machine type and drive when ordering.

Commodore 64

Hunter-Killer - Commodore 64	15
- authentic naval warfare game (complete with sonar)	
BASM Compiler and Assembler	85
WordPro 3+/64	75
Vanilla PILOT with Turtle Graphics	27
- also includes sound, Toolkit, joystick support	
Commodore 64 Programmer Reference Guide	16
EARLY GAMES for Young Children	25
PETSPEED Compiler C64	120
CALC RESULT ADVANCED Spread Sheet Package	135
1000 Miles (Mille Bornes Game)	9
MicroChess	19
Adventure (disk)	9
PILOT 64	45
LOGO 64	45
Easy Calc 64	65
MAE Assembler - C64	85
Synthy-64 music and sound synthesizer	26
Tiny BASIC Compiler	17
Assembly Language Tutorial - C64/VIC	27
ScreenGraphics-64 adds BASIC Graphics	22
Abacus Software in stock	

VICTORY SOFTWARE ®

All Victory Software in stock.

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

A B Computers

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

DEC	HEX	BASIC	6502	ASCII	SCREEN DISPLAY SET 1	SET 2
0	0	END LINE	BRK		@	
1	1		ORA ($ZZ,X)		A	a
2	2				B	b
3	3				C	c
4	4				D	d
5	5		ORA $ZZ	white	E	e
6	6		ASL $ZZ		F	f
7	7				G	g
8	8		PHP	disables shft/Cmdr	H	h
9	9		ORA #$II	enables shft/Cmdr	I	i
10	A		ASL		J	j
11	B				K	k
12	C				L	l
13	D		ORA $NNNN	return	M	m
14	E		ASL $NNNN	lowercase	N	n
15	F				O	o
16	10		BPL $NNNN		P	p
17	11		ORA ($ZZ),Y	crsr down	Q	q
18	12			reverse on	R	r
19	13			clr/home	S	s
20	14			inst/del	T	t
21	15		ORA $ZZ,X		U	u
22	16		ASL $ZZ,X		V	v
23	17				W	w
24	18		CLC		X	x
25	19		ORA $NNNN,Y		Y	y
26	1A				Z	z
27	1B				[	
28	1C			red	£	
29	1D		ORA $NNNN,X	crsr right	]	
30	1E		ASL $NNNN,X	green	↑	
31	1F			blue	←	
32	20	SPACE	JSR $NNNN	space	space	
33	21	!	AND ($ZZ,X)	!	!	
34	22	"		"	"	
35	23	#		#	#	
36	24	$	BIT $ZZ	$	$	
37	25	%	AND $ZZ	%	%	
38	26	&	ROL $ZZ	&	&	
39	27	'		'	'	
40	28	(	PLP	(	(	
41	29	)	AND #$II	)	)	
42	2A	*	ROL	*	*	
43	2B	+		+	+	
44	2C	,	BIT $NNNN	,	,	
45	2D	-	AND $NNNN	-	-	
46	2E	.	ROL $NNNN	.	.	
47	2F	/		/	/	
48	30	0	BMI $NN	0	0	
49	31	1	AND ($ZZ),Y	1	1	
50	32	2		2	2	
51	33	3		3	3	
52	34	4		4	4	
53	35	5	AND $ZZ,X	5	5	
54	36	6	ROL $ZZ,X	6	6	
55	37	7		7	7	
56	38	8	SEC	8	8	
57	39	9	AND $NNNN,Y	9	9	
58	3A	:		:	:	
59	3B	;		;	;	
60	3C	<		<	<	
61	3D	=	AND $NNNN,X	=	=	
62	3E	>	ROL $NNNN,X	>	>	
63	3F	?		?	?	

DEC	HEX	BASIC	6502	ASCII	SCREEN DISPLAY SET 1	SET 2
64	40	@	RTI	@		
65	41	A	EOR ($ZZ,X)	A		A
66	42	B		B		B
67	43	C		C		C
68	44	D		D		D
69	45	E	EOR $ZZ	E		E
70	46	F	LSR $ZZ	F		F
71	47	G		G		G
72	48	H	PHA	H		H
73	49	I	EOR #$II	I		I
74	4A	J	LSR	J		J
75	4B	K		K		K
76	4C	L	JMP $NNNN	L		L
77	4D	M	EOR $NNNN	M		M
78	4E	N	LSR $NNNN	N		N
79	4F	O		O		O
80	50	P	BVC $NN	P		P
81	51	Q	EOR ($ZZ),Y	Q		Q
82	52	R		R		R
83	53	S		S		S
84	54	T		T		T
85	55	U	EOR $ZZ,X	U		U
86	56	V	LSR $ZZ,X	V		V
87	57	W		W		W
88	58	X	CLI	X		X
89	59	Y	EOR $ZZ,Y	Y		Y
90	5A	Z		Z		Z
91	5B			[		
92	5C			£		
93	5D		EOR $NNNN,X	]		
94	5E		LSR $NNNN,X	↑	π	
95	5F			←		
96	60		RTS		space	
97	61		ADC ($ZZ),X			
98	62					
99	63					
100	64					
101	65		ADC $ZZ			
102	66		ROR $ZZ			
103	67					
104	68		PLA			
105	69		ADC #$II			
106	6A		ROR			
107	6B					
108	6C		JMP ($NNNN)			
109	6D		ADC $NNNN			
110	6E		ROR $NNNN			
111	6F					
112	70		BVS $NN			
113	71		ADC ($ZZ),Y			
114	72					
115	73					
116	74					
117	75		ADC $ZZ,X			
118	76		ROR $ZZ,X			
119	77					
120	78		SEI			
121	79		ADC $NNNN,Y			
122	7A					
123	7B					
124	7C					
125	7D		ADC $NNNN,X			
126	7E		ROR $NNNN,X	π		
127	7F					

ADDITIONAL NOTES: **SCREEN CODES:** CODES 128-255 are reversed images of CODES 0-127.

Now There's An Alternative Floppy Disk Drive For Commodore Computers!

Super Disk is a Commodore compatible disk drive designed to interface to the various Commodore computers such as the PET, VIC-20, and the Commodore 64. The disk drive is compatible to the model 4040, 2031, 1540, and 1541 disk drives and recognizes programs generated on any of these disk drives. The capacities are comparable to those found on the Commodore drives, and Super Disk recognizes the full instruction set of the Commodore drives. Super Disk offers 8K RAM area within the disk unit, and a serial and an IEEE bus interface. The following technical specifications apply to Super Disk.

Disk Parameters: 5.25 inch single-sided, double density diskette media; number of tracks — 35; number of sectors per track — 17 to 21 (variable); sector size — 256 bytes; number of sectors per disk — 683.

Capacities: User sectors per disk — 667; disk capacity — 174,848; data capacity — 168,656; relative data capacity — 167,132; program capacity — 168,656; directory capacity — 144 Filenames.

Electrical Interface: Power requirements — input voltage — 110/220 volts +/– 10%; frequency — 50 or 60 Hertz; power — 30 watts. Computer interface — Commodore type SERIAL bus or Commodore type IEEE bus.

System Features: 16K ROM based operating system; 8K RAM area; 6511Q Microprocessor; Power on self test diagnostics.

SINGLE DRIVE

DUAL DRIVE

Features

- Single or dual drives
- Reliable; proven drive mechanism
- Comes with both serial and IEEE bus ports
- Serial configuration compatible with Commodore 1541
- IEEE configuration compatible with Commodore 2031 and 4040
- Disk formats in less than 20 seconds
- Power on self-test diagnostics
- 8K RAM, 16K operating system

MSD Also Manufactures IEEE Interfaces and Cables For Commodore 64™ and VIC 20™

Designed For The Serious User

A Premium Drive At A Slightly Higher Price

MICRO SYSTEMS DEVELOPMENT, INC.

(214) 241-3743

11105 Shady Trail • Suite 104 • Dallas, Texas 75229

Call Toll Free

1-800-527-5285

For Purchase Information

Unbelievable

Commodore 64

Avalon Hill Game Company

180-701 B-1 Nuclear Bomber (C) $12
180-702 Midway Campaign (C) $12
180-703 North Atlantic Convoy Raider (C) $12
180-704 Nukewar (C) $12
180-706 Planet Miners (C) $12
180-712 Computer Stocks & Bonds (C) . $15
180-719 Andromeda Conquest (C) $14
181-721 Computer Football Strategy (C) $12
181-732 Telengard (C) $16

Broderbund

David's Midnight Magic $23
Choplifter (CT) $34
Serpentine (CT) $27
Sea Fox (CT) $27

Datamost

Roundabout $20
Bilestoad $20
Mating Zone $20

Don't Ask Computer Software

Wordrace .... (D) $ 23, (C) $ 23

EPYX/Automated Simulations

14E-036 Jump Man (D) $27

Human Engineered Software (HES)

6502 Professional Dev. System (C) ... $ 23
Retro Ball (CRT) .... $ 27
Hesmon (CRT) .... $ 27
Turtle Graphics II (CRT) .... $ 45
Heswriter 64 (CRT) .... $ 35
Gridrunner (CRT) .... $ 27

Infocom

Zork I, II, III .... Ea. $ 27
Deadline (D) .... $ 34
Starcross (D) .... $ 27
Witness .... $ 34

Sierra On-Line

Frogger (D) .... $ 25
Crossfire .... $ 15
Jaw Breaker .... $ 20
Threshold (CRT) .... $ 27
Sammy Light Foot (CRT) .... $ 25

Sirius Software

Blade of Blackpoole (D) .... $ 27
Type Attack (CRT) .... $ 27
Repton (D) .... $ 27
Critical Mass (D) .... $ 27
Snake Byte (D) .... $ 23
Bandits .... $ 27
Squish 'em (CRT) .... $ 23
Final Orbit (CRT) .... $ 23

Spinnaker

Snooper Troops #1 (D) .... $ 30
Facemaker (D) .... $ 23
Kindercomp (D) .... $ 20
Hey Diddle Diddle .... $ 20
In Search of the Most Amazing Thing . $ 27
Fraction Fever (CRT) .... $ 20
Alphabet Zoo (CRT) .... $ 20
Delta Drawing (CRT) .... $ 20

Synapse Software

Ft. Apocalypse .... (D) $ 23, (C) $ 23
Drelbs .... (D) $ 23, (C) $ 23
Survivor .... (D) $ 23, (C) $ 23
Pharoah's Curse .... (D) $ 23, (C) $ 23

United Microwave Industries

92E-302 Renaissance (C) .... $ 20
92E-331 Motor Mania (C) .... $ 20

Commodore 64 .... *$219*
VIC 20 .... *$ 99*
1525 Printer *$229*
1530 Datasette .... *$ 64*
1541 Disk Drive *$249*
1600 Modem .... *$ 89*
1701 Commodore Monitor *$289*
VIC 1311 Joystick .... *$ 8*
VIC 1312 Game Paddles .... *$ 16*

VIC 1210 3K Memory Expander $34
VIC 1110 8K Memory Expander $52
VIC 1111 16K Memory Expander $89
VIC 1011 RS 232 Terminal Interface ... $43
VIC 1211 Super Expander $59
VT 106A Recreation Pack .... $ 45
VT 107A Home Calculation Pack ..... $ 45
VIC 1600 Vicmodem .... $ 89
VM Programmer's Reference Guide .. $ 14
Commodore Programmer's Ref. Guide $ 18

Educational

Books

Kids and the Vic .... $ 18
Programmer's Reference Guide (Vic) . $ 14
Programmer's Reference Guide (64) .. $ 18

Language Arts

Super Hangman (C) .... $ 14
Simon/Hess (C) .... $ 13
Concentration (C) .... $ 13
Home Babysitting .... $ 23

Math

Sky Math (C) .... $ 12
Space Division .... $ 12
Bingo Speed Math (CT) .... $ 23
Number Crunch (CT) .... $ 23
Number Chaser .... $ 17
Number Gulper .... $ 17

Music

Vic Music Composer (CT) .... $ 42
HES Synthesound (CT) .... $ 49

Reading and Language Arts

Finger Spelling (D, C) .... $ 12
My Spelling Easel (Ages 3-10) .... $ 26

Programming Techniques

Intro to Basic Prog. I .... $ 22
Intro to Basic Prog. II .... $ 22
Programmer's Aid Cart .... $ 22
Turtle Graphics/Hess (CT) .... $ 29
Gorteck & the Microchip .... $ 23

Social Studies/Science

Visible Solar System .... $ 23
Reaganomics (CT) .... $ 27

Commodore Software

Avenger .... $ 23
Superslot .... $ 23
Super Alien .... $ 23
Jupiter Lander .... $ 23
Draw Poker .... $ 23
Midnight Drive .... $ 23
Radar Rat Race .... $ 23
Raid on Fort Knox .... $ 23
Sargon II Chess .... $ 29
Cosmic Cruncher .... $ 23
Gorf .... $ 29
Omega Race .... $ 29
Sea Wolf .... $ 23
Adventureland .... $ 29
Pirate Cove .... $ 29
Mission Impossible .... $ 29
The Count .... $ 29
Voodoo Castle .... $ 29
The Sky is Falling .... $ 23
Mole Attack .... $ 23
Bingo Speed Math .... $ 23
Home Babysitter .... $ 23
Visible Solar System .... $ 23
Personal Finance .... $ 29
Quick Brown Fox .... $ 65

Call Toll Free

1-800-634-6766

Order Line Only — Ask for Operator CO58

1095 E. Twain (702) 796-0296
Las Vegas, Nevada 89109

Information & Order Inquiries
(702) 369-5523
We accept VISA and MasterCard
Mon. — Fri. 8 A.M. to 6 P.M.
Sat. 9 A.M. to 6 P.M.
Dealers' Inquiries Invited

Vic 20

Avalon Hill

Tank Arcade .... $ 12
Nuke War .... $ 12

Automated Simulations

Rescue at Rigel (C) $20
Ricochet (C) $15
Monster Maze (CT) $27
Sword of Fargoal $27

Broderbund

Martian Raider $15
Multisound Synthesizer $15
Shark Trap $15
Sky Blazer (CT) $27
Sea Fox (CT) $27
A.E. (CT) $27

Creative Software

Black Hole (CRT) .... $ 36
Trashman (CRT) .... $ 36
Astroblitz (CRT) .... $ 36
City Bomber & Minefield (CRT) ...... $ 20
Apple Panic (CRT) .... $ 36
Serpentine (CRT) .... $ 36
Choplifter (CRT) .... $ 36
Terraguard (CRT) .... $ 36
Household Finance .... $ 36

HES Software

HES Mon (CT) .... $ 29
HES Writer (CT) .... $ 29
Synthesound Music Synthesizer (CT) . $ 49
Turtle Graphics (CT) .... $ 29
VIC Forth (CT) .... $ 45
Victrek (C) .... $ 15
Predator (CT) .... $ 27

Sirius

Type Attack $27
Snake Byte $27

Thorn EMI

River Rescue (CT) .... $ 29
Mutant Herd (CT) .... $ 29

Tronix

Galactic Blitz (C) $17
Swarm (C) $20
Sidewinder (C) $20
Gold Fever (CT) $27
Deadly Skies (CT) $27

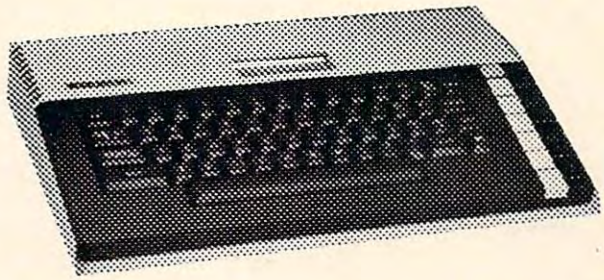
United Microware

Spiders of Mars (CT) .... $ 34
Meteor Run (CT) .... $ 34
Amok .... (C) $ 15 (CT) $ 27
Alien Blitz .... (C) $ 17 (CT) $ 27
Skymath (C) .... $ 12
Space Division (C) .... $ 12
Super Hangman (C) .... $ 14
The Alien (C) .... $ 17
3D Maze (C) .... $ 12
Kosmic Kamikaze (C) .... $ 17
Sub Chase (C) .... $ 15
Renaissance .... $ 20
Cloud Burst (CT) .... $ 27
Satellites & Meteorites (CT) .... $ 34
Outworld (CT) .... $ 34
Wordcraft .... $ 65

COMPUTER OUTLET

Great Savings!

600XL . . . CALL
800XL . . . CALL
1400XL . . CALL
1450XLD . CALL



Atari Inc.

1010 Recorder	$ 72
1050 Disk Drive	$339
1027 Printer	Call
1025 Printer	$409
830 Modem	$145
850 Interface	$159
Entertainer System	$ 64
482 Educator	$110
483 Programmer	$ 52
Communicator II	$209
Atari Accounting	$169
CX4104 Mailing List	$ 19
CXL 4007 Music Composer	$ 42
Programming 2 & 3	Ea. $ 22
Conversational Languages	$ 42
CX4018 Pilot	$ 55
CX405 Pilot	$ 92
CXL4003 Assembler Editor	$ 42
CXB126 Microsoft Basic II	$ 62
CXL4022 Pac-Man	$ 30
CXL4020 Centipede	$ 30
CXL4006 Super Breakout	$ 26
CXL4008 Space Invaders	$ 26
CXL4009 Computer Chess	$ 26
CXL4011 Star Raiders	$ 30
CXL4012 Missile Command	$ 26
CXL4013 Asteroids	$ 26
The Bookkeeper	$102
Home Filing Manager	$ 65
Atari Speed Reading	$ 54
Home Manager Kit	$ 55
Family Finance	$ 36
Time Wise	$ 23
Galaxian	$ 30
Defender	$ 30
Paint	$ 30
Qix	$ 30
Dig Dug	$ 30
ET Phone Ho	$ 34
Atari Writer	$ 65
Donkey Kong	$ 36
Donkey Kong Jr.	$ 38
Atari Logo	$ 72
Mickey in the Great Outdoors/D	$ 34
My First Alphabet/D	$ 25
Peter Pan's Daring Escape	$ 34

Business & Utilities

Visicalc	$169
Computari's Financial Wizard	$ 45
Color Accountant	$ 65
Mail Merge	$ 20
Data Perfect	$ 95
Letter Perfect	$105
Bank Street Writer	$ 45
Text Wizard	$ 55
File Manager 800 +	$ 65
Datasm 652.0	$ 59
Syn Assembler	$ 34
K-Dos	$ 59
Lisp Interpreter	$ 79
Basic Compiler	$ 55
Datalink	$ 27
Atari World	$ 39
Color Print	$ 27
Graphics Generator	$ 17
Micropainter	$ 23
Graphics Master	$ 17
P.M.P. Property Management	$149

Modems

Hayes Smartmodem 300 Baud	$209
Hayes Smarmodem 1200	$499
Signalman Modem II	$ 79
Atari 830 Modem	$155

Educational

Pre-School

Sammy the Sea Serpent (C) $ 13 (D)	$ 19
Pre-School I.Q. Builder . . . (C) $ 13 (D)	$ 24
My First Alphabet (D)	$ 26
Alien Counter/Face Flash (D, C)	$ 26
Humpty Dumpty/Jack & Jill (C)	$ 25
Facemaker (D)	$ 23
Juggles Rainbow (D, C)	$ 22
Juggles House	$ 22
Alphabet Zoo (CT)	$ 20

Music

Music Composer (CT)	$ 42
Jerry White Music Lessons (C)	$ 20
Magic Melody Box	$ 14

Social Studies and Geography

Globemaster (D)	$ 27
States and Capitals (C)	$ 12
European Countries and Capitals (C)	$ 12

Math

Monkey Up A Tree(D,C)	$19
Video Math Flash Cards(D,C)	$13
Algicalc(D,C)	$19
Compumath-Fractions	(C)$23(D)$29
Compumath-Decimals	(C)$23(D)$29
Golf Classic/Compubar (Angles)	(D,C)$26
Cash Register	(C)$13(D)$19
Big Math Attack	(C)$17(D)$22
Computation Concentration . (C) $13 (D)	$15

Reading and Language Arts

Wordmaker (D, C)	$ 19
Spelling Genie (D, C)	$ 19

Compuread . . . (C) $ 17 (D)	$ 23
Memory Builder/ Concentration . . . (C) $ 13 (D)	$ 19
Let's Spell (C)	$ 13
Do-It-Yourself Spelling (C)	$ 16
S.A.T. College Board Prep (C)	$ 89
Vocabulary Builder I . . . (C) $ 13 (D)	$ 19
Fishing for Homonyms (C)	$ 13
Hey Diddle Diddle (D)	$ 20
Snooper Troops 1 & 2 (D)	Ea. $ 30
Story Machine (D)	$ 23
Crossword Magic (D)	$ 34
Delta Drawing (CT)	$ 23

Programming Techniques

Pilot (Cons. of Educator) . . (C) $ 59 (D)	$ 99
Invitation to Prog. #2 (C)	$ 22
Invitation to Prog. #3 (C)	$ 22
Tricky Tutorials — Ed Software	
TT#1 Display Lists (C, D)	$ 17
TT#2 Horiz/Vert. Scrolling (C, D)	$ 17
TT#3 Page Flipping (C, D)	$ 17
TT#4 Basics of Animation (C, D)	$ 17
TT#5 Player Missile Graphics (C, D)	$ 24
TT#6 Sound & Music (C,D)	$ 24
TT#7 Disk Utilities (D)	$ 24
TT#8 Character Graphics	$ 19
TT#9 GTIA	$ 19
TT#10 Sound Effects	$ 19
Page 6	$ 20
The Next Step	$ 27

Stocking Stuffers

WICO "BOSS" Joystick	**$ 15.00**
WICO Redball Joystick	**$ 24.00**
Elephant Disks s/s	**$ 18.00**
Verbatim Disks s/d	**$ 24.00**
Disk Savers (Plastic Sleeves) Multi Colored 1 Doz.	**$ 4.50**
Disk Mailers	**$ 3.50**
Flip 'n File Diskette Holder w/Lock (holds 25)	**$ 18.50**
Flip 'n File Diskette Box (holds 50)	**$ 21.00**
Library Carrying Case (holds 10)	**$ 2.50**

Super Savers

Gorilla Banana Printer	**$ 209.00**
Percom Single Density Drive	**$ 389.00**
Percom Double Density Drive	**$ 515.00**
Amdek Color I Monitor	**$ 299.00**

Call Toll Free

1-800-634-6766

Order Line Only — Ask for Operator CO58
Information & Order Inquiries (702) 369-5523

We accept VISA and MasterCard
Dealers' Inquiries Invited

New Hit List

In Search of the Most Amazing Thing	$ 27
Witness	$ 34
Cosmic Balance II	$ 27
Temple of Apshai	$ 27
Raster Blaster	$ 20
Deadline	$ 34
Richochet	$ 15
Wiz & Princess	$ 22
Ali Baba and the Forty Thieves	$ 22
Canyon Climber	$ 20
Crush, Crumble & Chomp	$ 20
Zork I, II & III	Ea. $ 27
Frogger	$ 23
Choplifter	$ 23
Curse of Ra	$ 15
Ulysses and the Golden Fleece	$ 23
Tigers in the Snow	$ 27
David's Midnight Magic (D)	$ 23
Sky Blazer (D)	$ 22
Serpentine (D)	$ 22
Sea Fox (D)	$ 20
Sands of Egypt (D)	$ 27
Pool 400 (CT)	$ 27
Gorf . . . (D) $ 27 (CT)	$ 30
Wizard of Wor . . . (D) $ 27 (CT)	$ 30
Cyborg (D)	$ 23
Gold Rush (D)	$ 23
Bandits (D)	$ 23
Way Out (D)	$ 27
Fast Eddy (CT)	$ 24
World War (CT)	$ 24
The Cosmic Balance (D)	$ 27
Chess (D)	$ 45
Checker (D)	$ 34
Raptillian (D,C)	$ 23
Submarine Commander (CT)	$ 34
Jumbo Jet Pilot (CT)	$ 34
Soccer (CT)	$ 34
Starcross (D)	$ 27
Zaxxon (D,C)	$ 27
Miner 2049er (CT)	$ 34
Twerps (D)	$ 23
Flip Out (D)	$ 20
The Birth of the Phoenix	$ 16
Protector II . . . (D) $ 23 (CT)	$ 29
Baseball (CT)	$ 34
Preppie II (D,C)	$ 23
Arcade Machine (D)	$ 39
Cap n' Cosmos (D)	$ 29
Spy's Demise (D)	$ 15
Repton (D)	$ 27
Critical Mass (D)	$ 27
Millionaire (D)	$ 55
Poker Sam (D)	$ 17
Jump Man (D)	$ 27
Hellfire Warrior (D,C)	$ 27
Trion (D)	$ 27
Adventure in Time (D)	$ 20
Wavy Navy (D)	$ 23
Final Orbit (CT)	$ 23
Pharoah's Curse (D,CT)	$ 23

Printers

NEC 8023A	$ 439
Epson FX80	$ 599
Epson FX100FT	$ 799
Okidata ML82A	$ 419
Okidata ML83A	$ 649
Okidata ML84P	$ 979
Okidata 92	$ 489
Okidata 93	$ 869
Citoh 8510 Prowriter	$ 365
Mannesmann Tally MT160L	$ 599
Qume Sprint 11 +	$1349
Gemini 10X	$ 299
Gemini 15	$ 479

Monitors

Leading Edge Gorilla Hi-Res 12"	$ 85
Amdek Color I	$ 299
USI Amber	$ 149
NEC Green Screen	$ 169

1095 E. Twain, Las Vegas, Nevada 89109 (702) 796-0296 • Mon. — Fri. 8 A.M. to 6 P.M. — Sat. 9 A.M. to 5 P.M.

ORDERING INFORMATION AND TERMS: For Fast Delivery send cashier checks, money orders or direct bank wire transfers. Personal and company checks allow 3 weeks to clear. Charges for C.O.D. orders are $3.00 minimum or 1% for orders over $300. School purchase orders welcomed. Prices reflect a cash discount only and are subject to change without notice. Please enclose your phone number with any orders. SHIPPING — Software: $2.50 minimum. SHIPPING — Hardware: (Please call) SHIPPING — Foreign Orders: APO & FPO orders $10 minimum and 15% of all orders over $100. Nevada residents add 5 3/4% sales tax. All goods are new and include factory warranty. Due to our low prices, all sales are final. All returns must be accompanied by a return authorization number. Call 702-369-5523 to obtain one before returning goods for replacement.

DEC	HEX	BASIC	6502	ASCII
128	80	END		
129	81	FOR	STA ($ZZ,X)	orange
130	82	NEXT		
131	83	DATA		
132	84	INPUT#	STY $ZZ	
133	85	INPUT	STA $ZZ	F1
134	86	DIM	STX $ZZ	F3
135	87	READ		F5
136	88	LET	DEY	F7
137	89	GOTO		F2
138	8A	RUN	TXA	F4
139	8B	IF		F6
140	8C	RESTORE	STY $NNNN	F8
141	8D	GOSUB	STA $NNNN	shift/carriage return
142	8E	RETURN	STX $NNNN	uppercase
143	8F	REM		
144	90	STOP	BCC $NN	black
145	91	ON	STA ($ZZ),Y	crsr up
146	92	WAIT		reverse off
147	93	LOAD		clr/home
148	94	SAVE	STY $ZZ,X	inst/del
149	95	VERIFY	STA $ZZ,X	brown
150	96	DEF	STX $ZZ,Y	lt. red
151	97	POKE		gray 1
152	98	PRINT#	TYA	gray 2
153	99	PRINT	STA $NNNN,Y	lt. green
154	9A	CONT	TXS	lt. blue
155	9B	LIST		gray 3
156	9C	CLR		purple
157	9D	CMD	STA $NNNN,X	crsr left
158	9E	SYS		yellow
159	9F	OPEN		cyan
160	A0	CLOSE	LDY #$II	space
161	A1	GET	LDA ($ZZ,X)	▌
162	A2	NEW	LDX #$II	▄
163	A3	TAB(		▔
164	A4	TO	LDY $ZZ	▁
165	A5	FN	LDA $ZZ	▏
166	A6	SPC(	LDX $ZZ	▒
167	A7	THEN		▕
168	A8	NOT	TAY	▄
169	A9	STEP	LDA #$II	◤
170	AA	+	TAX	▕
171	AB	-		├
172	AC	*	LDY $NNNN	▗
173	AD	/	LDA $NNNN	└
174	AE	↑	LDX $NNNN	┐
175	AF	AND		▂
176	B0	OR	BCS $NN	┌
177	B1	>	LDA ($ZZ),Y	┴
178	B2	=		┬
179	B3	<		┤
180	B4	SGN	LDY $ZZ,X	▎
181	B5	INT	LDA $ZZ,X	▍
182	B6	ABS	LDX $ZZ,Y	▐
183	B7	USR		▔
184	B8	FRE	CLV	▀
185	B9	POS	LDA $NNNN,Y	▃
186	BA	SQR	TSX	┘
187	BB	RND		▖
188	BC	LOG	LDY $NNNN,X	▝
189	BD	EXP	LDA $NNNN,X	┘
190	BE	COS	LDX $NNNN,Y	▘
191	BF	SIN		▚
192	C0	TAN	CPY #$II	
193	C1	ATN	CMP (ZZ,X)	
194	C2	PEEK		
195	C3	LEN		
196	C4	STR$	CPY $ZZ	
197	C5	VAL	CMP $ZZ	
198	C6	ASC	DEC $ZZ	
199	C7	CHR$		
200	C8	LEFT$	INY	
201	C9	RIGHT$	CMP #$II	
202	CA	MID$	DEX	
203	CB	GO		
204	CC		CPY $NNNN	
205	CD		CMP $NNNN	
206	CE		DEC $NNNN	
207	CF			
208	D0		BNE $NN	
209	D1		CMP ($ZZ),Y	
210	D2			
211	D3			
212	D4			
213	D5		CMP $ZZ,X	
214	D6		DEC $ZZ,X	
215	D7			
216	D8		CLD	
217	D9		CMP $NNNN,Y	
218	DA			
219	DB			
220	DC			
221	DD		CMP $NNNN,X	
222	DE		DEC $NNNN,X	
223	DF			
224	E0		CPX #$II	
225	E1		SBC ($ZZ,X)	
226	E2			
227	E3			
228	E4		CPX $ZZ	
229	E5		SBC $ZZ	
230	E6		INC $ZZ	
231	E7			
232	E8		INX	
233	E9		SBC #$II	
234	EA		NOP	
235	EB			
236	EC		CPX $NNNN	
237	ED		SBC $NNNN	
238	EE		INC $NNNN	
239	EF			
240	F0		BEQ $NN	
241	F1		SBC ($ZZ),Y	
242	F2			
243	F3			
244	F4			
245	F5		SBC $ZZ,X	
246	F6		INC $ZZ,X	
247	F7			
248	F8		SED	
249	F9		SBC $NNNN,Y	
250	FA			
251	FB			
252	FC			
253	FD		SBC $NNNN,X	
254	FE		INC $NNNN,X	
255	FF			

ASCII CODES: CODES 192-223 same as 96-127 CODES 224-254 same as 160-190 CODES 255 same as 126

YOUR PROBLEM IS SOLVED!

Now you can rely on *PACE* for ONE STOP shopping for all your Micro Computer needs. We have picked out the BEST 2000 Programs, Books and Accessories—covering all the major brands and put them into one friendly store. And, this is backed up by THOUSANDS of additional items we stock in our central warehouse, ready for overnight shipping to our stores. Magazines? You bet! We carry almost 60 different Micro Magazines on our racks! Plan to visit us soon. Can't visit? Then you can order from our gigantic product list—just write for your *FREE*, personal copy *TODAY!*

pace

For: VIC-20

NEW **THE VIC-20™ CONNECTION** By James W. Coffron. Learn the simple techniques for putting your VIC to work controlling external devices: lights, alarms, etc. Softcover. 7" X 9" format. 260 Pages.
4795-000128 $ 7.95

NEW **YOUR FIRST VIC-20™ PROGRAM** By Rodney Zaks. Write your first BASIC program in one hour! First time user guide, ages 8 to 88! Softcover. 7" X 11". 150 Pages.
4795-000129 $ 9.95

NEW **COMPUTERS FOR KIDS: VIC-20™ Edition** By Sally Greenwood Larsen. Tailored for even the youngest child, it has full instructions on writing programs for graphics, calculations and more! Special section of tips for teachers/parents. Softcover. 11" X 8½" format. 70-plus pages long.
4198-000042 $ 5.95

NEW **HOW TO USE THE COMMORODRE® VIC-20™** By Jerry Willis and Deborrah Willis. The beginner's text. Includes a special Problem-Solving section for any difficulties you might encounter. Softcover 5½" X 8½". 124 Pages.
4250-000134 $ 3.95

VIC-20™ PROGRAMMER'S REFERENCE GUIDE Commodore. Complete BASIC vocabulary guide, machine language guide, tips, tricks, short-cuts and general operations. Belongs beside your VIC! Softcover. Spiralbound. 290 Pages.
4760-021948 $16.95

COMPUTE!s FIRST BOOK OF VIC™ A compilation of articles from Compute! Magazine.
4105-000007 Soft. 212 Pages $12.95

For: Atari

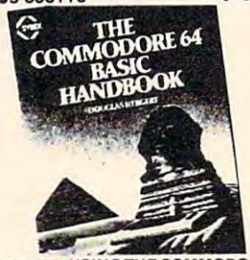
NEW **THE EASY GUIDE TO YOUR ATARI® 400/800™** By Joseph Kascmer. Jargon-free operating manual. Shows the easy way to use your Atari. . .word processing, math, budgeting, filing and more! The easiest way to start using your computer! Softcover. 5½" X 8¼" format. 160 Pages.
4795-000125 $ 7.95

NEW **BASIC EXERCISES FOR THE ATARI®** By J. P. Lamoitier. Learn true style and subtleties of Atari BASIC through actual practice. Graduated exercises in math, business, operations research, games and statistics. Each exercise contains a statement and analysis of the problem, a solution with flowchart, and a program with detailed explanation. Softcover. 7" X 11". 251 Pages.
4795-000101 $12.95

For: Commodore 64

NEW **HOW TO USE THE COMMODORE® 64™** By Jerry Willis & Deborrah Willis. All you'll need to know to successfully operate your C-64. Good beginning text and includes a Problem-Solving section for any difficulties you might encounter. Softcover. 5½" X 8½" format. 124 Pages.
4250-000133 $ 3.95

NEW **THE COMMODORE® 64™ BASIC HANDBOOK** By Douglas Hergert. Need to check or clarify a BASIC keyword or function? Do it quickly with this dictionary-style handbook. Clear descriptions and examples of the 64's vocabulary. Fully explained in easy to understand English. Softcover. 5½" X 8¼" format. 170 Pages.
4795-000116 $ 9.95



NEW **USING THE COMMODORE® 64™ IN THE HOME** By William Behrendt & hank Liubrach. Beginner-expert, kids-adults, can now have 20 programs that include home management and games that would cost hundreds of dollars if bought separately. Helps teach basic programming techniques. Sample runs included. Softcover.
4690-940072 $10.95

NEW **COMMODORE® 64™ COMPUTING** By Ian Sinclair. Both an intro how-to and a reference manual. Discusses set up, operations and covers special features such as Sprite graphics, color commands and programming for sound. Illustrated with BASIC examples. Softcover.
4690-152306 $12.95

NEW **SPRITE GRAPHICS FOR THE COMMODORE® 64™** By Sally Larsen. For beginners, an intro to Sprite graphics on the Commodore 63—the documentation that is lacking when they buy their 64. Covers how to use Sprites for the practical tools they are and includes 25 ready-to-run programs. Softcover.
4690-838136 $15.95

SAVE UP TO 20% ON THESE POPULAR GAMES

GAMES/VIC-20

TURMOIL Sirius. Can you survive the onslaught of aliens attacking from both sides? Fast paced arcade action. Joystick. 5K.
4770-022005 Cart. (Reg. 39.95) **$31.95**

DEMON ATTACK Imagic. Wave after wave of attacking aliens. Are you quick enough? 1 or 2 player action.
5431-005200 Cart. (Reg. 39.95) **$31.95**

SHAMUS HESware. Not for the high blood pressure crowd! Fight your way through over 60 rooms, each full of numerous different robot henchmen. Then you must face the SHADOW!
4428-000307 Cart. (Reg. 39.95) **$31.95**

GRIDRUNNER HESware. Attacks from above, the left, right and below you! Can you move your fighter quickly enough to avoid destruction and eliminate the enemy hoardes? "Centipede' type game with some twists! Joystick.
4428-000312 Cart. (Reg. 29.95) **$23.95**

RAT HOTEL Creative Software. Six floors of action as you, the RAT, must avoid the maintenance man. Joystick.
4200-000124 Cart. (Reg. 39.95) **$31.95**

CHOPLIFTER Creative Software. Fly your helicopter to rescue your people! Fast action arcade type game. Joystick required. 5K.
4200-000119 Cart. (Reg. 39.95) **$31.95**

PIPES Creative Software. Combines games with education. You must connect all the homes in town to the cental water main. You must select the right pipe at the right price, be cost effective and watch your inventory! Joystick.
4200-000128 Cart. (Reg. 39.95) **$31.95**

PRICES & AVAILABILITY SUBJECT TO CHANGE

GAMES/COMMODORE 64

FAST EDDIE Sirius. Fast paced arcade style game. Collect treasures as you climb ladders and jump barrels. For C-64. Disk.
4770-030006 (Reg. 34.95) *SALE* **$27.99**

CENTROPODS CommData. Fast action "Centepede" type game. Cassette Only.
4103-001132 (Reg. 19.95) *SALE* **$15.99**

JUMPMAN Epyx. Popular arcade game now available for the C-64! Two versions.
4085-005873 Cass. (Reg. 39.95) **$31.95**
4085-005874 Disk (Reg. 39.95) **$31.95**

SUSPENDED Infocom. Adventure-style game. Your mind is in control of six robots, each with a different view of the world! Disk.
5432-245421 (Reg. 49.95) *SALE* **$39.99**

TOTL TEXT Totl Software. Powerful word processor with twice features at half the cost. Machine language, menu driven, compatible with 80 col. adapters, full documentation.
5841-006401 Cass. (Reg. 40.00) **$31.95**
5841-046401 Disk (Reg. 40.00) **$31.95**

GAMES/ATARI 400/800

CASTLE WOLFENSTEIN Muse. You must escape from the Nazis in this popular adventure. Disk. Requires 32K.
5574-392422 (Reg. 29.95) **$23.99**

ZAXXON™ Datasoft. Atari version of the fast paced arcade action game. 16K Req.
5206-012003 Cass. (Reg. 39.95) **$31.95**
5206-012004 Disk (Reg. 39.95) **$31.95**

PACE MICRO SOFTWARE CENTER
345 East Irving Park Road. Wood Dale. Il 60191
PHONE: (312) 595-3860

MODEMS. . .

	RS232 MODEMS	
5042-000001	Mark 1 RS232 (Cable Included) 300 Baud	$99.00
5042-000007	Mark VII RS232 (Cable Incl) 300 Baud (Auto Answer/Dial)	$159.00
5042-000008	Mark VIII RS232 (Cable Incl) 300/1200 Baud (Auto Ans/Dial)	$399.00
5615-490190	Cat RS232 (LESS Cable) 300 Baud	$189.00
5615-490268	D-Cat RS232 (LESS Cable) 300 Baud	$199.00
5615-490533	J-Cat RS232 (LESS Cable) 300 Baud	$149.00
5615-490501	212 Auto Cat RS232 (LESS Cable) 300/1200 Baud	$695.00
5615-490418	Smart Cat 103 RS232 (LESS Cable) 110/300 Baud	$249.00
5615-490521	Smart Cat 103/212 (LESS Cable) 300/1200 Baud	$595.00
5417-701942	Smart Modem 300 RS232 (LESS Cable)	$289.00
5417-701752	Smart Modem 1200 RS232 (LESS Cable)	$699.00
	IBM® MODEMS	
5417-701990	Smart Modem 1200B W/Software (NO Cable Needed)	$599.00
5042-000006	Mark VI (NO Cable Needed)	$279.00
	APPLE® MODEMS	
5417-701387	MicroModem II (NO Cable Needed)	$379.00
5417-701431	Micro Modem II W/Software	$409.00
5615-490402	Apple Cat II (Plugs In)	$389.00
5615-490502	Apple Cat 212 (Plugs In)	$725.00
5615-490412	Apple Cat Upgrade (II to 212)	$389.00
	COMMODORE® MODEMS	
4428-000404	HESMODEM 1 (NO Cable Needed) Includes Software	$69.95
5042-000003	Mark II for all PET/CBM (NO Cables Needed)	$169.00
	TEXAS INSTRUMENTS® MODEMS	
5042-000003	Mark III (REQUIRES Expansion Card & Interface)	$139.00
	ATARI® MODEMS	
5042-000002	Mark II (REQUIRES Expansion Interface)	$99.00
	OSBORNE® MODEMS	
5042-000005	Mark V (NO Cable Required)	$129.00

INTERFACES REQUIRED USING RS232C MODEMS:

	FOR ATARI®:	
5083-000850	Atari 850 Interface	$219.95
5615-490411	EIA-RS232 Cable	$19.00
	FOR APPLE® (Both Are REQUIRED)	
5590-642647	Apple Cal. Comp. Sys. Serial Interface	$150.00
5590-639287	RS232 Data Cable, 8 Ft.	$24.95
	FOR COMMODORE® (Both Are REQUIRED)	
4100-001011	RS232C Adapter	$49.95
5615-490411	EIA RS232 Cable	$19.00
	FOR TEXAS INSTRUMENTS® (Requires TWO Below)	
5590-881250	TI99/4 RS232 Card, Parallel	$174.95
OR 5590-881200	TI99/4 Expansion Chassi	$249.95
AND 5615-490411	EIA RS232C Cable	$19.00
	FOR IBM® (Both Are REQUIRED)	
5590-828433	IBM I/O Serial Card	$279.00
5615-490411	EIA RS232 Cable	$19.00

BOOKS ON MODEMS

MICROCOMPUTER DATACOMMUNICATION SYSTEMS Derfler. A guidebook to the operations of modems, terminals, electronic bulletin boards and information utilities for users of TRS-80®, Apple II®, Heath-89®, Commodore® Pet® and other general systems. Softcover. 6¾" X 9¼" format. 131 pages.
4690-580712 $12.95

HELLO CENTRAL! Bruce Kallick. Simple, menu-controlled terminal program and any compatible modem to communicate with information networks or other computers. Built-in text editor. Easel-back binder, 7" X 9" X 1". For 48K Apple II® w/Applesoft™ in ROM. One disk drive and modem required.
4760-026081 $99.00

TRS-80® COMMUNICATION SYSTEMS Frank J. Derfler, Jr. A guide to the operation of TRS-80 micropmputers as communication devices. Softcover. 6¾" X 9" format. 159 Pages.
4690-931220 $12.95

By the time that you read this ad, our first **PACE** store in the western suburbs of Chicago, at **345 East Irving Park Road, Wood Dale,** will be open, soon to be followed by many others. Until a **PACE** store opens in your city, you may order direct from the **PACE** central warehouse with confidence, where we currently stock over 6,000 microcomputer products.

COUPON Please Send Me:

QTY	NUMBER	DESCRIPTION	COST	TOTAL

Illinois Residents Please Add 6% Sales Tax.
Foreign Orders. (All outside Continental US). Add 10% Shipping (Minimum $4.00)
Catalogs Shipped Postage Paid

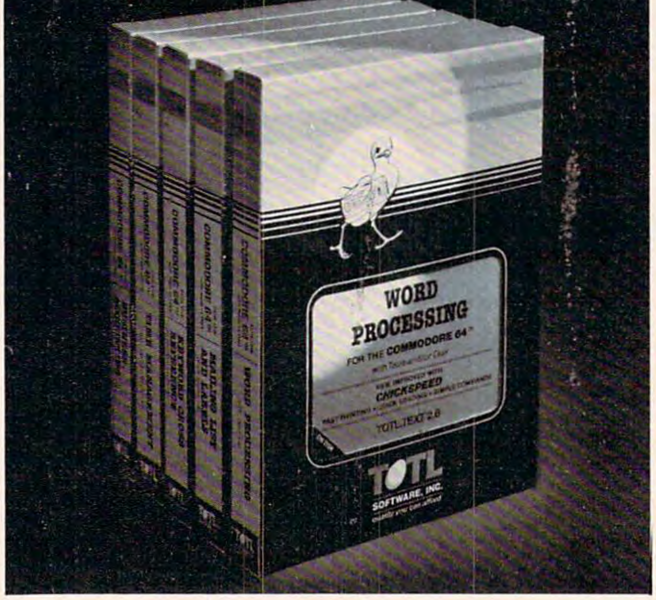
SHIPPING $ 2.50
TOTAL

PAYMENT ENCLOSED: ☐CASH ☐CHECK ☐MONEY ORDER
PLEASE CHARGE TO MY: ☐MASTERCARD ☐VISA (Min. Chg. $25)
CARD # ____________
EXPIRES ________ INTERBANK # ________
NAME ____________
ADDRESS ____________ APT ______
CITY ________ STATE ______ ZIP ______

pace micro software centers

DEPARTMENT: C-P
LOCK BOX 328
BENSENVILLE, IL 60106

There's a New Chick in Town



... with a wing load of Quality Software for the Commodore 64™ and VIC 20™

Once you've compared our programs, their features and prices, you'll agree there's no competition in sight. You'll also discover another important reason to go TOTL... Customer Assistance After You Buy... Something nearly unknown in the low cost software field.

some features

Menu driven, Easy to use ■ Available on tape or disk ■ Compatible With Most Column Expansion Hardware ■ Built-in ASCII Translation for Non Commodore Hardware ■ Color Variables for Easy Modification ■ Machine language speed for Word Processing and Mailing List & Label software.

and there are 5 unique programs to choose from

1. **WORD PROCESSING** has the speed and versatility to produce documents, forms and letters in a straightforward approach that is easily and quickly learned.
2. **MAILING LIST AND LABEL** lets you organize your mailing lists, collection catalogs, menus, recipes and anything that demands listing or sorting.
3. **TOTL TIME MANAGER** helps you plan schedules and analyze events and activities by persons, project catagory and date. Ideal for project planning in the home or business.
4. **RESEARCH ASSISTANT** turns your computer into an advanced, automated indexing and cross reference system. A must for the student, educator or the research professional.
5. **SMALL BUSINESS ACCOUNTING** is a set of straightforward accounting programs. Frees the salesman, entrepreneur or service professional from time consuming record keeping tasks.

TOTL SOFTWARE, INC.
quality you can afford

Ask your dealer about TOTL Software or send in the coupon for further details and ordering information.

1555 Third Avenue, Walnut Creek, CA 94596

PLEASE SEND ME MORE INFORMATION ON TOTL SOFTWARE

Name: ______________________

Address: ______________________

______________________ Zip __________

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 919-275-9809

VIC® 20 OWNERS

Fulfill the expansion needs of your computer with the

RAM-SLOT MACHINE

This versatile memory and slot expansion peripheral for the Commodore Vic-20 Computer consists of a plug-in cartridge with up to 24KBytes of low power CMOS RAM and 3 additional expansion slots for ROM, RAM and I/O. The cartridge also includes a reset button (eliminates using the power-on switch) and an auto start ROM selection switch.

#RSM-8K, 8K RAM + 3 slots....... **$ 84.50**
#RSM-16K, 16K RAM + 3 slots .... **$ 99.50**
#RSM-24K, 24K RAM + 3 slots **$119.50**

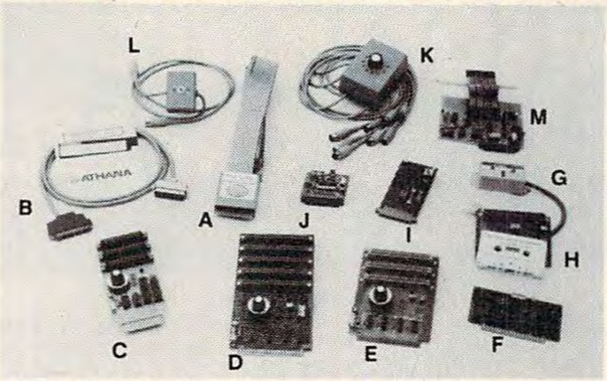
We accept checks, money order, Visa/Mastercard. Add $2.50 for shipping, an additional $2.50 for COD. Michigan residents add 4% sales tax. Personal checks—allow 10 days to clear.

® Trademark of Commodore.

K2 ELECTRONICS DESIGN CORPORATION
3990 Varsity Drive • Ann Arbor, MI 48104 • (313) 973-6266

VIC-20* AND CBM-64* OWNERS

Our products have the features you need. Whether you know it or not!



A. Printmaster-64

A new Centronics parallel printer interface for your Epson or Gemini printer that prints the "full" Commodore graphics set or ASCII data. This "cartridge" adds 16 new commands to the C-64* for simplified printer control.

More features include:

- Single keystroke hi-res graphics or text screen dumps
- Formatted basic listings
- Choice of either graphic symbols or English translation during basic listings
- Left and right margin control
- Special UN new command

Printmaster-64 Cartridge **$89.95**

B. Print-64

A parallel printer interface for the Epson MX80, MX100, RX80, FX80, and Gemini printers. The interface cable connects to the C-64 user-port and with the software diskette supplied allows printing of the "full" Commodore graphics set or standard ASCII data. This package takes full advantage of Graftrax and bit mapped graphics.

Print-64 .... **$69.95**

C. Dataspan-64 for the CBM 64*

a) 3 slots, fuse protected, and master reset button.
b) Rotary switch convenience. Choose any slot, the first two, or all three.
c) Exclusive buffered electronic switching

Dataspan-64 Kit **$49.95**
Dataspan-64 Assembled **$59.95**

D. Dataspan-50 for the VIC-20*

a) 5 slots, fully buffered, fuse protected and master reset button
b) Combination **rotary** and **rocker switch** selectable expansion board conveniently **covers all switching needs** now and in the future without using common hazardous slot-by-slot power switching.
c) Independent **write protection** on two slots

Dataspan-50 Kit **$69.95**
Dataspan-50 Assembled **$84.95**

E. Dataspan-30 for the VIC-20*

a) 3 slots, fuse protected, and master reset button.
b) Rotary switch convenience. Choose any slot, the first two, or all three.
c) Independent **write protection** on two slots.

Dataspan-30 Kit **$34.95**
Dataspan-30 Assembled **$44.95**

F. Champagne Memory on a Beer Budget

Highest quality glass epoxy 16K memory board with gold fingers provides full block switching and write-protection on **each** 8K block. All block switches are conveniently located at the **top edge** of the board.

DataRAM
a) **Bare memory board (RAM/ROM)** **$13.95**
b) **Bare memory board Kit** **$17.95**
includes all components except RAM/ROM chips

DataRAM 8
c) **Board with 8K RAM - Complete Kit** **$34.95**
d) **Board with 8K RAM Assembled** **$37.95**
(Suggested Retail Price Assembled $47.95)

DataRAM 16
e) **Board with 16K RAM - Complete Kit** **$48.95**
f) **Board with 16K RAM Assembled** **$54.95**
(Suggested Retail Price Assembled $69.95)

Protect your investment with our software back-up products

G. CB-2 for the VIC-20* & CBM-64*

The CB-2 is a complete hardware package that allows you to easily and efficiently make a back-up copy of your cassette software library.

Unique features:

- Allows connection for one or two Datasette* recorders (or equivalent). Two recorders required for **simple** back-up copies.
- Exclusive state-of-the-art circuitry lets you **actually hear and see** tape data being loaded or saved.

CB-2 Assembled, Only **$39.95**

H. "Super Blocksaver"

Software and interface card for back-up copies of VIC-20* cartridges. Additional memory required. (Specify cartridge to tape or disk) **$34.95**

CBM-64* Cartridge Back-up System .... **Call**

I. Datablast-16

Finally a low cost, high quality 2716 EPROM programmer for the VIC-20! Put your most often used machine language programs into EPROMs.

- On-board 25 volt power supply.
- Can be used with COMPUTE!'s, "Micromon" or our software below.
- Program/read mode switch.
- We suggest you use the EPROMs with our DataRAM memory board.
- Zero insertion force socket.

Datablast-16
a) **Bare Board** **$13.95**
b) **Kit** **$49.95**
c) **Assembled** **$59.95**
d) **Software for Datablast-16 (tape)** .... **$ 9.95**
e) **2716 EPROMs** .... **$ 4.50**

J. Compuvox

A complete speech synthesis package for the C-64* with arbitrary vocabulary audio output capability. Single general purpose machine language program for random access of vocabulary files.

Compuvox **$69.95**

K. Octopus

A low-cost networking system for connecting as many as 8 C-64's* or VIC-20's* to a single disk drive and/or printer. Manual switching provides direct control and establishes access priority.

Octopus **$79.95**

L. Bi-Net

Network 2 C-64's* or VIC-20's* to a single disk drive and/or printer. Simple manual switching provides full control.

Bi-Net **$24.95**

M. Multi-Lab

Create intelligent custom instrumentation with this electronic "Lab-on-a-Card" for the C-64* and Vic-20*. Make multichannel biomedical oscilloscope, frequency and period counters, voltmeters, bridges, audio function generators, converters, digital sensors, frequency synthesizers, and many more.

Multi-Lab **$239.95**

RAMcharger

- Turn your Commodore 8K cartridge into a full 16K cartridge.
- Full block switching capabilities.
- Sockets allow future EPROM substitution.

RAMcharger Kit **$28.95**

Monitor Cables

VIC-20* **$5.95**
C-64* (1 prong) **$5.95**
C-64* to Color Monitor (3 prong) **$9.95**

CompuTron Business Systems

Marketing Representatives for Digital Interface Systems Co.

1139 S.W. 11th AVENUE • PORTLAND, OREGON 97205 • (503) 224-2220

Kits for Experienced Builders Only!
All assembled units have full 90-Day Limited Guarantee.
*Trademark Commodore Bus. Machines

NOTES: These prices are subject to change without notice. All kits supplied with **complete assembly and operating instructions.**

TERMS: No C.O.D. orders. Shipping and Handling **$3.00**
VISA/MASTERCARD - Add 3% Most orders shipped within 48 hours. (Personal checks — allow 2 weeks to clear)

SEE YOUR DEALER OR ORDER FACTORY DIRECT
DEALERS INQUIRIES INVITED

WE DISCOUNT ALL COMMODORE PRODUCTS

*Send ad with order.

SPECIAL PRICES ON ALL MONITORS	GREEN		$95
	12" AMBER		$109
	13" COLOR		$229
ON PRINTERS			
Gorilla Banana		50 cps	$199
Prowriter 8510		120 cps	$349
STX-80 (Thermal)		60 cps	$169
Gemini 10X		120 cps	CALL
Gemini 15X		15" carriage	CALL

Dust Covers for Commodore Computers or Disk Drives $5.00

Send me: ____________
Name ____________ Phone ____________
Address ____________
City, State, Zip ____________
Driver's Lic. # ____________ Expires ____________

Please complete and mail this form with your check or money order, plus $11.00 shipping & handling, to:

COMPUTER OUTLET, S.D.

5857 Mission Gorge Road, San Diego, CA 92120
619-282-6200

California residents please add 6% sales tax.

Use the handy reader service cards in the back of the magazine for information on products advertised in **COMPUTE!**

NOVEMBER SPECIALS

DOT MATRIX PRINTERS
GORILLA BANANA 50 CPS..........195.00
C.ITOH PROWRITER 120 CPS
PARALLEL..........379.00
SERIAL..........539.00
C.ITOH PROWRITER 2 120 CPS.....679.00
C.ITOH 8600 180CPS NEARLETTER QUALITY
PARALLEL..........1100.00
SERIAL..........1150.00

LETTER QUALITY PRINTERS
C.ITOH STARWRITER 40 CPS......1200.00
C.ITOH PRINTMASTER 55 CPS.....1550.00
NEC 3530 SPINWRITER 33 CPS....1550.00
NEC 3550 SPINWRITER 33 CPS....1975.00
NEC 7710/7730 55 CPS..........2199.00

OTHER EQUIPMENT AVAILABLE
C.ITOH CX-4800 COLOR PRINTER/PLOTTER
4 COLORS, 10" CARRIAGE......560.00
GORILLA 12" HI-RES GREEN SCREEN.89.00
MICROBUFFER STAND-ALONE 32K....225.00
C.ITOH STARWRITER TRACTOR......225.00
C.ITOH 8600 TRACTOR..........120.00

DISKETTES (OTHERS AVAILABLE)
ELEPHANT SS/SD..........20.00
ELEPHANT SS/DD..........23.00
ELEPHANT DS/DD..........29.00
VERBATIM SS/SD..........24.00
VERBATIM SS/DD..........28.00
VERBATIM DS/DD..........41.00
WORDSWORTH SS/SD..........23.00
WORDSWORTH SS/DD..........26.00
WORDSWORTH DS/DD..........30.00

CALL FOR PRICE AND AVAILABILITY OF OTHER LINES OF EQUIPMENT OR DISKETTES

TERMS AND CONDITIONS: NO CASH, NO COD'S, PLEASE. PERSONAL CHECKS/MONEY ORDERS OK. PERSONAL CHECKS ALLOW 2 WEEKS. TEXAS RESIDENTS ADD 5% TAX, HOUSTON RESIDENTS ADD 6% TAX. FOR SHIPPING ADD [illegible], INCLUDE U.P.S. STREET ADDRESS. INCLUDE PHONE NUMBER, PLEASE. PRICES AND AVAILABILITY SUBJECT TO CHANGE WITHOUT NOTICE.

AUTHORIZED LEADING EDGE REPAIR CENTER

MicroDimensions, Inc.

P.O.BOX 10970-G15 HOUSTON,TX.77292
(713)682-2088

It's time for your computer to grow up.

Meet PractiCalc.™ The world's most versatile spreadsheet at only $40.*

Games are fun when it's time to play. But at heart, your Commodore 64™ or VIC-20™ is a full-grown computer straining to gallop at the touch of your fingers.

Commodore 64 and VIC-20. Not just for games...

But the problem is, most software's been designed for toys. Not for real computers. Until now.

Now comes PractiCalc. And there's not much in the world of grown-up computing it won't do. It'll keep addresses (in alphabetical or numerical order), phone numbers (likewise), make budgets and menus, project profits, keep track of expenses, inventories, investments, what have you. For your home, your school, your business.

Budgets, inventories, projections, you name it...

Spreadsheet analysis...

PractiCalc gives you everything you'd expect from a spreadsheet, like adding and subtracting, multiplying and dividing, calculating square roots, logarithms, exponential numbers and even trig functions. And PractiCalc gives you a lot more.

Want to make a chart? Hit a key and the high or low resolution graphics of PractiCalc Plus or PractiCalc 64 will turn numbers into graphs. It'll even print out everything that shows on the screen.

High or low resolution graphics...

Want a lot of facts and figures at your fingertips? PractiCalc stores more than 2,000 cells in up to 100 columns and 250 rows. It'll sort them, search them, or shuffle them for you.

High-speed and wild card search...

You can add entries, delete them, or move them around —numerically or alphabetically, and instantly, of course. And you'll find PractiCalc unusually friendly, flexible, and forgiving.

All of which is a far cry from just playing with a joystick. Still, PractiCalc has one thing in common with a computer game. The price. Only $40.

Alpha or numeric sort...

PRACTICALC™ by COMPUTER SOFTWARE ASSOCIATES, INC.

Exclusively distributed by **Micro Software International, Inc.**
The Silk Mill, 44 Oak Street, Newton Upper Falls, MA 02164 • (617) 527-7510

*$40 is suggested retail price for tape version of PractiCalc-20 ($45 for disk version; PractiCalc Plus or PractiCalc 64 = $50 for tape version, $55 for disk).

VICTORY SOFTWARE

INTRODUCES

THE EARTH WARRIOR SERIES

Programs for the **VIC-20** and the **COMMODORE 64**
All games available on **TAPE** or **DISK** and are packed full of **MACHINE CODE.**

METAMORPHOSIS $19.95

You stumbled into the nest of the Cyglorx and find yourself fighting off robot tanks guarding the Cyglorx eggs. You think you have everything under control and then the eggs start hatching. Commodore 64 version features 4 different screens.
Available for **COMMODORE 64** and **VIC-20.** Played with **JOYSTICK.**

CREATOR'S REVENGE $19.95

The creator assembled a massive army of robots and insects to take revenge on the earth. Destroy insects, get treasures, and get the neutron bomb diactivator. Battle robots and destroy the neutron bomb before it annihilates your city. Miss and you must face the mutants. Features 4 different screens.
Available for **COMMODORE 64.** Played with **JOYSTICK.**

LABYRINTH OF THE CREATOR $19.95

Journey into the most complex and dangerous fortress ever build by the creator. You will encounter deadly robots, skulls, lakes, avalanches, false creators, and a creature who roams 256 rooms relentlessly pursuing you.
Available for **COMMODORE 64.** Played with **JOYSTICK.**

BOUNTY HUNTER $19.95

An adventure in the Old West. Journey back with us into the days of Jessie James and Billy the Kid where the only form of justice was a loaded revolver and a hangman's noose. In this full-length text adventure, you play the role of Bounty Hunter, battling against ruthless outlaws, hostile Indians, wild animals and the elements of the wilderness with only your wits and your six gun. Average solving time: 20-30 hours. If you love adventures, this one is a real treat.
Available for **COMMODORE 64** or the **VIC-20** (with 8K or 16K expander). Played with **KEYBOARD.**

KONGO KONG $19.95

Climb ladders, avoid the barrels the crazy ape is rolling at you, and rescue the damsel. Commodore 64 version features 4 different screens!
Available for **COMMODORE 64.** Played with **JOYSTICK** or **KEYBOARD.**

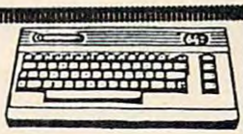
CAVE KOOKS $14.95

Earth's surface is threatened by collapse from a strange group of creatures who bore out the earth's crust to make their dens. Your objective is to enter the creature's habitat and spin the invaders to death.
Available for **COMMODORE 64.** Played with **JOYSTICK.**

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

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

64K



WE DON'T CLAIM ARCADE WE ARE ARCADE

KOOKY CLIMBER

As Alpine Joe, world famous mountain climber you must climb to the top of your apartment building. But watch out for Mr. Boofoo and his trained attack duck, Walter.

An incredible mixture of sound and graphics makes this an all time favorite.

Hires sprites, extended real time music and Arcade quality graphics.

100% MACHINE CODE DISK **29.95**

POGO

- *100% MACHINE CODE*
- *ARCADE QUALITY GRAPHICS*
- *EXTENDED REAL TIME MUSIC*
- *EXCITING ANIMATED SPRITES*

DISK **36.95**

SHOOTING GALLERY

- *100% MACHINE CODE*
- *REAL TIME MUSIC*
- *COLORFUL SCROLLING GRAPHICS*
- *GOOD PLAY ABILITY*

CASSETTE
DISK **27.95**

STAR SLAYER

Sent on a mission, you must destroy all the enemy base stations. But your task won't be easy, they have their best fighter pilots on your tail and they are hard to shake.

—And don't forget about asteroids — they are just as dangerous.

Never before has there been such realism and excitement in one game. STAR SLAYER brings you and your 64 into real-time space combat.

DISK **36.95**
100% MACHINE LANGUAGE

WE ARE RENSOFT SOFTWARE SYSTEMS

SPECIFY CASSETTE OR DISK
N.Y. RESIDENTS PLEASE ADD 7¼% SALES TAX
PLEASE ADD $2.00 SHIPPING AND HANDLING.
PREPAID (CHECK OR MONEY ORDER) OR C.O.D. ONLY

Contact Your Local Dealer.
Or Send Check or Money Order For the Total

DISTRIBUTORS AND DEALER INQUIRES INVITED

SOFTWARE AUTHORS WANTED
DEALER INQUIRIES INVITED

P.O.Box 170 Coram
New York 11727
(516) 732-3390

Atari GTIA Textwriter

Matthew Ratcliff

Here is a utility that will put text on your Atari graphics screens 3-11. It was designed specifically for modes 9-11, thus the name "GTIA Textwriter." If you have tried the GTIA demos published in earlier issues of COMPUTE!, *you know what superb displays can be generated in these modes.*

"Textwriter" gives you 15 colors of text in mode 11, 15 shades of the same color in mode 9, and 8 different color-shade combinations in mode 10. CTIA users, don't despair; this routine will put solid-colored text on the screen in modes 3-8 as well. You can even get two-color characters in modes 3, 5, and 7. Also included is "Color Type," a fast-action typing game. But first, an explanation of Textwriter.

Until now, the only way to get text on the GTIA screens was through Display List Interrupts (DLTs) or slow PLOT, DRAWTO combinations. "Textplot" and "Textplot II" (COMPUTE!, November 1981 and December 1982) will work with a custom character set, but they limit the number of colors allowed to three. The characters will have to be generated according to Figure 1 or 2 (both for up to six colors of text – but characters will have to be duplicated). Every other column must be empty or identical to the one before to prevent the unreadable multicolored text that results when using Textplot with the standard character set. This approach might be sufficient for some needs, but I wanted something that would take full advantage of the 16 possible colors in these modes. The result is a flexible USR routine called GTIA Textwriter.

Character types for using Textplot and Textplot II in GTIA graphics modes 9-11

Figure 1

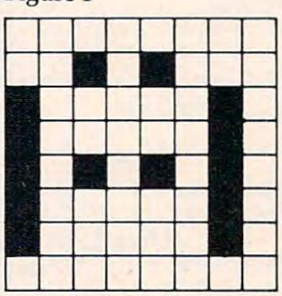
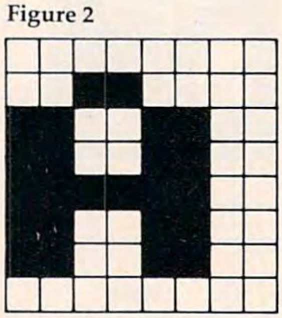


Figure 2



This program is a fully position-independent machine language routine. Using Atari's "auto return" mode, Program 1 converts the DATA statements in lines 10000 through 10400 into a series of A$ string equates (they become lines 11 through 17). When run, it will create these new program lines and stop after printing the following:

LPRINT
LIST "C:",10,18

Use the cursor control keys and RETURN to execute these two commands (LPRINT makes the tape output more reliable). This is the complete GTIA Textwriter program, all 410 bytes of it. Be sure to save the BASIC source code before trying Textwriter, in case you have to go back for a little debugging. Line 18 shows the general command format:

```
18 REM A=USR(AD,BD,COLOR,LEN(B$),ROW
   ,COL) ** AD=ADR(A$) * BD=ADR(B$)
   **
```

To check out Textwriter, try NEW, ENTER "C:" the listed code, and add the following lines:

```
20 GRAPHICS 11:COLOR 1:PLOT 0,0:DRAW
   TO 79,191:C=1
30 A=USR(AD,ADR("TEST"),C,4,90,24)
40 C=C+1:IF C>15 THEN C=1
50 GOTO 30
```

This program will draw a diagonal line across the screen and then flash "TEST" near the center of the screen. If all goes well, you are ready to proceed. If not, go back to the source and track down the problem in the DATA.

Color And Text Variations

Now that you have Textwriter running, let's learn a little about how to take advantage of some of its features. First of all, Textwriter is a very USR friendly routine. It has a flexible calling format to save time and effort in coding. If too much information is given to Textwriter, it executes what it needs, cleans up the stack (ignoring the extra), and returns safely to BASIC. When parameters are left off, certain default conditions are set by the routine. Below is a table showing all the valid calls for Textwriter and the associated default conditions.

Valid Calls And Default Conditions

1. **A=USR(AD,ADR("TEXT"),COLR,4,Y,X)**
   General calling format, no defaults.
2. **A=USR(AD,BD,COLR,LEN(B$),Y)**
   No X, then left justified (X=0).
3. **A=USR(AD,ADR("PLOT"),COLR,4)**
   No Y or X, then placed at first character position following current graphics cursor position (set by PLOT DRAWTO, etc.).
4. **A=USR(AD,ADR("T"),COLR)**
   No X, Y, or length specified. The current graphics cursor position is defaulted and a length of 1 is used.
5. **A=USR(AD,ADR("A"))**
   No X, Y, length, or color. All the #4 defaults are used, and the last color set by the BASIC COLOR command is used.

The only invalid call to Textwriter is A=USR(AD). With no string address it can do little, so it will ring the console buzzer (like Textplot) and return safely to BASIC.

To get three colors of text in modes 3, 5, and 7, use colors 5, 10, and 15. Other color choices will give interesting characters with blank or colored vertical bands through them. The most interesting are colors 6 and 9, which give two-color text. These colors can be changed with SETCOLOR registers 0 and 1. The color band sequence for 6 is the complement of 9, which can be used to make the text appear animated. For example:

```
20 GRAPHICS 21:SETCOLOR 1,5,10:SETCO
   LOR 0,9,8
30 A=USR(AD,ADR("AB"),6,1):FOR J=1 T
   O 50:NEXT J
40 A=USR(AD,ADR("AB"),9,1):FOR J=1 T
   O 50:NEXT J:GOTO 30
```

Use color 15 to get solid text in modes 4, 6, and 8. In mode 8, color 5 results in blue text, and 10 in red text, through artifacting (this is seen best on a black background, GRAPHICS 8:POKE 710,0). If you have Textplot and/or Textplot II in your program library, you will be happy to know that GTIA Textwriter is quite compatible with either of them, since all of Textwriter's workspace is floating point RAM on page zero, which is free for USR routines. These routines can work together to put two different sizes of text on the same screen without having to go to DLIs.

Typing Practice Game

The Color Type game (Program 3) uses GTIA Textwriter and another USR routine. This USR routine and the game's music data will be loaded into strings with Program 2. Run Program 2, creating lines 1-9, ENTER "C:" Textwriter and then LIST "C:",1,18. This code can then be ENTERed into RAM so that you can begin typing in Program 3. The USR routine loaded with Program 2 is a clear screen utility put in CLS$. The screen will blank with random color pixels rather than the basic black you get with a GRAPHICS 11 command. The CLS$ USR routine is called after each word is typed successfully on the screen, since the word destroys part of the background as it is displayed.

Color Type is a touch-typing practice game. It has lots of colors (using GTIA graphics mode 11), sound, and two complete songs. The introductory song played with the title page should be interesting, though not perfect. The second song sounds quite nice, since it is played completely in minor chords (line numbers for DATA in lines 935 through 981 are very important). You must be a pretty good typist to get to this musical part of the game, as you will see.

Press any key to leave the title page and begin typing. Color Type restores to a random line number and reads the sentence stored there, one word or short phrase at a time. The words are put at or near the top of the screen and begin to drift toward the bottom. You must type the word correctly before it gets that far and blows you up, thus ending the game. One wrong key and you have to retype the entire word. After each sentence is typed, the difficulty factor increases and the next sentence appears a little lower on the screen.

After five complete sentences (this factor will vary as difficulty increases), your current score and words per minute (WPM) for that set of sentences (SET) will be displayed. Press the space bar to continue typing or the Q key to quit and see a summary of your typing performance, including:

1) Average WPM
2) Best WPM set
3) Total characters typed
4) Total typo errors
5) Final score

By continuing after each SET, you may reach several bonuses. Once the difficulty is increased to the point where the words originate below the center of the screen, you will get a bonus at the end of the current SET. If you type through six bonus SETs successfully, you will get an extra bonus of 1000 points and hear the second song mentioned above. At an average typing speed of 40 WPM, it takes about 15 minutes to get this far in the game – a good practice session. If you decide to continue, the game play loop will start over with the minimum difficulty. My guess is that an average typing speed of 30-35 WPM is required to ever get this far. If you are a beginner and would like to hear the music more often, replace line 620 with the following:

```
620 DIF=2*TX:IF DIF>60 THEN TX=0:DIF
    =1:GOSUB 840
```

This will give you an extra bonus after every three bonus SETs. If you are a real pro, change the IF statement to DIF>160.

Color Type is fast enough to allow typing speeds of up to 70 WPM. It will not run any faster due to software overhead. I've gotten up to 64 WPM by practicing the same line repeatedly (only 38 WPM on the 400 membrane keyboard). Feel free to customize Program 3 by changing the DATA statements beginning at line 1000 (line increment of 1). Each sentence must begin with a number equal to the total number of words or short phrases in the sentence, followed by the word list, with commas separating the words. For example:

```
1009 DATA 3,THIS,IS A,TEST
```

Upper- and lowercase may be used in the data as well. Some punctuation may be added, like a period at the end of the sentences. Even inverse video or control characters may be used (but this might make the game just a *bit* difficult). If more data is added, the LINES equate at line 220 must be updated accordingly. Note that word length is limited to only nine characters; with more than nine, wraparound will occur.

This program is also good for memorization and spelling practice. If you need to memorize something, arrange the DATA sentences in the proper squence and add the following line:

```
201 LNO=0
```

Then replace line 250 with:

```
250 LNO=LNO+1:IF LNO>LINES THEN LNO=
    0:RESTORE 1000
```

Program 1: GTIA Textwriter

```
5 GRAPHICS 0:? "CONVERTING...11"
10 DIM A$(410),B$(25):AD=ADR(A$):BD=
   ADR(B$)
18 REM A=USR(AD,BD,COLOR,LEN(B$),ROW
   ,COL) **AD=ADR(A$) * BD=ADR(B$)**
19 NA=0:L=11:RESTORE
20 FOR I=1 TO 410
30 READ A:A$(I,I)=CHR$(A)
40 IF INT(I/60)=I/60 AND I>59 OR I=4
   10 THEN LA=NA+1:NA=I:GOSUB 700
50 NEXT I
60 ? "LPRINT":? :? :? "LIST ";CHR$(3
   4);"C:";CHR$(34);",10,18":STOP
700 T=PEEK(709):POKE 709,PEEK(710):?
    CHR$(125)
710 POSITION PEEK(82),2:? :? L;" A$(
    ";LA;",";NA;")=";CHR$(34);
715 FOR K=LA TO NA:? CHR$(27);A$(K,K
    );:NEXT K
720 ? CHR$(34)
730 ? "CONT"
740 POSITION PEEK(82),0:POKE 842,13:
    STOP
750 POKE 842,12:L=L+1:? CHR$(125);"C
    ONVERTING...";L:POKE 709,T:RETUR
    N
10000 DATA 165,200,133,223,169,1,133
      ,222,165,84
10010 DATA 133,229,165,85,133,227,16
      5,86,133,228
10020 DATA 166,87,169,10,224,5,144,8
      ,169,20
10030 DATA 224,7,144,2,169,40,133,23
      9,104,240
10040 DATA 48,170,104,133,213,104,13
      3,212,202,240
10050 DATA 49,104,104,133,223,202,24
      0,42,104,104
10060 DATA 133,222,202,240,35,104,10
      4,133,229,202
10070 DATA 240,22,104,133,228,104,13
      3,227,202,240
10080 DATA 19,104,104,202,208,251,24
      ,144,11,169
10090 DATA 253,76,164,246,169,0,133,
      227,133,228
10100 DATA 165,229,133,233,165,239,1
      33,236,169,0
10110 DATA 133,230,162,8,10,38,230,6
      ,233,144
10120 DATA 7,24,101,236,144,2,230,23
      0,202,208
10130 DATA 239,133,229,165,88,133,21
      6,165,89,133
10140 DATA 217,165,229,24,101,216,13
      3,216,165,230
10150 DATA 101,217,133,217,162,1,165
      ,87,201,9
10160 DATA 176,7,162,2,74,176,2,162,
      3,165
10170 DATA 227,101,228,240,25,70,228
      ,102,227,202
10180 DATA 240,5,70,227,202,208,251,
      230,227,24
10190 DATA 165,227,101,216,133,216,1
      44,2,230,217
10200 DATA 165,216,133,224,165,217,1
      33,225,169,0
10210 DATA 133,226,162,0,169,0,149,2
      18,232,165
10220 DATA 223,149,218,10,10,10,10,2
      32,149,218
10230 DATA 5,219,232,149,218,164,226
      ,177,212,162
10240 DATA 0,134,234,201,128,144,4,4
      1,127,198
10250 DATA 234,170,133,236,224,96,17
      6,13,169,64
10260 DATA 224,32,144,2,169,224,24,1
      01,236,133
10270 DATA 236,169,8,133,233,169,0,1
      33,215,162
10280 DATA 8,10,38,215,6,233,144,7,2
      4,101
10290 DATA 236,144,2,230,215,202,208
      ,239,133,214
10300 DATA 24,173,244,2,101,215,133,
      215,160,0
10310 DATA 132,235,160,8,132,238,24,
      144,3,24
10320 DATA 144,169,164,235,177,214,2
      30,235,69,234
10330 DATA 133,232,169,0,133,237,169
```

COMPUTE!'s
First Book Of Atari Graphics

Authors: **COMPUTE!** Magazine editors and contributors
Price: $12.95
On Sale: Now

COMPUTE!, the leading magazine of home, educational, and recreational computing, has led the way for Atari owners since the computers were first introduced in 1979. **COMPUTE!** has published scores of articles on Atari graphics, and was the first to divulge many important details on such techniques as redefined characters, custom graphics modes, and player/missile graphics. But those articles are scattered across dozens of issues, many of which are scarce or out of print.

That's why the editors of **COMPUTE!** decided to gather the very best Atari graphics articles published over the past three years into *COMPUTE!'s First Book Of Atari Graphics*. From the fundamentals to advanced techniques, here are some of the most instructive articles ever published for the Atari.

But that's not all. *COMPUTE!'s First Book Of Atari Graphics* also presents articles never before published anywhere, and additional sections written especially for this book. These include "The Basics Of Atari Graphics," an introductory tutorial which prepares beginners for the rest of the book; "How To Design Custom Graphics Modes," which covers the fundamentals of mixing modes on a single screen; and "Introduction To Player/Missile Graphics," a guide to understanding one of the Atari's most advanced features, written by Bill Wilkinson, a **COMPUTE!** columnist and a creator of Atari BASIC and the Atari Disk Operating System.

Numerous other articles include "Designing Your Own Character Sets," a new and improved "SuperFont," "High Speed Animation With Character Graphics," "Animation And Player/Missile Graphics," "The Collision Registers," and "GRAPHICS 8 In Four Colors Using Artifacts." There's even a brand new article by Wilkinson, "The Priority Registers," which for the first time shows how to use player/missile graphics to create a *fifth* player.

In the **COMPUTE!** tradition, *Atari Graphics* is crisply written and edited to be useful to beginners and experts alike. And it's spiral-bound for easy access to its dozens of ready-to-type program listings.

V. Introduction . . . . Robert C. Lock

1 Chapter One: Fundamentals Of Atari Graphics
3 The Basics Of Atari Graphics . . . . Tom R. Halfhill
16 Using Strings For Graphics Storage . . . . . . Michael Boom
20 Using The COLOR And LOCATE Instructions To Program Pong-Type Games . . . . . Michael A. Greenspan

23 Chapter Two: Customizing The Graphics Modes
25 How To Design Custom Graphics Modes . . . . . . Craig Chamberlain
37 Put Graphics Modes 1 And 2 At The Bottom Of Your Screen . . . . R. Alan Belke
41 Printing Characters In Mixed Graphics Modes . . . . . Craig Patchett
44 Add A Text Window To GRAPHICS 0 . . . . . . Charles Brannon
46 Mixing Graphics Modes 0 And 8 . . . . . . Douglas Crockford

51 Chapter Three: Redefining Character Sets
53 Designing Your Own Character Sets . . . . . . Craig Patchett
62 SuperFont . . . . Charles Brannon
77 Character Set Utilities . . . . Fred Pinho

89 Chapter Four: Animation With Character Graphics
91 TextPlot . . . . Charles Brannon
98 Using TextPlot For Animated Games . . . . . . David Plotkin
108 High-Speed Animation With Character Graphics . . Charles Brannon

127 Chapter Five: Animation With Player/Missile Graphics
129 Introduction To Player/Missile Graphics . . . . . . Bill Wilkinson
140 A Self-Modifying P/M Graphics Utility . . . . . Kenneth Grace, Jr.
154 Adding High-Speed Vertical Positioning To P/M Graphics . . . . David H. Markley
164 P/M Graphics Made Easy . . . . Tom Sak and Sid Meier
172 Animation And P/M Graphics . . . . . . Tom Sak and Sid Meier
184 Extending Player/Missile Graphics . . . . . . Eric Stoltman
188 The Collision Registers . . . . Matt Giwer
192 The Priority Registers . . . . Bill Wilkinson

201 Chapter Six: Advanced Graphics Techniques
203 GRAPHICS 8 In Four Colors Using Artifacts . . . . . . David Diamond
208 Atari Video Graphics And The New GTIA, Part 1 . Craig Chamberlain
215 Atari Video Graphics And The New GTIA, Part 2 . Craig Chamberlain
224 Atari Video Graphics And The New GTIA, Part 3 . Craig Chamberlain
236 Protecting Memory For P/M And Character Sets . . . . Fred Pinho
239 Screen Save Routine . . . . Joseph Trem

245 Listing Conventions (Guide To Typing In Programs)
246 Index

Available at computer dealers and bookstores nationwide. To order directly call TOLL FREE 800-334-0868. In North Carolina call 919-275-9809. Or send check or money order to **COMPUTE! Books**, P.O. Box 5406, Greensboro, NC 27403.

Add $2 shipping and handling. Outside the U.S. add $5 for air mail, $2 for surface mail. All orders prepaid, U.S. funds only.

```
   ,4,133,231
10340 DATA 169,Ø,133,233,160,2,6,232
   ,38,233
10350 DATA 136,208,249,166,233,181,2
   18,164,237,145
10360 DATA 216,230,237,198,231,208,2
   29,24,165,216
10370 DATA 101,239,133,216,144,2,230
   ,217,198,238
10380 DATA 208,196,198,222,208,1,96,
   230,226,24
10390 DATA 165,224,105,4,133,224,144
   ,2,230,225
10400 DATA 133,216,165,225,133,217,2
   4,144,166,96
```

Program 2: Music Data Loader For Color Type

```
100 DIM CLS$(33),SONG$(124),TUNE$(37
    6):GRAPHICS Ø:? "WORKING ON IT..
    ."
11Ø RESTORE 1ØØØØ
12Ø FOR I=1 TO 33:READ A
13Ø CLS$(I,I)=CHR$(A):NEXT I
14Ø FOR I=1 TO 124 STEP 2
15Ø READ C,N
16Ø SONG$(I,I)=CHR$(C):SONG$(I+1,I+1
    )=CHR$(N):NEXT I
2ØØ ? CHR$(125):POSITION PEEK(82),2:
    ? :? "1 DIM CLS$(33),SONG$(124),
    TUNE$(376):CLS$=";CHR$(34);
21Ø FOR I=1 TO LEN(CLS$):? CHR$(27);
    CLS$(I,I);:NEXT I:? CHR$(34)
22Ø GOSUB 5ØØ:REM ENTER IT
23Ø ? CHR$(125):POSITION PEEK(82),2:
    ? :? "2 SONG$(1,62)=";CHR$(34);
24Ø FOR I=1 TO 62:? CHR$(27);SONG$(I
    ,I);:NEXT I:? CHR$(34)
25Ø GOSUB 5ØØ
26Ø ? CHR$(125):POSITION PEEK(82),2:
    ? "3 SONG$(63,124)=";CHR$(34);
27Ø FOR I=63 TO 124:? CHR$(27);SONG$
    (I,I);:NEXT I:? CHR$(34)
28Ø GOSUB 5ØØ
29Ø RESTORE 6ØØØ
3ØØ FOR I=1 TO 376:READ C:TUNE$(I,I)
    =CHR$(C):SOUND Ø,C,1Ø,3:NEXT I
31Ø D=1
32Ø FOR I=1 TO 5:? CHR$(125):POSITIO
    N PEEK(82),2:? I+3;" TUNE$(";D;"
    ,";D+61;")=";CHR$(34);
33Ø FOR J=D TO D+61:? CHR$(27);TUNE$
    (J,J);:NEXT J
34Ø ? CHR$(34):GOSUB 5ØØ
35Ø D=D+62:NEXT I
36Ø ? CHR$(125):POSITION PEEK(82),2:
    ? "9 TUNE$(311,376)=";CHR$(34);
37Ø FOR I=311 TO 376:? CHR$(27);TUNE
    $(I,I);:NEXT I
38Ø ? CHR$(34):GOSUB 5ØØ
39Ø ? CHR$(125):LIST 1,9
4ØØ ? :? "LPRINT":? :? :? "LIST ";CH
    R$(34);"C:";CHR$(34);",1,9":END
5ØØ ? "CONT"
51Ø POSITION PEEK(82),Ø:POKE 842,13:
    STOP
52Ø POKE 842,12
53Ø RETURN
6ØØØ DATA 5,162,5,162,5,162,5,162,1Ø
     ,128,1Ø,128,5,1Ø8,5,1Ø8,5,1Ø8,5
     ,1Ø8,1Ø,96,1Ø,81,1Ø,121,1Ø,121,
     1Ø,96,5,96
6Ø1Ø DATA 1Ø,81,5,96,1Ø,81,5,72,5,81
     ,1Ø,96,1Ø,81,1Ø,96,1Ø,81,1Ø,96,
     1Ø,96,5,1Ø8,1Ø,96,5,85,3Ø,1Ø8,2
     5,1ØØ
6Ø2Ø DATA 4Ø,1ØØ,5,162,1Ø,162,5,162,
     5,128,5,128,5,128,5,128,1Ø,1Ø8,
     1Ø,1Ø8,1Ø,96,1Ø,81
6Ø3Ø DATA 5,121,1Ø,121,5,121,1Ø,96,5
     ,96,1Ø,81,5,96,1Ø,81,5,72,5,81,
     5,81,15,64,1Ø,72,1Ø,81,5,1Ø8,5,
     96
6Ø4Ø DATA 1Ø,1Ø8,1Ø,128,1Ø,128,1Ø,12
     8,2Ø,162,4Ø,1ØØ,1Ø,96,1Ø,81,1Ø,
     68,15,72,3Ø,81
6Ø5Ø DATA 1Ø,96,1Ø,81,1Ø,81,5,96,1Ø,
     81,5,96,1Ø,72,1Ø,72,1Ø,64,1Ø,96
     ,1Ø,81
6Ø6Ø DATA 1Ø,1ØØ,15,64,15,72,1Ø,81,1
     1Ø,96,5,96,15,81,1Ø,81,1Ø,81,5,
     96,1Ø,81,15,81,1Ø,81,1Ø,72,1Ø,6
     4,1Ø,96,1Ø,81
6Ø7Ø DATA 1Ø,1ØØ,15,68,15,72,1Ø,81,1
     Ø,96,1Ø,96,1Ø,81,1Ø,81,5,96,15,
     81,1Ø,72,1Ø,64,1Ø,96,1Ø,81
6Ø8Ø DATA 2Ø,72,1Ø,72,2Ø,72,11Ø,72,1
     Ø,96,1Ø,64,1Ø,1ØØ,1Ø,47,5,64,15
     ,53,1Ø,64,1Ø,53,1Ø,64,2Ø,1ØØ
61ØØ DATA 1Ø,68,15,72,3Ø,81,1Ø,1ØØ,1
     Ø,96,1Ø,81,5,96,1Ø,81,5,96,1Ø5,
     81,1Ø,72,1Ø,64,1Ø,96,1Ø,81,1Ø,1
     ØØ,15,64,1Ø,1ØØ
611Ø DATA 15,72,11Ø,81,1Ø,96,5,96,15
     ,81,1Ø,81,1Ø,81,5,96,15,81,15,8
     1,1Ø,81,1Ø,72,1Ø,64,1Ø,96,1Ø,81
612Ø DATA 1Ø,1ØØ,15,68,15,72,11Ø,81,
     1Ø,96,1Ø,96,1Ø,81,1Ø,81,5,96,15
     ,81,1Ø,81,1Ø,72,1Ø,64,1Ø,96,1Ø,
     81
613Ø DATA 2Ø,72,1Ø,72,2Ø,72,11Ø,72,1
     Ø,96,1Ø,64,1Ø,53,1Ø,47,5,64,15,
     53,1Ø,64,1Ø,53,1Ø,64,2Ø,1ØØ
1ØØØØ DATA 1Ø4,165,88,133,214,165,89
      ,133,215,169
1ØØ1Ø DATA 3Ø,133,216,16Ø,Ø,173,1Ø,2
      1Ø,9,17
1ØØ2Ø DATA 145,214,23Ø,214,2Ø8,245,2
      3Ø,215,198,216
1ØØ3Ø DATA 2Ø8,239,96
1ØØ4Ø DATA 15,4Ø,5,4Ø,5,35,5,4Ø,5,47
1ØØ5Ø DATA 5,6Ø,2Ø,6Ø,2Ø,72,15,81,5,
      6Ø
1ØØ6Ø DATA 5,47,5,6Ø,5,4Ø,5,47,4Ø,53
1ØØ7Ø DATA 15,4Ø,5,4Ø,5,35,5,4Ø,5,47
1ØØ8Ø DATA 5,6Ø,2Ø,6Ø,2Ø,72,15,81,5,
      6Ø
1ØØ9Ø DATA 5,47,5,53,5,6Ø,5,64,3Ø,6Ø
1Ø1ØØ DATA 1Ø,1ØØ,15,53,5,57,5,53,5,
      47
1Ø11Ø DATA 5,45,5,53,2Ø,47,2Ø,4Ø,15,
      35
1Ø12Ø DATA 5,35,5,4Ø,5,47,5,45,5,47
1Ø13Ø DATA 4Ø,53,15,4Ø,5,4Ø,5,35,5,4
      Ø
1Ø14Ø DATA 5,47,5,6Ø,2Ø,6Ø,2Ø,72,15,
      81
1Ø15Ø DATA 5,6Ø,5,47,5,53,5,6Ø,5,64
1Ø16Ø DATA 4Ø,6Ø,4Ø,1ØØ
```

Program 3: Color Type

```
2Ø SØ=5376Ø:S1=SØ+2:GOSUB 5ØØØ:GOTO
   2ØØ
3Ø MINUTES=(PEEK(2Ø)+256*PEEK(19)+65
   536*PEEK(18))/36ØØ:RETURN
```

```
40 POKE 18,Ø:POKE 19,Ø:POKE 20,Ø:RET
   URN
200 DIM WORD$(1Ø),BL$(1Ø):OPEN #1,4,
    Ø,"K:":RD=5377Ø:BL$="{1Ø SPACES}"
    :BL=ADR(BL$):TX=Ø:CHARCNT=Ø:CLS=
    ADR(CLS$)
21Ø GOSUB 4Ø
22Ø LINES=9:REM TOTAL # OF LINES DEF
    INED IN DATA
23Ø GRAPHICS 11:DIF=Ø:CC=1:SC=Ø:SØ=5
    376Ø:TTLNS=Ø:TCHAR=Ø:TERR=Ø
24Ø TTLNS=TTLNS+1:TRAP 25Ø:SOUND Ø,Ø
    ,1Ø,6
25Ø LNO=INT(RND(I)*LINES)+1ØØØ:RESTO
    RE LNO
26Ø READ WORDS:TRAP 4ØØØØ
27Ø FOR I=1 TO WORDS:READ WORD$:LB=Ø
    :B$="":WW=LEN(WORD$):XP=INT((8Ø-
    8*WW)/2)-8:XP=XP*(XP>Ø):CHARCNT=
    CHARCNT+WW
28Ø A=USR(CLS)
29Ø FOR Y=DIF TO 175
3ØØ CC=CC+1:IF CC>15 THEN CC=1
31Ø A=USR(AD,ADR(WORD$),CC,WW,Y,XP):
    POKE SØ,Y:IF (PEEK(764)=255) THE
    N 38Ø
32Ø GET #1,K:LB=LB+1:B$(LB,LB)=CHR$(
    K)
33Ø Q=LB-1:A=USR(AD,BD+Q,CC,1,183,8*
    Q+XP)
34Ø ON B$=WORD$ GOTO 43Ø
35Ø ON (B$(LB,LB)=WORD$(LB,LB)) GOTO
     38Ø
36Ø FOR Q=15 TO Ø STEP -Ø.5:SOUND Ø,
    2ØØ,8,Q:NEXT Q:SOUND Ø,Y,1Ø,6:PO
    KE 764,255:A=USR(AD,BL,1,LB,183,
    XP):LB=Ø
37Ø SC=SC-WW:B$="":TERR=TERR+1
38Ø NEXT Y
39Ø A=USR(AD,ADR("* CRASH *"),6,9,8Ø
    )
4ØØ A=175:FOR Q=15 TO Ø STEP -Ø.2:PL
    OT XP,A:DRAWTO 8*WW+XP+1,A:IF IN
    T(Q)=Q THEN A=A+1:COLOR Q
41Ø SOUND Ø,PEEK(RD),8,Q:NEXT Q:SC=S
    C-WW:SOUND Ø,Ø,Ø,Ø:POKE 764,255
42Ø COLOR Ø:FOR Q=175 TO 191:PLOT XP
    ,Q:DRAWTO 8*WW+XP+1,Q:NEXT Q:TCH
    AR=TCHAR+CHARCNT:GOTO 74Ø
43Ø SC=SC+WW:POKE SØ,6Ø
44Ø NEXT I:GOSUB 3Ø
45Ø DIF=DIF+6:IF DIF>=143 THEN 48Ø
47Ø IF TTLNS<5 THEN GOTO 24Ø
48Ø TTLNS=Ø
49Ø GRAPHICS 11:B$=STR$(SC)
5ØØ A=USR(AD,ADR("* SCORE *"),5,9,18
    ):I=INT(1Ø-(LEN(B$)/2)+5)*8:IF I
    <Ø THEN I=Ø
51Ø A=USR(AD,BD,3,LEN(B$),4Ø,I):POKE
     764,255
52Ø FOR I=Ø TO 74 STEP 2:COLOR INT(1
    4*RND(I))+1
53Ø PLOT I,Ø:PLOT I,19Ø:POKE SØ,PEEK
    (RD)
54Ø PLOT I,1:PLOT I,189:NEXT I
55Ø FOR I=Ø TO 189 STEP 5:COLOR INT(
    RND(I)*14)+1
56Ø PLOT Ø,I:PLOT 74,I+1:PLOT Ø,I+1:
    PLOT 74,I+1:PLOT Ø,I+2:PLOT 74,I
    +2
57Ø POKE SØ,PEEK(RD):NEXT I
58Ø FOR I=15 TO Ø STEP -Ø.1:SOUND Ø,
    12Ø,12,I:NEXT I
59Ø IF DIF<143 THEN GOTO 65Ø
6ØØ A=USR(AD,ADR("BONUS"),7,5,64,16)
    :TX=TX+1Ø:Q=1ØØ*(TX/1Ø):SC=SC+Q:
    B$=STR$(Q)
61Ø A=USR(AD,BD,9,LEN(B$),8Ø,24)
62Ø DIF=TX*2:IF DIF>1ØØ THEN TX=Ø:DI
    F=1:GOSUB 84Ø
63Ø FOR I=1 TO 5Ø:A=24Ø*RND(Ø):FOR Q
    =Ø TO 15 STEP 5:SOUND Ø,A,1Ø,Q:S
    OUND 1,A+5,1Ø,Q:NEXT Q:NEXT I
64Ø SOUND Ø,Ø,Ø,Ø:SOUND 1,Ø,Ø,Ø
65Ø REM CURRENT WPM
66Ø POKE 764,255
67Ø WPM=INT((CHARCNT/5)/MINUTES):TWP
    M=TWPM+WPM:TSETS=TSETS+1:TCHAR=T
    CHAR+CHARCNT:CHARCNT=Ø
68Ø B$=STR$(WPM):IF WPM>BWPM THEN BW
    PM=WPM
69Ø A=USR(AD,ADR("* WPM *"),CC,7,14Ø
    ,8):CC=CC+1:IF CC>15 THEN CC=1
7ØØ A=USR(AD,BD,CC,LEN(B$),156,24):C
    C=CC+1:IF CC>15 THEN CC=1
71Ø IF PEEK(764)=255 THEN 69Ø
72Ø GET #1,K:IF CHR$(K)="Q" THEN GOT
    O 74Ø
73Ø GRAPHICS 11:GOSUB 4Ø:GOTO 24Ø
74Ø GRAPHICS Ø:POSITION 2,2:IF TSETS
    =Ø THEN TSETS=1
75Ø ? "AVERAGE WORDS PER MINUTE = ";
    INT(TWPM/TSETS)
76Ø POSITION 1Ø,4:? "BEST WPM = ";BW
    PM
77Ø POSITION 2,6:? "TOTAL CHARACTERS
     TYPED = ";TCHAR
78Ø POSITION 6,8:? "TOTAL TYPO ERROR
    S = ";TERR:POSITION 8,1Ø:? "FINA
    L SCORE = ";SC
79Ø POSITION 7,14:? "(S)TOP OR (R)EP
    LAY ";
8ØØ SETCOLOR 2,CC,3:CC=CC+1:IF CC>15
     THEN CC=1
81Ø IF PEEK(764)=255 THEN 8ØØ
82Ø GET #1,A:IF CHR$(A)="S" THEN GRA
    PHICS Ø:END
83Ø RUN
84Ø REM SONG ROUTINE TO GIVE A REST
85Ø S=ADR(SONG$):L=LEN(SONG$):REM TO
     THE WEARY TYPIST
86Ø FOR I=S TO S+L-1 STEP 2
87Ø C=PEEK(I):N1=PEEK(I+1):IF N1=1ØØ
     THEN 89Ø
88Ø RESTORE 9ØØ+N1:READ N2,N3,N4
882 SOUND Ø,N1,1Ø,1Ø:SOUND 1,N2,1Ø,8
    :SOUND 2,N3,1Ø,6:SOUND 3,N4,1Ø,5
89Ø FOR DELA=1 TO 6*C:NEXT DELA:FOR
    G=Ø TO 3:SOUND G,Ø,Ø,Ø:NEXT G:NE
    XT I
9ØØ SC=SC+1ØØØ:A=USR(AD,ADR("1ØØØ"),
    12,4,72,24):RETURN
935 DATA 42,52,7Ø
94Ø DATA 47,6Ø,81
945 DATA 54,67,9Ø
947 DATA 56,7Ø,94
953 DATA 63,79,1Ø6
957 DATA 68,85,114
96Ø DATA 71,9Ø,12Ø
964 DATA 76,96,128
972 DATA 85,1Ø8,144
981 DATA 96,121,162
1ØØØ DATA 16,NOW,IS,THE,TIME,FOR,ALL
     ,GOOD,MEN,TO,COME,TO,THE,AID,OF
     ,THEIR,COUNTRY
1ØØ1 DATA 5,MARY,HAD,A,LITTLE,LAMB
```

```
1002 DATA 4,TIRED,FINGERS,TYPE,SLOWL
     Y
1003 DATA 9,THE,RAIN,IN,SPAIN,FALLS,
     MAINLY,ON,THE,PLAIN
1004 DATA 6,TYPING,IS,GOOD,FOR,THE,F
     INGERS
1005 DATA 5,I,EAT,BANANAS,FOR,LUNCH
1006 DATA 7,TOUCH,TYPING,IS A,TRUE,T
     EST,OF,SKILL
1007 DATA 5,VIDEO,GAMES,ARE,GREAT,FU
     N
1008 DATA 6,WE,SAW A,ZEBRA,AT,THE,ZO
     O
5000 REM TITLE PAGE
5010 GRAPHICS 11
5020 B$="COLOR":CC=1:Y=24
5030 FOR I=0 TO LEN(B$)-1
5040 A=USR(AD,BD+I,CC,1,Y,8*I+16)
5050 CC=CC+1:IF Y=24 THEN Y=28:GOTO
     5070
5060 Y=24
5070 NEXT I
5080 B$="TYPE":Y=48
5090 FOR I=0 TO LEN(B$)-1
5100 A=USR(AD,BD+I,CC,1,Y,8*I+16)
5110 CC=CC+1:IF CC>15 THEN CC=1
5120 IF Y=48 THEN Y=52:GOTO 5140
5130 Y=48
5140 NEXT I
5150 A=USR(AD,ADR(" BY "),9,4,86,16)
5160 T=ADR(TUNE$):SOUND 0,0,10,8:SOU
     ND 1,0,10,6
5170 A=USR(AD,ADR("PRESS ANY"),3,9,1
     48,0):A=USR(AD,ADR("KEY"),4,3,1
     64,24):COLOR 0:PLOT 8,124
5180 FOR I=T TO T+LEN(TUNE$)-1 STEP
     2
5190 C=PEEK(I):N=PEEK(I+1):F=0
5200 IF C>100 THEN C=C-100:F=1
5210 IF N=100 THEN GOTO 5230
5220 POKE S0,N:POKE S1,N+1
5230 A=USR(AD,ADR("MAT RAT"),CC,7):C
     C=CC+1:IF CC>15 THEN CC=1
5240 FOR A=1 TO 1.2*C:NEXT A
5250 IF  NOT (F) THEN POKE S0,0:POKE
      S1,0
5260 IF PEEK(764)<>255 THEN 5280
5270 NEXT I:GOTO 5180
5280 POKE 764,255:SOUND 0,0,0,0:SOUN
     D 1,0,0,0
5290 RETURN
```

©

Using The VIC/64 Function Keys

Jim Butterfield, Associate Editor

The function keys, f1 to f8, seem easy to use and understand. Yet, if you haven't made the right mental connections, they must seem baffling. One of the questions we're asked most often is "How do you make the functions keys work?"

You Can't Input

Let's talk about the INPUT statement for a moment. If your program contains an INPUT statement – or for that matter, if you try typing a direct command – the function keys don't seem to work. They really do work, but to little avail.

The point here is that INPUT takes its information from the screen – and the function keys don't show up on the screen.

When you press one of the keys, it is received and placed into the keyboard input buffer. During an input or direct statement – in other words, whenever the cursor is flashing – the keyboard buffer is promptly emptied and the characters there are printed to the screen. There's the problem: f1, f2, etc., have no printable equivalent, so at this point the characters are lost. Later, INPUT will see you press RETURN and will take its information from the screen, but there are no f-characters there.

In other words, INPUT or normal direct statements will lose the function keys. There's a way around this, but it's awkward: put your input within quotation marks, and the keys will be detected. They will also print oddly, but that's another story.

Another Story – GET

The GET command takes information directly from the keyboard buffer, so it will read these keys without problem. The question is: how does your program test to see if it has an f-key? The answer is easy, but it's rather graphic in nature – so try this on your machine:

```
100 GET X$
110 IF X$="
```

and hold it right there.

At this moment, we're in the middle of line 110, and we've just typed the quotation mark. Now, press the f1 key, and you'll see an odd reverse graphic symbol printed. It's the "programmed cursor" equivalent of the key f1; it looks like a reversed horizontal bar, and for all intents and purposes it is key f1. Now finish the line so it looks like this:

```
110 IF X$="f1" THEN PRINT "FUNCTION 1"
```

Note to readers who have been skimming: don't type the characters f and 1 within the quotation marks; tap key f1 at this point.

Using the same system, we may work through all eight functions:

```
120 IF X$="f2" THEN PRINT "FUNCTION 2"
```

Key f2 is f1 with the shift key held down, of course; it prints a reverse quarter-circle. Keep going:

```
130 IF X$="f3" THEN PRINT "FUNCTION 3"
```

and so on until:

```
180 IF X$="f8" THEN PRINT "FUNCTION 8"
190 GOTO 100
```

You can run this program and play with the f-keys as long as you like. As you can see, the computer recognizes the keys without trouble.

In larger programs, you'll often want to GOTO when you see a given key. That's no trouble at all, of course.

Another Way

Those funny characters can be puzzling. They are hard to read and may be confused with each other. There's no listing standard for them yet. Sometimes it's useful to write them another way, without the funny characters.

```
100 GET X$:IF X$="" GOTO 100
110 X=ASC(X$)
```

We've changed our input key to an ASCII number. Every key has its own ASCII value; if we know the value, we'll know which key.

Now I could tell you the ASCII values for the eight function keys, but I'm not going to do that. Instead, I'll tell you how to find these values for yourself.

Suppose you want to find the ASCII number for key f1. Just type:

```
PRINT ASC("f1")
```

Remember to press the f1 key (don't type f and 1 as two characters), and you'll see the computer respond with a value of 133. That's the ASCII value of key f1.

Now we can continue the above program:

```
120 IF X=133 THEN PRINT "FN 1"
130 IF X=137 THEN PRINT "FN 2"
```

continuing to:

```
190 IF X=140 THEN PRINT "FN 8"
200 GOTO 100
```

I haven't given you the ASCII numbers to fill in the missing lines – but with a little care and attention, you can find them for yourself.

A Simple Example

Let's do a simple quiz, using the odd-numbered f-keys.

```
100 DATA WHO DISCOVERED AMERICA
110 DATA GALILEO, COLUMBUS, REAGAN, EINST
    EIN
120 DATA 2
130 DATA THE CHARGE ON AN ELECTRON IS
140 DATA NEUTRAL,POSITIVE,NEGATIVE,VARIAB
    LE
150 DATA 3
160 DATA UGANDA IS IN
170 DATA ASIA, SOUTH AMERICA, EUROPE, AFR
    ICA
180 DATA 4
190 DATA "*"
200 READ Q$:IF Q$="*" THEN END
210 PRINT Q$;"--"
220 READ A$:PRINT"F1 - ";A$
230 READ A$:PRINT"F3 - ";A$
240 READ A$:PRINT"F5 - ";A$
250 READ A$:PRINT"F7 - ";A$
260 PRINT "YOUR ANSWER? ";
270 READ A
280 GET X$:IF X$="" GOTO 280
290 X=ASC(X$)
300 IF X<133 OR X>136 GOTO 280
310 X=X-132:PRINT "F";X*2-1
320 IF X=A THEN PRINT "RIGHT!":GOTO 340
330 PRINT "WRONG!"
340 GOTO 200
```

You'll notice that line 220 calls for you to type the actual characters (F and 1), and the same goes for lines 230 to 250.

The program isn't the definitive educational package – but it does show how the function keys can be used effectively.

Without A Program

Sometimes you might like to have the function keys do something even when there is no program running. That's much tougher: if your program is not running, it can't do the job. You may have noticed that packages like the Super Expander provide this feature: pressing the f1 key might produce the word GRAPHIC on the screen.

This kind of thing utilizes advanced techniques. You would need to know machine language, and how to implement a wedge. It can be done – but it's not for beginners. ©

All About The TI Character Set

Michael A Covington

This brief outline of the TI character set explains how the computer recognizes each character. The author discusses some uses of the characters' numeric codes and indicates which characters' graphic representations can be assigned or changed.

Chances are you've never given your computer's character set much thought. You press keys on the keyboard and the characters appear on the screen; that's all there is to it, or so it seems. But there's a lot more going on than meets the eye.

Inside the computer, each character is represented by a *numeric code* – a number between 0 and 255 inclusive. For instance, the code for capital E is 69; the code for an exclamation mark is 33; the code for a blank (a blank is a character just like all the others) is 32. To associate these codes with the characters you see on the screen, the computer has to know two more things about each of them: a *graphic representation* that describes how the character is supposed to look on the screen, and a *key assignment* that indicates what key or combination of keys you can hit on the keyboard to type the character. For instance, the character string "HELLO THERE!" (not counting the quotation marks) involves the following:

Graphic representation:	H	E	L	L	O		T	H	E	R	E	!
Numeric code:	72	69	76	76	79	32	84	72	69	82	69	33
Key assignment:	H key	E key	L key	L key	O key	space bar	T key	H key	E key	R key	E key	shift & 1 keys

Statements Using Numeric Codes

Normally (when you type characters in response to a string INPUT statement or when you type them as part of a program) you enter characters by hitting the keys that correspond to them. That is, you access them by means of their key assignments, and within the program you treat them as character-string data. But there are ways of referring to characters by their numeric codes and treating them as numbers. For instance, the CALL HCHAR and CALL VCHAR statements, which you meet at an early stage as you work through the manuals that come with the computer, refer to characters by their numbers. The statement

```
CALL HCHAR(3,3,69,2Ø)
```

will place a row of 20 capital E's (character number 69) on the screen beginning at row 3, column 3.

Also, you can input characters as numeric codes. The CALL KEY statement senses whether a particular key on the keyboard is up or down; when a key is pressed, CALL KEY gives you the numeric code corresponding to it. For instance, here is a program which will tell you the numeric code of any key on the keyboard:

```
1Ø PRINT "PRESS ANY KEY..."
2Ø CALL KEY(5,CODE,STATUS)
3Ø IF STATUS <> 1 THEN 2Ø
4Ø PRINT CODE
5Ø GO TO 1Ø
```

The heart of the program is lines 20 and 30. Line 20 tells the CALL KEY subroutine to look at the keyboard and report what's going on. The variable STATUS will equal 1 only if the condition of the keyboard has changed since the last time the routine looked at it. If STATUS does not equal 1, we simply go back to line 20, since we don't want to do anything more if the user hasn't pressed a key or hasn't yet let go of the one already looked at. The variable CODE contains the numeric code associated with the key being pressed, if any. (The first parameter of CALL KEY, the number 5, simply indicates that we want the usual BASIC set of codes; specifying other numbers there instructs the computer to use other sets of key assignments for various special purposes.)

The ASC and CHR$ functions allow you to convert back and forth between numeric codes and character strings. If A$ is a character string, ASC(A$) is the numeric code of its first character; thus ASC("E") is 69. Conversely, if N is a number, CHR$(N) is a one-character string of which N is the numeric code; thus CHR$(69) is E. If we want the program above to print the characters themselves rather than their codes, we can convert the codes into characters by changing line 40 to:

```
40 PRINT CHR$(CODE)
```

The CALL CHAR subroutine allows you to alter graphic representations using a hexadecimal code that the manual describes in detail. For instance, if you want to change the dollar sign ($) into a British pound sign (£), just execute the statement:

```
CALL CHAR(36,"001C22207C20207E")
```

That will do it, at least as long as the program is running: the key assignment and numeric code will be the same, but the dollar sign will look like a pound sign. (It will revert to its original appearance when your program stops executing.)

What's Not In The Manual

Those are the preliminaries; now we get to the really interesting part (the part that isn't in the manual, at least not entirely). Internally, the computer can use any number from 0 to 255 as a character code; any such code can be an element in a character string and can be referred to by CALL VCHAR, CALL HCHAR, and CHR$. (In fact, CALL VCHAR, CALL HCHAR, and CHR$ will actually take numbers up to 32767; multiples of 256 are subtracted as necessary to get a number in the 0-to-255 range.) But not all the codes have key assignments or graphic representations. The breakdown (by numeric codes) is as follows:

0 – Undefined (no key assignment, no graphic representation).

1 to 15 – Function keys (Table 1). Most of these characters can be input by means of the CALL KEY statement, but they cannot be typed in normal contexts (for example, in response to an INPUT) because there they are interpreted as requests to perform cursor movements or the like. They have no graphic representations (if you print them, you get blanks or garbled patches).

16 to 29 – Undefined (like 0, these codes have no key assignments and no graphic representations, and there is no straightforward way of giving them either).

30 – The graphic representation of this character is the black square that marks the cursor; thus, CHR$(30) is handy if you want a black square. No key is assigned to it.

31 – This is the screen border character – a blank that is the color of the border rather than the typing area. No key is assigned to it.

32 to 126 – Standard ASCII characters (Table 2). These are the characters you use every day, including the alphabet, the numbers, and all the punctuation marks and mathematical symbols. Their graphic representations can be changed with CALL CHAR but will revert to their original form when the program ends.

127 to 159 – User-defined characters (Table 3). These start out with no graphic representations, but you can define them with CALL CHAR, and, contrary to what the TI manual says, such definitions remain in effect after the program stops running (though most are disrupted when another program is loaded).

What most people don't realize is that these characters can be typed – they have key assignments and are acceptable in the same context as any other character (that is, in response to an INPUT or CALL KEY, or within quotes in a program). All but one of them require you to hold down the CTRL key (at the lower-left corner of the keyboard) when typing them; character number 127 uses the FCTN key instead.

160 to 175 – Undefined.

176 to 198 – These characters have key assignments (Table 4), but no graphic representations and no direct way of giving them any. They can be used as special function keys of some sort (in response to either CALL KEY or INPUT), but not as displayable characters.

199 to 255 – Undefined.

Even the "undefined" character codes (those that cannot be typed on the keyboard or displayed on the screen) are not completely useless. You can refer to them by means of CHR$ and ASC and use them as special markers of various kinds when manipulating character strings. They also may come into play when you are transmitting data to other devices (for example, printers or other computers) that have definitions for characters that are undefined on the TI-99.

Finally, consider this possibility. Each character in a character string has a code between 0 and 255 inclusive, accessible through CHR$ and ASC. Also, the SEG$ function allows you to address individual characters in a string, and the & (concatenation) operator allows you to construct strings out of individual characters. This means that a character string gives you a compact way of storing a set of integers between 0 and 255 – each element occupies only one byte in memory, as compared to the eight bytes normally needed to store a number. So if you have a program that needs to keep track of thousands of small integers – more than will fit in available memory in numeric form – then character strings may be the answer.

Table 1:
Function Key Codes

(None of these characters have graphic representations, nor can they be given them. They can be typed only through the CALL KEY statement, not in response to a string INPUT statement, or within a program.)

Code	Key
1	FCTN 7 ("AID")
2	None usable. The key definition associated with this code is FCTN 4, but in BASIC, hitting that key interrupts the program.
3	FCTN 1 ("DELETE")
4	FCTN 2 ("INSERT")
5	None usable. The key definition associated with this code is FCTN =, but hitting that key forces a machine reset and the program in memory is lost.
6	FCTN 8 ("REDO")
7	FCTN 3 ("ERASE")
8	FCTN S (left arrow)
9	FCTN D (right arrow)
10	FCTN X (down arrow)
11	FCTN E (up arrow)
12	FCTN 6 ("PROC'D")
13	ENTER
14	FCTN 5 ("BEGIN")
15	FCTN 9 ("BACK")

Table 3:
User-definable Graphics Characters

These characters can be typed using the key combinations listed and are acceptable in any context (that is, they can be input using the CALL KEY or INPUT statements and can appear between quotes within a BASIC program).

Graphic representations can be given to these characters with the CALL CHAR statement. Contrary to TI documentation, such representations, once assigned, will persist after the program stops running.

Code	Key	Code	Key
127	FCTN V	144	CTRL P
128	CTRL , (comma)	145	CTRL Q
129	CTRL A	146	CTRL R
130	CTRL B	147	CTRL S
131	CTRL C	148	CTRL T
132	CTRL D	149	CTRL U
133	CTRL E	150	CTRL V
134	CTRL F	151	CTRL W
135	CTRL G	152	CTRL X
136	CTRL H	153	CTRL Y
137	CTRL I	154	CTRL Z
138	CTRL J	155	CTRL . (period)
139	CTRL K	156	CTRL ;
140	CTRL L	157	CTRL =
141	CTRL M	158	CTRL 8
142	CTRL N	159	CTRL 9
143	CTRL O		

Table 2:
ASCII Graphic Characters On The TI-99/4A

(This table gives the numeric codes and graphic representations; the key assignments are marked on the keyboard. The graphic representations can be changed by the CALL CHAR statements but revert to their original form when the program stops running.)

Code	Graphic Representation	Code	Graphic Representation
32	(space)	53	5
33	!	54	6
34	"	55	7
35	#	56	8
36	$	57	9
37	%	58	:
38	&	59	;
39	'	60	<
40	(	61	=
41	)	62	>
42	*	63	?
43	+	64	@
44	,	65	A
45	- (minus)	66	B
46	.	67	C
47	/	68	D
48	0	69	E
49	1	70	F
50	2	71	G
51	3	72	H
52	4	73	I
74	J	97	a
75	K	98	b
76	L	99	c
77	M	100	d
78	N	101	e
79	O	102	f
80	P	103	g
81	Q	104	h
82	R	105	i
83	S	106	j
84	T	107	k
85	U	108	l
86	V	109	m
87	W	110	n
88	X	111	o
89	Y	112	p
90	Z	113	q
91	[	114	r
92	\ (back slash)	115	s
93	]	116	t
94	^	117	u
95	_ (underline)	118	v
96		119	w
120	x		
121	y		
122	z		
123	{		
124	|		
125	}		
126	~		
127	See Table 3. TI documentation mistakenly classifies this character with the wrong group.		

Table 4:
Characters With Key Assignments But No Graphic Representations

These characters are not mentioned in TI documentation. They can be typed in any context (that is, in response to an INPUT or CALL KEY statement or between quotes in a program), but they have no graphic representations and cannot be given any.

Code	Key	Code	Key
176	CTRL 0	188	FCTN 0 (zero)
177	CTRL 1	189	FCTN ;
178	CTRL 2	190	FCTN B
179	CTRL 3	191	FCTN H
180	CTRL 4	192	FCTN J
181	CTRL 5	193	FCTN K
182	CTRL 6	194	FCTN L
183	CTRL 7	195	FCTN M
184	FCTN , (comma)	196	FCTN N
185	FCTN . (period)	197	FCTN Q
186	FCTN /	198	FCTN Y
187	CTRL /		

©

VIC/64 Tape Aids

Andrew Au

When you get a LOAD ERROR, what can you do? Can you recover the program? This article deals with the frustrating problem of tape loading errors. Also, there's a technique here which allows you to LOAD programs twice as fast as usual.

When you save your program, the VIC or 64 saves it twice on the tape. When the program is loaded, the first version is put into the computer and checked against the second version. Any mismatch will cause the LOAD ERROR.

If you get a load error, LIST the program. If it LISTs properly to the end, it can probably be recovered. Don't RUN the program yet, or it will be destroyed. The problem here is that the start of variable pointers are not set. Set these pointers by POKEing the value found by PEEKing 831 into locations 45,47,49; and POKEing the value found by PEEKing 832 into 46,48,50. After performing these six POKEs, make sure that you have done everything correctly by typing PRINT FRE(0). If it shows a decrease in free RAM, your program is ready. Type RUN, and it should work.

However, if the computer just locks up, you have POKEd the wrong values. Press RUN-STOP/RESTORE and rewind the tape to the beginning. Type OPEN 1 and play the tape. After the header is loaded, it will stop. Now 831 and 832 should contain the location of the variable pointer. Perform the above POKEs and try again.

The Solution

When the VIC or 64 loads a program, it sets zero page pointers (which tell it how long the program is) and loads the program itself into BASIC RAM. Apparently, the computer sets the pointers after the program is loaded and checked. If the computer detects a load error, it does not update the pointers. So although you can LIST the program, the computer thinks there is nothing in memory. If the program is RUN at this stage, the variables will overwrite the program and destroy it.

The POKEs given above set the variable pointers to the correct values, which are found in the cassette buffer. Locations 829,830 and 831,832 hold the starting addresses (Lo/Hi) of the program and variables, respectively.

This cure works only if the program itself is loaded successfully. If the program is garbled, this method will not work. Fortunately, many load errors result from this pointer problem, so this technique is well worth knowing.

Loading Time Cut In Half

An 8K program takes two and a half minutes to load. Since the computer loads the program and then checks it, the actual loading takes only half the time. The value of the verification is doubtful since it won't correct any detected errors. All it does is report ?LOADING ERROR. As a rule, machine language programs can be stopped at the middle of the load and RUN (or SYSed to) without problems, since there are no zero page pointers involved. On the other hand, BASIC programs need more attention if they are to RUN at "half time." Since the pointers are not set until the end of the load, they must be typed in manually or incorporated into the program.

Let's prepare a program so that it can be stopped at the middle and RUN (and still work). Add a line at the beginning of the program:

```
POKE 46,PEEK(832): POKE 48,PEEK(832): POK
  E 50,PEEK(832): POKE 45,PEEK(831): POKE
   47,PEEK(831): POKE 49,PEEK(831)
```

Type in the above line (or make it two lines), and make sure it is at the beginning of your program. Now SAVE the program.

When this program is RUN after being stopped at the middle, it will first reset the pointers. Now we don't have to wait for the computer to do it at the end. The result: the program LOADs twice as fast.

This technique can be used with any program to shorten loading time. Use the tape counter to find the middle of the program and allow it to go slightly past.

If you have consistent tape problems, try changing the location of your cassette recorder. Remember that data transmission is sensitive to electrical fields. You should keep the connecting cord away from the back of the TV, which is a strong source. Always verify your programs after saving. ©

Products for Commodore, Atari, Apple, and others!

NEW

THE MONKEY WRENCH II

A PROGRAMMERS AID FOR ATARI 800

NEW AND IMPROVED — 18 COMMANDS
PLUGS INTO RIGHT CARTRIDGE SLOT

$59.95

If you are a person who likes to monkey around with the ATARI 800, then THE MONKEY WRENCH II is for you!! Make your programming tasks easier, less time-consuming and more fun. Why spend extra hours working on a BASIC program when the MONKEY WRENCH can do it for you in seconds. It can also make backup copies of boot type cassette programs. Plugs into the right slot and works with ATARI BASIC cartridge.

The MONKEY WRENCH provides 18 direct mode commands. They are: AUTO LINE NUMBERING — Provides new line numbers when entering BASIC program lines. RENUMBER — Renumbers BASIC's line numbers including internal references. DELETE LINE NUMBERS — Removes a range BASIC line numbers.

VARIABLES — Display all BASIC variables and their current value. Scrolling — Use the START & SELECT keys to display BASIC lines automatically. Scroll up or down BASIC program. FIND STRING — Find every occurrence of a string, XCHANGE STRING — Find every occurrence of a string and replace it with another string. MOVE LINES — Move lines from one part of program to another part of program. COPY LINES — Copy lines from one part of program to another part of program. FORMATTED LIST — Print BASIC program in special line format and automatic page numbering. DISK DIRECTORY — Display Disk Directory. CHANGE MARGINS — Provides the capability to easily change the screen margins. MEMORY TEST — Provides the capability to test RAM memory. CURSOR EXCHANGE — Allows usage of the cursor keys without holding down the CTRL key. UPPER CASE LOCK — Keeps the computer in the upper case character set. HEX CONVERSION — Converts a hexadecimal number to a decimal number. DECIMAL CONVERSION — Converts a decimal number to a hexadecimal number. MONITOR — Enter the machine language monitor.

In addition to the BASIC commands, the Monkey Wrench also contains a machine language monitor with 16 commands used to interact with the powerful features of the 6502 microprocessor.

NEW

VIC RABBIT CARTRIDGE AND CBM 64 RABBIT CARTRIDGE

NEW FEATURE!
DATA FILES!

"High Speed Cassette Load and Save!"

$39.95

(includes cartridge and manual)

for VIC

Don't waste your Life away waiting to LOAD and SAVE programs on Cassette Deck.

Load or Save 8K in approximately 30 seconds! Try it—your Un-Rabbitized VIC or 64 takes almost 3 minutes. It's not only fast but VERY RELIABLE.

Almost as fast as 1541 Disk Drive! Don't be foolish — Why buy the disk when you can get the Rabbit for much, much less!

Allows one to APPEND Basic Programs!
Easy to install — just plugs in.
Expansion Connector on rear of the VIC Rabbit.
Works with or without Expansion Memory.
Works with VIC or 64 Cassette Deck.
12 Commands provide other neat features.
Fast Data Files - two data file modes.
Also Available for 2001, 4001, and 8032.

Now for the "64"

STCP — 300/1200 Baud

Standard Terminal Communications Package

·PFO· IOD OOA CP<D1>D2 BELL = 12:30:00 10:14:36

Don't settle for non-standard Communications Protocol! Access Micro Net, Source, Bulletin Boards, Local Mainframe, etc.

- Complete Package — Includes RS232 Interface Board and software (does not include modem)
- Communicates in Industry Standard ASCII
- Upload/Download to/from Disk
- Automatic File Translation
- Can be controlled from keyboard or user supplied basic or machine language program

Specify 3.0 or 4.0 ROMS or 8032 Commodore Computer 4040 or 8050 or PEDISK II Disk or CBM64 on 1541.

Price: $129.95

ATARI AND PET EPROM PROGRAMMER

Programs 2716 and 2532 EPROMs. Includes hardware and software. PET = $75.00 — ATARI (includes sophisticated machine language monitor) = $119.95

Prowriter Printer - Excellent dot matrix print parallel =call
Serial = Call IEEE = Call

PET BASIC SCROLL PROGRAM

Scroll thru Basic Programs using cursor up/down keys. Specify computer. $6.00 on cassette, $9.00 on diskette.

65C02 MAE

Same as our MAE but enhanced for the new 65C02 Opcodes. Turns your computer into a development system for the new ROCKWELL 65C02 Microprocessor. $200.00 — Specify Computer.

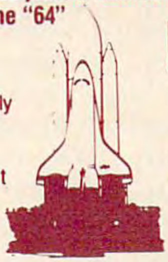
EHS's CBM64 Monitor Cartridge

$25.00

Every Commodore 64 owner should have this Machine Language Monitor to explore the interesting and exciting world of Machine Level Code. If you can't think of a need for it now, some day you'll wish it was in your Software Library. Commands include Display Registers/Memory, Disassemble, Assemble, Transfer, etc. Over 20 commands.
Cartridge and Manual - 25.00.

More than just an Assembler/Editor!
Now for the "64"

It's a Professionally Designed Software Development System



MAE

for
PET
APPLE
ATARI
~~$169.95~~
New
Price
~~$99.95~~

XMAS Special $59.95

Blast off with the software used on the space shuttle project!

- Designed to improve Programmer Productivity
- Similar syntax and commands — No need to relearn peculiar syntaxes and commands when you go from PET to APPLE to ATARI.
- Coresident Assembler/Editor — No need to load the Editor then the Assembler then the Editor, etc.
- Also includes Word Processor, Relocating Loader, and much more.
- Options: EPROM Programmer, unimplemented opcode circuitry.
- STILL NOT CONVINCED Send for free spec sheet!

5¼ INCH SOFT SECTORED DISKETTES

Highest quality. We use them on our PETs, APPLEs, ATARIs, and other computers. $22.50/10 or $44.50/20

EPROMS 2716 = $4.50 2532 = $7.50
Over 40 Commodore Programs by Baker (on 4040) = $25.00

eastern house

3239 Linda Dr.
Winston-Salem, N.C. 27106
(919) 924-2889 (919) 748-8446
Send for free catalog!

VISA · MasterCard

Make Your Apple User-Friendly

Karen Goeller McCullough

With an Apple II and a disk drive, you can use this versatile utility program to create menus that call other programs – or you can merge it with your own multifunction programs to create an effective master menu.

If you have an Apple II with Applesoft BASIC and at least one disk drive, this handy utility can save time and prevent confusion by generating menu programs. All you do is tell the program the number of options on your menu and their names. From that information a BASIC program is generated which presents a nicely formatted display of the options, allows the entry of a selection, and checks it for validity.

The figure shows a sample menu created using "Menu Generator." You simply add the code to tell the program where to go after the option has been selected. The Menu program can be used on its own to call other programs, or it can be merged into your own programs using the renumber and merge options in the "Renumber" program on the *System Master* diskette.

```
<1> INITIALIZE DATA DISKETTE
<2> SET UP NEW FILE
<3> ADD ENTRIES TO FILE
<4> CHANGE ENTRIES ON FILE
<5> DELETE ENTRIES ON FILE
<6> PRINT MAILING LIST
<7> PRINT MAILING LABELS
<8> EXIT FROM SYSTEM
```

Menu Generator uses an Apple DOS feature to create a program as a text file and then EXEC it. The EXEC command treats a text file as a series of commands that are executed just as though they had been entered from the keyboard. Delayed execution commands (those that have line numbers in front of them) are saved in memory to await a RUN command. (For more information on EXEC files, see pages 75-79 of the *DOS Manual*.) The EXEC command lets you write a BASIC program that will produce another BASIC program which can be immediately EXECed into memory and RUN, or SAVEd to disk as a program file. Menu Generator is an example of how this feature can be used for almost unlimited program generation.

Program Breakdown

In the Menu Generator program, lines 10-50 initialize the screen and variables. Line 100 sends us to line 1000 to begin processing. Lines 200 and 250 are subroutines that clear either a part of the screen (200) or a given line V (250). These are placed close to the beginning of the program to speed execution.

Lines 1000-1060 input the number of options desired on the menu. A string variable is used to input the number, and lines 1030-1040 scan the input string for valid numeric characters (ASCII 48-57). If an invalid character is detected, a flag (E) is set. The flag is then tested in line 1050, and, if true, execution is returned to the input statement at 1020.

This may seem a cumbersome process, since using a numeric variable would obviate the need for lines 1030-1050. However, in applications where an attractive screen format is important, this routine avoids the ?REENTER statement which appears if you try to enter a nonnumeric character into a numeric variable.

The options to appear on the menu are entered in lines 1070-1200. The array OP$ holds the option names, and the array element number also functions as the option number. For example, if option number 1 on the menu is to be INITIALIZE

DATA DISKS, then that will be the contents of OP$(1). After all the options are entered and checked for length of less than 30 characters, the program checks to see if changes are desired (1210-1420).

Beginning at line 2000, the text file which builds the program is created. The text file is opened in lines 2010-2040, and the first line is printed at 2045. Since the EXEC command itself does not clear the program currently in memory, the first line of the exec file issues an FP command, which prevents the EXECed program from overlaying the calling program. The POKE 34,24 in lines 2047 and 1420 prevents the screen from scrolling and the cursor from bouncing around while the EXEC file is being processed.

Menu Generator Variable List

A$	– yes or no answer input
CH$	– holds a single character for error checking
D$	– return + control-D (CHR$(13) + CHR$(4))
E	– error flag
H	– horizontal print location
I	– counter for FOR/NEXT loops
L	– length of longest option
L1	– temporarily holds length of each option
N	– number of options on menu
N$	– number of options (input string)
NN	– option number to change
NN$	– option number to change (input string)
N1	– option number selected on menu
OP$	– array holding option names
Q$	– quote mark (CHR$(34))
V	– vertical print location

Creating The New Program

The beginning of the new program being created (the menu program itself) is at line 2050. Lines 2050-2220 actually write the menu program, beginning with the header which will be lines 10-30 in the new program (lines 2050-2070 in the creating program). The variable N is set equal to the number of options, and the array OP$ is DIMed to N in line 40 (2080). The array OP$ is loaded with the option names in line 50 (2090), and line 2110 of the creating program causes the OP$ array to be printed as the DATA statement of line 70 of the new program. The length of the longest option line is found in line 2120; this information is used to calculate the horizontal positioning in line 2140. The same line also calculates the vertical positioning using the number of options (N).

After displaying the menu options, the program asks for the selection to be input. Input and validation procedures (2180-2220) are the same as those used for the option number input in the creating program. Line 2220 is the end of the delayed execution part of the text file, and it remains in memory while the EXEC function continues to the last two lines of the text file. Line 3010 causes the program which has been LOADed into memory from the text file to be SAVEd to disk as a program file called MENU-PROGRAM, and the next line causes it to be RUN.

Printing of the text file is concluded by line 3040, which CLOSEs the text file. The last line of the program issues the DOS EXEC command, which executes the text file. You now have the menu program SAVEd on disk and displayed on the screen, ready to make any modifications you might wish.

Menu Generator

```
1  REM   "**********************"
2  REM   "*    MENU GENERATOR    *"
6  REM   "**********************"
10  REM
20  TEXT : HOME : HTAB 13: PRINT "MENU
     GENERATOR"
30  VTAB 2: HTAB 1: FOR I = 1 TO 40: PRINT
     "-";: NEXT I
40  DIM OP$(12)
50 D$ =  CHR$ (13) +  CHR$ (4):Q$ =  CHR$
     (34): REM D$=CONTROL-D;Q$=QUOTE MA
     RK
100  GOTO 1000
199  REM  COMMONLY USED SUBROUTINES
200  VTAB 22: HTAB 1: CALL  - 956: VTAB
     22: RETURN
250  VTAB V: HTAB 1: CALL  - 868: VTAB
     V: RETURN
999  REM
1000  REM  ENTER NUMBER OF OPTIONS DESI
     RED ON THE MENU
1001  REM
1010  GOSUB 200: HTAB 1: PRINT "YOU MAY
      CHOOSE UP TO 12 MENU OPTIONS OF":
      PRINT "UP TO 30 CHARACTERS EACH I
     N LENGTH"
1020 V = 5: GOSUB 250: INPUT "ENTER NUM
     BER OF MENU OPTIONS (1-12) ";N$
1030 E = 0: FOR I = 1 TO  LEN (N$):CH$ =
      MID$ (N$,I,1):CH =  ASC (CH$): IF
     CH < 48 OR CH > 57 THEN E = 1
1040  NEXT I
1050  ON E GOTO 1060,1020
1054  REM
1055  REM  ENTER THE MENU OPTIONS
1056  REM
1060 N =  VAL (N$): IF N < 1 OR N > 12 THEN
      GOTO 1020
1070 V = 7: GOSUB 250: PRINT "ENTER OPT
     ION NAME NEXT TO THE NUMBER"
1075  GOSUB 200: PRINT "NO COMMAS, COLO
     NS OR QUOTE MARKS IN THE": PRINT "
     MENU OPTIONS PLEASE"
1080  FOR I = 1 TO N
1090 V = 7 + I: GOSUB 250: PRINT I;: INPUT
     " ";OP$(I)
1100  IF  LEN (OP$(I)) > 30 THEN  GOTO
      1090
1200  NEXT I
1210 V = 20: GOSUB 250: PRINT  TAB( 10)
     "ANY CHANGES (Y/N) ";: GET A$: IF
     A$ = "N" THEN  GOTO 1400
1220  IF A$ <  > "Y" THEN  GOTO 1210
1230 V = 20: GOSUB 250: PRINT  TAB( 5)"
     CHANGE NO. OF OPTIONS (Y/N) ";: GET
```

```
       A$: IF A$ = "N" THEN  GOTO 1300
1240  IF A$ <  > "Y" THEN  GOTO 1230
1250  GOTO 1000
1300 V = 20: GOSUB 250: HTAB 5: INPUT "
      ENTER OPTION NUMBER TO CHANGE ";NN
      $: IF  LEN (NN$) > 2 THEN  GOTO 13
      00
1310 E = 0: FOR I = 1 TO  LEN (NN$):CH$
       =  MID$ (NN$,I,1): IF  ASC (CH$) <
      48 OR  ASC (CH$) > 57 THEN E = 1
1320  NEXT I
1330  ON E GOTO 1340,1300
1340 NN =  VAL (NN$): IF NN < 1 OR NN >
      N THEN  GOTO 1300
1350 V = 7 + NN: GOSUB 250: PRINT NN;: INP
      UT" ";OP$(NN)
1360  IF  LEN (OP$(NN)) > 30 THEN  GOTO
      1350
1370  GOTO 1210
1400  FOR I = 1 TO N:L1 =  LEN (OP$(I))
      : IF L1 > L THEN L = L1
1410  NEXT I
1420  POKE 34,24
2000  REM  BUILD TEXT FILE
2010  PRINT D$;"OPEN MENU-FILE"
2020  PRINT D$;"DELETE MENU-FILE"
2030  PRINT D$;"OPEN MENU-FILE"
2040  PRINT D$;"WRITE MENU-FILE"
2045  PRINT "FP"
2047  PRINT "POKE 34,24"
2050  PRINT "10 REM MENU PROGRAM"
2053  PRINT "12 TEXT:HOME"
2055  PRINT "15 VTAB 1:FOR I=1 TO 40:PR
     INT "Q$"-"Q$";:NEXT I"
2060  PRINT "20 VTAB 2:PRINT TAB(18) "Q
     $"MENU"Q$
2070  PRINT "30 VTAB 3:FOR I=1 TO 40:PR
     INT "Q$"-"Q$";:NEXT I"
2080  PRINT "40 N=";N;":DIM OP$(";N;")"
2090  PRINT "50 FOR I=1 TO N:READ OP$(I
     ):NEXT I"
2100  PRINT "60 GOTO 80"
2110  PRINT "70 DATA ";: FOR I = 1 TO N
     - 1: PRINT OP$(I);",";: NEXT I: PRINT
     OP$(N)
2120  PRINT "80 FOR I=1 TO N:L1=LEN(OP$
     (I)):IF L1>L THEN L=L1"
2130  PRINT "90 NEXT I"
2140  PRINT "100 V=(INT(24-N)/2)-1:H=IN
     T((40-(L+4))/2)"
2160  PRINT "110 IF N<=9 THEN FOR I=1 T
     O N:VTAB V+I:PRINT TAB(H) "Q$"<"Q$
     "I"Q$"> "Q$";OP$(I):NEXT:GOTO 120"
2170  PRINT "112 FOR I=1 TO 9:VTAB V+I:
     PRINT TAB(H) "Q$"<"Q$"I"Q$"> "Q$";
     OP$(I):NEXT I"
2175  PRINT "114 FOR I=10 TO N:VTAB V+I
     :PRINT TAB(H-1) "Q$"<"Q$"I"Q$"> "Q
     $";OP$(I):NEXT I"
2180  PRINT "120 VTAB 23:HTAB 8: INPUT
     "Q$"ENTER SELECTION : "Q$";N$"
2190  PRINT "130 IF LEN(N$)>2 OR LEN(N$
     )<1 THEN GOTO 120"
2200  PRINT "140 E=0:FOR I=1 TO LEN(N$)
     :CH$=MID$(N$,I,1):IF ASC(CH$)<48 O
     R ASC(CH$)>59 THEN E=1"
2210  PRINT "150 NEXT I:IF E=1 THEN GOT
     O 120"
2220  PRINT "160 N1=VAL(N$):IF N1<1 OR
     N1>N THEN GOTO 120"
3010  PRINT "SAVE MENU-PROGRAM"
3030  PRINT "RUN"
3040  PRINT D$;"CLOSE"
3050  PRINT D$;"EXEC MENU-FILE"
```

©

Variable Lister

E. A. Cottrell

This utility lists all your program variables in order, including variable type (simple, array). It's a helpful tool, especially for checking long programs and writing documentation. For the VIC and 64.

There are two types of variables, simple and array, and three categories in each type, floating point numeric, integer numeric, and string. All of these variables are stored in the VIC and 64 immediately after the BASIC program.

The simple variables are stored below the arrays starting at the address pointed to by memory locations 45 and 46 (see box). Each of these simple variables occupies seven bytes of memory. The first two bytes contain the first two characters (in ASCII) of the name of the variable, with coding to indicate which type of variable it is. This coding is accomplished by adding 128 to both characters if it is an integer variable and by adding 128 to the second character if it is a string variable. No coding indicates a floating point variable. The remaining bytes in numeric variables contain the value of the variable. In the case of string variables, the remaining bytes contain the length of the string and the location at the top of memory which contains the first character of the string.

Arrays are quite different in that the length of the variable is determined by the number of elements in the array. The information which must be stored for an array variable includes the name of the variable, which is coded the same as for a simple variable, a pointer to the location of the next variable, the number of dimensions in the array, and the number of elements in the array.

In addition, the value of each element in the case of numeric arrays, or the pointer to the string and its length for string arrays, must be stored. As you can see, array variables can eat up a lot of memory in a hurry. It is best to use the lowest possible number of elements in your arrays. If you do not specify the size of an array, the computer will set it at ten elements. If you need less than ten, you will save a minimum of five bytes per element if you establish the size of the array with a DIMension statement. Although a simple integer variable takes up the same amount of memory as a simple floating point variable, three bytes per element can be saved if you use integer instead of floating point variables in arrays.

LOADing The Lister

"Variable Lister" is a machine language (ML) program which is loaded by POKEs using a BASIC program, thus eliminating the need for an assembler. The ML is automatically loaded into the top of memory and protected from your BASIC program. Before you RUN the program, be sure to SAVE a copy since it self-destructs after it is run. When the machine language is loaded, the loader program will give you the location to SYS to when you want to list your variables. For example, with 16K of expansion memory plugged into your VIC, you would type SYS 24320 in order to list your variables. The program will then list the simple variables in the order of appearance in the program, with indicators of their type. Next the array variables will be listed with proper indicators.

Variable Lister is especially useful when you write programs with many variables and have to find new names. It is also valuable for documenting programs when they are completed.

The variables are listed across the screen to prevent them from scrolling out of view. If you have a printer, the following changes may be made to give you a listing which may be easier to read.

```
160 IF PA <> 33632 THEN PRINT "DATA ERROR
    ":END
```

Address Pointers

Now and then you'll see a reference to "pointers" within the computer's memory. These are two-byte long numbers, usually located in the first 256 memory cells of the computer, which hold an important address.

Things change while a program is running or being written. For example, if you add a line to a BASIC program, you've expanded the amount of space that the program is taking up in RAM memory. Obviously, when you go to SAVE the program, the computer has to know where the BASIC program ends. So, it keeps track of the "current top of BASIC program" in a pointer. This pointer is located (in the VIC and 64) in addresses 45 and 46. The number held in cell 46 is multiplied by 256 and then added to the number in cell 45. To see at which address in RAM memory your current BASIC program ends, you can type: ? PEEK (45) + PEEK (46) * 256.

There are a number of other pointers as well, including "limit of memory," "start of arrays," "string storage," and "start of BASIC." The locations of these pointers are listed in *memory maps* for each computer which have been published in COMPUTE! and in various COMPUTE! Books. They are also frequently available from user groups. There are some interesting things you can do by manipulating these pointers with POKEs. For one thing, you could fool the computer into reserving space for programs in odd places, or even partitioning memory so that two independent BASIC programs could run simultaneously. In any event, pointers hold information essential to the computer, and their values can be accessed using the formula above.

```
260 DATA 32,210,255,169,13,32,210
420 DATA 41,32,210,255,169,13,32
```

To send the list to your printer, simply OPEN a file to your printer:

```
OPEN1,4 :CMD1:SYSXXXXX
```

The BASIC program for which you wish to list variables must be RUN before you give the SYS to start the Lister. This is because the variables are not set up in memory until a program is RUN. If you're a VIC owner, this program works well in conjunction with "VIC Searcher" (COMPUTE!, February 1983). First, list the variables with Variable Lister, then find the lines on which they appear with the Searcher. Both of these programs may be loaded together. Remember that if you use the Searcher program, remove line 0 and RUN the program before using the Lister.

Variable Lister

```
120 ME= PEEK(55) + 256 * PEEK(56)
130 VS= ME - 256: PA= 0
140 POKE 56, PEEK(56) -1
150 FORI= VS TO VS + 240: READ A: POKE I,
    A: PA= PA + A: NEXT
160 IF PA <> 33670 THEN PRINT "DATA ERROR
    ": END
170 PRINT "SYS" VS "TO START": NEW
180 DATA 165,45,197,47,240,93,133
190 DATA 253,165,46,133,254,160,0
200 DATA 169,0,141,61,3,177,253
210 DATA 41,128,208,60,177,253,41
220 DATA 127,32,210,255,200,173,61
230 DATA 3,201,0,208,6,177,253
240 DATA 41,128,208,46,177,253,41
250 DATA 127,32,210,255,173,61,3
260 DATA 32,210,255,169,32,32,210
270 DATA 255,152,24,105,6,144,5
280 DATA 164,254,200,132,254,168,101
290 DATA 253,197,47,240,17,208,186
300 DATA 96,169,37,141,61,3,208
310 DATA 189,169,36,141,61,3,208
320 DATA 203,165,49,197,47,240,114
330 DATA 165,47,133,253,165,48,133
340 DATA 254,160,0,169,0,141,61
350 DATA 3,177,253,240,216,41,128
360 DATA 208,77,177,253,41,127,32
370 DATA 210,255,200,173,61,3,201
380 DATA 0,208,6,177,253,41,128
390 DATA 208,63,177,253,41,127,32
400 DATA 210,255,173,61,3,32,210
410 DATA 255,169,40,32,210,255,169
420 DATA 41,32,210,255,169,32,32
430 DATA 210,255,200,177,253,24,101
440 DATA 253,197,49,240,39,177,253
450 DATA 24,101,253,170,200,177,253
460 DATA 101,254,133,254,134,253,208
470 DATA 165,96,169,37,141,61,3
480 DATA 208,172,169,36,141,61,3
490 DATA 208,186,165,48,197,50,208
500 DATA 136,96,200,234,177,253,101
510 DATA 254,197,50,240,224,16,222
520 DATA 136,208,202
```

©

HERE'S SOME GREAT GIFT IDEAS FOR YOUR COMMODORE-64 OR VIC-20 Computer

Every ABACUS SOFTWARE Package includes an attractive outer carton, a manual and a disk or cassette.

YOU CAN COUNT ON

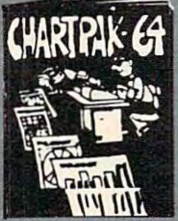
Abacus Software

ULTRABASIC 64
- TURTLE, hires, multicolor and sprite graphics
- Sound and effects
- Hardcopy
- 50 Powerful Commands
- Includes demos, tutorial, manual
- **$39.95 FOR TAPE**
- **$42.95 FOR DISK**

SUPER DISK UTILITY 64
- Single Disk Copy (3 types) or Append
- Dump or Modify Any Disk Sector (Has Scrolling)
- Directory Display with scrolling
- BAM Display
- Printer Output
- Fast Machine Code
- **$22.95 DISK ONLY**

SYNTHY 64
- Full featured music and sound synthesizer
- Easy entry and editing of notes and commands
- Control ASDR, filters, waveforms etc.
- Includes sample music and manual
- **$29.95 FOR TAPE**
- **$32.95 FOR DISK**

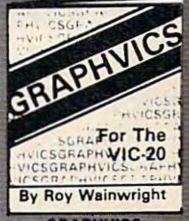
CHECKBOOK MANAGER 64
- Menu/Function Key makes it very easy to use
- Complete editing
- Full checkbook balancing
- Printer Output
- **$22.95 DISK ONLY**



CHARTPAK 64
- Create Pie, Bar and Line charts in high resolution
- Enter, edit, save and recall to/from disc
- Choose any of 8 chart formats and design charts interactively
- Produce hard copy onto 1515, 1525 or Epson printer
- **$42.95DISK ONLY**

SKIER 64
- Arcade type game
- Joystick or keyboard control
- 3 different games
- 3 different levels
- **$14.95 FOR TAPE**
- **$17.95 FOR DISK**

TINY FORTH
- Easily extendable language
- Based on fig-FORTH
- Commodore 64 or Vic-20 w/8k exp.
- **$19.95 FOR TAPE**
- **$22.95 FOR DISK**



GRAPHVICS
- Hires and multicolor graphics
- Fast ploting at machine language speed
- Two screens-graphics and text
- Save screen to tape/disk.
- VIC-20 w/3k or 8k exp.
- **$19.95 FOR TAPE**
- **$22.95 FOR DISK**

DISTRIBUTORS

Great Britain: ADAMSOFT, 18 Norwich Ave., Rochdale, Lancs.

West Germany: DATA BECKER, Merowingerstr 30, 4000 Dusseldorf, 0211/312085

Great Britain: CCI Software, 167 Great Portland St., London WI, 01-636-6354

Sweden: TIAL TRADING, PO 516, 34300 Almhult, 476-12304

Canada East: KING MICROWARE LTD, 5950 Cote des Neiges, Montreal, Quebec H3S 1Z6, 514/737-9335

Australia: CW ELECTRONICS, 416 Logan Road, Brisbane, Queens., 07-397-0808

Canada West: L.S.I. Distributors Ltd., 810 W Broadway #163, Vancouver, BC V5Z 4C9, 604/733-0211

New Zealand: VISCOUNT ELECTRONICS, 306-308 Church Street, Palmerston North, 63-86-696

TO ORDER NOW: Write Abacus Software P.O. Box 7211 Grand Rapids, Mich. 49510

For postage & handling, add $1.50 (U.S. and Canada), add $3.00 for foreign. Make payment in U.S. dollars by check, money order or charge card. (Michigan Residents add 4% sales tax).

FOR QUICK SERVICE PHONE 616/241-5510

NAME__________

ADDRESS__________

CITY__________ STATE______ ZIP______

CHARGE CARD__________ NUMBER__________

FREE CATALOG Ask for a listing of other Abacus Software for Commodore-64 or Vic-20

Atari Polycopy

Edward Boyer

This convenient utility program lets you copy at least 16 selected files from one or more disks – all in a single pass. It's also useful for reformatting a disk with multiple files.

"Polycopy" is a disk file copy utility designed for users with only one disk drive; many Atari owners will therefore find this program useful. Using a minimum DOS on a 48K system, it can copy over 225 sectors in a single pass. With the exception of DOS.SYS (which it never copies), an entire disk can be copied in only three passes. Polycopy is expecially handy if you must reformat a disk with multiple files. You will also find it useful when you're copying programs or files.

What makes this program different from Atari DOS disk and file copy is its ability to selectively copy more than one file in a single pass. Polycopy uses machine language routines to perform the disk I/O. These routines are far more efficient than the BASIC GET and PUT instructions. In an effort to save all available RAM, Polycopy deletes its initialization lines once initialization is complete. It will become a valuable part of your disk library.

Program Operation

After initialization, you will be asked to insert a FROM disk; insert the disk and press any key. Each filename on the disk will be displayed. Type a C if you want the file copied, a Q if you're finished with this disk, or any other key to bypass this file. The files will continue until all have been displayed or the maximum of 16 files has been selected. You can increase or decrease the limit of 16 by changing the value of DSN in line 395.

You will then be asked for another FROM disk if fewer than 16 files have been requested. If you respond with a Y, the process will continue, allowing you to select more files from a different disk.

If you respond N for additional disks, you will be asked to review the filenames selected (all will still be on the screen). If you're in agreement, press Y and the copying will proceed. If you press any other key, the copy is aborted.

As the program runs, you will be prompted to insert FROM and TO disks at the appropriate times. Also, progress reports will be given to inform you that all is proceeding well.

Error Handling

During the loading pass, if Polycopy can't find a file on the disk in use, it will ask you if you want to try another disk. If you respond Y, it will prompt you to insert a new FROM disk. If you respond N, it will skip to the next requested file, thus ignoring the file it couldn't find.

If a full disk condition occurs while writing on the TO disk, you will be told of the condition and asked if you want to try another disk. If you answer Y, the partial file will be erased from the current disk, and you will be requested to insert a new TO disk. It is extremely important that you not remove the current disk or insert a new one until requested, because Polycopy erases the partial file from the current disk *before* requesting you to change it.

If the program uses all the available RAM before it finishes reading a file, it will write out the current portion to the new disk then retrieve the remainder on the next pass. This will be noted by the presence of a slash (/) after the size. If a full disk condition occurs during the writing of a split file, you will be asked to insert the FROM disk (the one with the file that was originally split). It will then recopy the file in its entirety, before requesting a new TO disk.

It is important that you respond to the questions and prompts only when requested; don't change disks until you are asked to. Impatience here can lead to irrecoverable files.

Any other I/O errors encountered will result in the skipping of the file currently being processed.

You may abort Polycopy anytime by pressing the ESCAPE key. It's important to use this rather than BREAK or SYSTEM RESET since it provides an orderly (although not always immediate) termination, thus insuring the integrity of your files and no misallocated sectors on the disk.

Program Explanation

Lines 15-40 set up the table of files to be copied.

Lines 45-125 load files into buffer, check for errors and full buffer.

Lines 130-225 write files on new disk from buffer, check for errors and full disk.

Line 230 places the next filename to be read or written in DSN$.

Lines 250-320 build table of filenames to be copied by displaying filenames found in disk directory.

Lines 325-370 – miscellaneous prompts and input routines.

Lines 375-455 define variables, open the keyboard, and POKE the machine language routine into the printer buffer.

Lines 450-455 delete lines 375-450 to make additional room.

Line 460 allocates most of available RAM as the copy buffer.

Note: Line 445 bypasses the routine to delete lines 375 through 450 to allow you to test Polycopy conveniently. This line should be removed from your operational version to allow the largest possible buffer.

If you don't want to type in Polycopy, I will make a copy for you. Send a blank formatted disk, a postage-paid return mailer, and $3 to:

Edward Boyer
81 Sequoia Drive
Coram, NY 11727

Polycopy

```
10 GOTO 375
15 GRAPHICS K0:PRINT "POLYCOPY - ATA
   RI version 2.0"
20 PRINT "(Space for ";INT(BUFF/125)
   ;" sectors)":PRINT :IP=-K1
25 GOSUB 250:IF IP=DSN THEN 40
30 IF IP<DSN THEN GOSUB 355:IF Z=YES
    THEN 25
35 IF IP<K1 THEN 235
40 MAX=IP-K1:PRINT "Type 'Y' if o.k.
    ";:GOSUB 335:PRINT :IF Z<>YES TH
   EN 235
45 IP=K0:OP=K0:SPLIT=HI:APND=HI:GOTO
    55
50 PRINT :GOSUB 325
55 ADDR=ADR(Y$):ROOM=BUFF
60 IF IP>MAX THEN IP=IP-K1:GOSUB 145
   :GOTO 240
65 Y=IP:GOSUB 230:TRAP 80:OPEN #K1,K
   4,K0,DSN$:TRAP TOFF
70 IF SPLIT<>IP THEN PRINT "Loading
   ";DSN$;:GOTO 100
75 TRAP 80:POINT #K1,SEC,BYTE:TRAP T
   OFF:APND=IP:PRINT "Contin'g ";DSN
   $;:GOTO 100
80 TRAP TOFF:Z=PEEK(195):CLOSE #K1:I
   F Z<>170 THEN 110
85 PRINT :PRINT DSN$;" not found,":P
   RINT "..do you want to try anothe
   r disk?";
90 GOSUB 350:PRINT :IF Z=YES THEN PR
   INT "Insert new disk";:GOSUB 330:
   GOTO 65
95 X(IP,K0)=K0:IP=IP+K1:GOTO 60
100 X(IP,K0)=ADDR
105 SIZE=USR(CIO,K1,7,ADDR,ROOM):Z=P
    EEK(851):IF Z<128 OR Z=136 THEN
    115
110 CLOSE #K1:PRINT :GOSUB 370:GOTO
    95
115 PRINT " size=";SIZE;:X(IP,K1)=SI
    ZE:IF Z=136 THEN 125
120 PRINT "/";:NOTE #K1,SEC,BYTE:SPL
    IT=IP
125 PRINT :CLOSE #K1
130 ADDR=ADDR+SIZE:ROOM=ROOM-SIZE:IF
     ROOM>K0 THEN IP=IP+K1:GOTO 60
135 GOSUB 145:IF SPLIT<>IP THEN IP=I
    P+K1:IF IP>MAX THEN 60
140 GOTO 50
145 IF ROOM=BUFF THEN RETURN
150 PRINT :PRINT "Insert 'to' disk";
    :GOSUB 330
155 ADDR=X(OP,K0):SIZE=X(OP,K1):IF A
    DDR=K0 THEN 220
160 Y=OP:GOSUB 230:Z=8:IF APND=OP TH
    EN Z=Z+K1
165 TRAP 185:OPEN #K2,Z,K0,DSN$:TRAP
     TOFF:IF APND=OP THEN PRINT "App
    end'g ";:GOTO 175
170 PRINT "Writing ";
175 PRINT DSN$;" size=";SIZE
180 Z=USR(CIO,K2,11,ADDR,SIZE)
185 Z=PEEK(867)
190 TRAP 190:CLOSE #K2:TRAP TOFF:IF
    Z<128 THEN 220
195 IF Z<>162 THEN GOSUB 370:GOTO 22
    0
200 PRINT "Disk full, try another? ▮
    answer first!";:GOSUB 335:IF Z<>
    YES THEN 235
205 TRAP 210:XIO 33,#K2,K0,K0,DSN$
210 TRAP TOFF:IF APND<>OP THEN PRINT
     "Insert new disk";:GOSUB 330:GO
    TO 155
215 IP=OP:SPLIT=HI:APND=HI:POP :GOTO
     50
220 IF OP<IP THEN OP=OP+K1:GOTO 155
225 RETURN
230 DSN$="D:":DSN$(K3)=X$(Y*K12+K1,Y
    *K12+K12):GOTO 360
235 PRINT :PRINT " User aborted!! "
240 GOSUB 355:IF Z=YES THEN 15
245 CLR :END
250 GOSUB 325:CLOSE #K3:OPEN #K3,6,K
    0,"D:*.*":IF IP<K0 THEN IP=K0:X$
    ="":PRINT " C-copy, Q-quit, any
    key to skip "
255 INPUT #K3,Y$:IF LEN(Y$)<17 THEN
    320
```

```
260 DSN$="":FOR Z=K3 TO 13:IF Z=11 T
    HEN DSN$(LEN(DSN$)+K1)="."
265{3 SPACES}IF Y$(Z,Z)=" " THEN 275
270{3 SPACES}DSN$(LEN(DSN$)+K1)=Y$(Z
    ,Z)
275{3 SPACES}NEXT Z:IF DSN$="DOS.SYS
    " THEN 255
280 IF LEN(DSN$)<K12 THEN DSN$(LEN(D
    SN$)+K1,K12)="{11 SPACES}"
285 SEC=K0:TRAP 290:SEC=VAL(Y$(15,17
    ))
290 TRAP TOFF:PRINT IP+K1;:POKE COL,
    5:PRINT DSN$;:POKE COL,22:PRINT
    SEC;:POKE COL,28:PRINT "?";:GOSU
    B 340
295 IF Z=67 THEN PRINT CHR$(30);"C":
    GOTO 310
300 PRINT CHR$(ERASE);:IF Z=81 THEN
    320
305 GOTO 255
310 X$(LEN(X$)+K1)=DSN$:IP=IP+K1:IF
    IP=DSN THEN 320
315 GOTO 255
320 CLOSE #K3:RETURN
325 PRINT "Insert 'from' disk";
330 PRINT ", press any key!";
335 GOSUB 340:PRINT CHR$(ERASE);:RET
    URN
340 GET #K4,Z:IF Z=27 THEN POP :GOTO
     235
345 RETURN
350 GOSUB 340:PRINT :RETURN
355 PRINT :PRINT "Any more files?";:
    GOTO 335
360 Z=PEEK(764):POKE 764,HI-K1:IF Z=
    28 THEN 235
365 RETURN
370 PRINT "[";Z;"] I/O error on ";DS
    N$:PRINT "...skipping to next fi
    le!":PRINT :RETURN
375 REM Delete lines beginning here
    !
380 GRAPHICS K0:POSITION 13,12:PRINT
     " PLEASE WAIT!";
385 K0=0:K1=1:K2=K1+K1:K3=K2+K1:K4=K
    3+K1:K12=K4*K3:HI=256:TOFF=40000
390 YES=89:LET ERASE=156:COL=YES-K4:
    OPEN #K4,K4,K0,"K:"
395 DSN=16:DIM DSN$(16),X$(DSN*K12),
    X(DSN-K1,K1)
400 CIO=960:FOR Y=K0 TO 42:READ Z:PO
    KE CIO+Y,Z:NEXT Y:POKE 709,PEEK(
    710)
405 DATA 104,104,104,10,10,10,10,170
410 DATA 104,104,157,66,3,104,157,69
415 DATA 3,104,157,68,3,104,157,73
420 DATA 3,104,157,72,3,32,86,228
425 DATA 189,72,3,133,212,189,73,3
430 DATA 133,213,96
435 MAX=842:APND=35
440 IP=375:OP=450:REM These are the
    'from' and 'to' delete line num
    bers!
445 GOTO 460:REM Remove this line t
    o allow deletes!
450 PRINT CHR$(125):PRINT :FOR Z=IP
    TO OP STEP 5:PRINT Z:NEXT Z:PRIN
    T "CONT":POSITION K0,K0:POKE MAX
    ,13
455 STOP
460 POKE MAX,K12:BUFF=FRE(K0)-APND:D
    IM Y$(BUFF):BUFF=BUFF-K3:GOTO 15
```

PEEK And PRINT For The VIC-20

Carolyn D. Bellah

These two programs let you design and display characters four times normal size. You can store up to seven of these larger characters and recall them later for screen displays or a printout.

The two programs here allow you to design large custom characters, twice as high and twice as wide as regular characters, save them, and print them.

Program 1 sets up a programmable character grid in which you move your cursor to a desired location on the grid and hit any letter key to print at that location.

Several options are available in Program 1. There is a color (1 to 8) choice, an option to save or erase your created character, and an option to draw another. Seven characters can be stored in a protected area of RAM and can be recalled, printed, and listed in sequence by using Program 2.

Another useful feature is that the program will display the decimal PEEK values (the numeric values that are used for DATA statements) that represent your finished character. You can use Program 1 without Program 2. Program 1 isn't a long program, but it does use most of the available memory in the unexpanded VIC. For this reason, REM statements are not included. When typing in this program, do not use unnecessary spaces.

When you finish designing with Program 1, type NEW (be sure to save a copy first), and enter Program 2. This second program allows you to examine memory to see the decimal values for the data created by Program 1, and to print the values (to screen and graphics printer) along with your created character.

Program 2 also allows you to print a reversed image of your created character and the corresponding decimal PEEK values.

Here's an explanation of the programs.

Program 1

Lines	
8-9	Reset top of memory pointers; copy 32 characters into protected RAM; DIMension array to recall marked grid for revision; set variables for characters and display; leave a clean slate to draw on.
10	Create string to draw grid (shifted @ key) and strings for positioning characters, inputs, etc., throughout program.
11	Array from which bit values are read.
14-16	Draw grid and set up display.
17-30	Keyboard controls.
31-33	Read grid, store values, display design.
34-36	Offer options and, with 59-63, show figure in chosen color moving around screen.
48-49	Print figure and values. The last number printed is the next address to be POKEd. If this is 7672, all available characters have been programmed.
49-54	Offer options.
64-70	Redraw marked grid for revision after display in color and motion.

Program 2

Lines	
10-50	Set up display of design; print DATA values; show first and last address PEEKed.
55	Get a name for the design being displayed.
60-140	Set up display of character reversed horizontally.
170	Increment CHR$ for next design.
200-250	PEEK registers; reverse values; list them in proper order for program use.
600-620	Restructure the data for use by the screen dump routine.
60000-60070	Print a copy of the character design on a VIC 1515 or 1525 graphics printer.

(Program 2 is also handy for checking out the contents of other locations. Just change the value of PC in line 10 from 7448 to whatever address interests you.)

Program 1: Character Creator

```
2 PRINT"{CLR}"SPC(5)"{DOWN}{RED}PROGRAMMA
  BLE":PRINTSPC(4)"CHARACTER GRID"
3 PRINT"{2 DOWN}{BLK}BEGIN AT TOP LEFT. "
  ;
4 PRINT"THECURSOR{2 SPACES}CONTROLS
  {2 SPACES}AND SPACE BAR WILL BEHAVE NOR
  MALLY. ALPHANUMERICKEYS ";
5 PRINT"MARK THE GRID.":PRINT"{2 DOWN}HIT
   F1 TO SEE DESIGNEDCHARACTER.
6 PRINT"{GRN}{2 DOWN}{RIGHT}HIT RETURN TO
   BEGIN
7 GETM$:IFM$<>CHR$(13)THEN7
```

ATARI

Computers for people.

CORDLESS TELEPHONES

up to 700 ft. range

from $69.75

Educator....................$109.75
Programmer..................$ 51.75
Entertainer..................$ 63.75
Bookeeper....................$164.75

600XL $$$$$149.00
800 XL... $CALL$
1200 XL. $499.00
1400 XL . $CALL$
1450 XL . $CALL$
1025 Printer $399.00
1020 Color Printer . $245.00
1027 Printer $299.00
1010 Recorder.......$75.00
410 Recorder $75.00
810 Disk Drive......$399.00
1050 Disk Drive $335.00
850 Interface........$CALL$

PARKER BROTHERS
Tutankham R............$33.75
Super Cobra R $33.75
Astro Chase R............$33.75
Frogger R $33.75
QBert R $33.75
Popeye R.................$33.75
Risk R $42.75
Chess R $42.75

SPINNAKER
Story Machine R $26.75
Face Maker R $24.75
Kinderomp R.............$20.75
Fraction Fever R $24.75
Delta Drawing R.........$26.75

SSI
Battle of Shilo C/D.........$26.75
Tigers in the Snow C/D....$26.75
Battle for Normandy C/D ..$26.75
Knights of the Desert C/D .$26.75
Cosmic Balance C/D $26.75

ON-LINE
Frogger $24.95
Wizard & Prin $26.95

ROKLAN
Wizard of War.............$29.75
Gorf $29.75
Delux Invader $27.95

BIG 5
Miner 2049................$32.75

BUSINESS
Visicalc $159.75
Letter Perfect......... $89.75
Letter Perfect......... $89.75
Data Perfect........... $89.75
Text Wizzard $49.75
Spell Wizzard $64.75
File Manager...........$69.75
Home File Mgr..........$69.75
Bookeeper............ $119.75
C.R.I.S. $199.75
Atari Word Pro $109.75
Tax Advantage $35.75
Home Accountant $59.75
Bank Street W $49.75
ATARI WRITER...........$74.75

ADVENTURE
Preppie................. $19.95
Preppie 2............... $19.95
Diskey.................. $39.95
Sea Dragon $25.50
Stratos $25.00
Treasure Quest $13.50
Series 1-12..............$15.95
Saga Series 1-6 ....$24.95
Stone of Sisyphus $29.95
Eliminator.............. $18.95

apple

Apple is a Registered Trademark of Apple Computer Inc.

SSI
Battle of Shilo $26.75
Tigers in the Snow.... $26.75
Cosmic Balance $26.75
Knights of the Desert . $26.75
Battle for Normandy .. $26.75
Germany 1985 .... $36.75

RANA DISK DRIVES
Elite 1............ $295.00
Elite 2............ $449.00
Elite 3............ $559.00

MICRO-SCI
A2 ....call
A40.......................call
A70.......................call

MUSE
Castle Wolfenstein $20.75
Caverns of Frietag $20.75
Robot War.............$26.75

CONTINENTAL
Home Accountant $51.75
Book of Apple Software . $16.75

BRODERBUND
Bank Street Writer........$44.75
AE $24.75
Apple Panic $21.75
Choplifter $24.75
David's Midnight..........$24.75

SPINNAKER
Kindercomp............. $21.75
Story Machine $23.75
FaceMaker.............. $23.75
Snooper Trooper......... $29.75
Delta Drawing........... $34.75

EPYX
Temple of Apshai.......... $26.95
Star Warrior.............. $26.95
Crush, Crumble & Chomp . $22.75

ADVENTURE
Saga#1 Adventureland ...$29.95
Saga#2 Pirate Adventure.$29.95
Saga#3 Secret Mission...$29.95
Stone of Sisyphus $24.95

ALIEN GROUP
Atari Voice Box $99.00
Apple Voice Box......... $129.00

commodore

VIC 64...................$AVE
1541 Disk Drive ....Call
1525 Printer...............Call
1530 Datasette $64.00
1110 8k Ram...........$53.75
1211 Super Expander .$53.75
1212 Programmers Ad.$44.75
1213 Vicmon............$44.75
Vic 20 dust cover........$6.99
Vic 64 dust cover........$6.99

MICROSPEC 64
Inventory Mgr..........$79.75
General Ledger $79.75
Payroll Mgr.............$79.75
Accounts Rec $79.75
Accounts Pay $70.75
Data Base $69.75
Checkbook Bal $64.75

HES 64
64Forth R $55.75
Hesmon R....... $29.75
Turtle Graphics R......$49.75
Heswriter R $38.75
Gridrunner R $29.75
Attack of Mut Cam R...$34.75
Turtle Tutor R..........$29.75
Turtle Trainer R........$29.75
Paint Brush R $23.75
Benji Space Rescue D .$29.75
Home Manager C/D ...$39.75
Time Money Mgr D $55.75
OmniCalc D $79.75
Sword Point D..........$24.75

EPYX 64
Temple of Apshai.......$28.00
Upper Reaches of A $15.00
Crush Crumble & C $23.00
Jumpman $28.00

CARDCO
Cardprinter / LQ1....... $499.00
Cardprint DM1.......... $109.00
5 Slot Expansion 64 ..... $54.00
64 Write NOW............ $39.00
64 Mail NOW............. $29.00
20 Write NOW............ $29.00
64 Keypad $29.00
Universal Cass. Int....... $29.75
Printer Utility........... $19.75
6 Slot Expansion......... $79.95
3 Slot Expansion......... $24.95
Vic 20/64 Printer int.....$59.95

BRODERBUND 64
Serpentine R............. $26.75
Choplifter R.............. $32.75
Seafox R $26.75

PARKER 20
Frogger (ROM) $33.75
QBert (ROM) $33.75
Tutankham (rom) $33.75

SPINNAKER 64
Kindercomp...............$21.75
Story Machine $23.75
Face Maker $23.75
Snooper Trooper...........$29.75
Delta Drawing.............$34.75
Shamus II c/d............. $24.95
Pinhead c/d $22.95

QUICK BROWN FOX
QBF Word Processor $49.95

LJK
Letter Perfect........... $105.00
Data Perfect.............. $95.00

ADVENTURE INTERNATIONAL
S. Adams Adventure..... $28.75

VIC-64
WORDPRO 3+.............$69.75

VIC 20
King Arthurs Heir Cass $24.75
Monster Maze Rom.........$24.75

PERCOM

HARD DISK DRIVES for

APPLE IBM-PC TRS-80*

5MEG.....................$1349.00
10MEG....................$1599.00
15MEG....................$1999.00
20MEG....................$2359.00

*Add $30.00 for TRS-80 Drives

SINGLE DRIVE AT88 S1 ... $349.00
DUAL DRIVE AT88 S2 ...... $535.00
AT88 with printer port...... $449.00
AT88 Doubler Board $129.00
DUAL HEAD 44S1 .... $575.00
DUAL DRIVE 44S2 .... $899.00
SINGLE DRIVE 40S1 .... $499.00
TI Drive $339.00

Lyco Computer Marketing & Consultants

TO ORDER CALL US

TOLL FREE **800-233-8760**

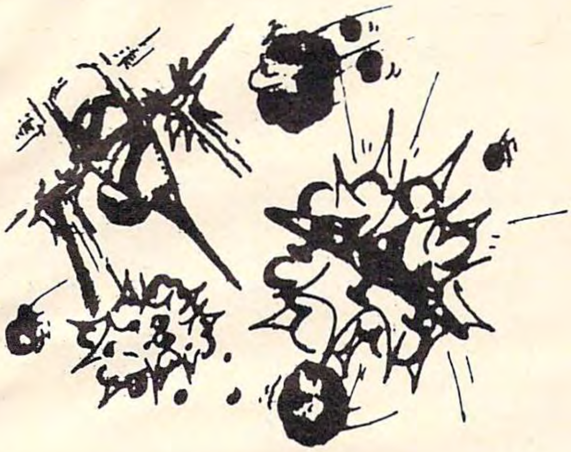
In PA 1-717-327-1824

```
8 POKE52,29:POKE56,29:CLR:FORI=7424TO7679
  :POKEI,PEEK(I+25600):NEXT:DIMCT%(255):J
  =35
9 PC=7448:FORI=7448TO7679:POKEI,Ø:NEXT
1Ø G$="{RVS}@@@@@@@@@@@@@@@@":P$="{HOME}
  {19 DOWN}":P1$="{22 RIGHT}"
11 A$(Ø)="128":A$(1)="64":A$(2)="32":A$(3
  )="16":A$(4)="8":A$(5)="4":A$(6)="2":A
  $(7)="1"
12 PRINT"{CLR}":POKE36869,255:FORG=1TO16:
  PRINT"{BLK}"G$:NEXT
13 PRINT"{HOME}{UP}{RVS}{BLK}876543218765
  4321":FORX=1TO8:PRINT"{RVS}"LEFT$(P$,1
  +X)LEFT$(P1$,16)X:NEXT
14 FORX=1TO8:PRINT"{RVS}"LEFT$(P$,9+X)LEF
  T$(P1$,16)X:NEXT
15 CR$="{2 SPACES}{DOWN}{2 LEFT}"+CHR$(J)
  +CHR$(J+1)+"{DOWN}{2 LEFT}"+CHR$(J+2)+
  CHR$(J+3)+"{DOWN}{2 LEFT}{2 SPACES}"
16 PRINT"{OFF}{HOME}{2 DOWN}"LEFT$(P1$,18
  )CR$LEFT$(P$,2Ø)"{RVS}{GRN}HIT F1 TO S
  EE DESIGN.{HOME}"
17 GETM$:IFM$=""THEN17
18 IFPEEK(211)=16ANDPEEK(21Ø)=31ANDPEEK(2
  Ø9)>9ØTHENPRINT"{HOME}{DOWN}";
19 IFPEEK(21Ø)=31ANDPEEK(2Ø9)>1ØØTHENPRIN
  T"{HOME}{DOWN}";
2Ø IFPEEK(211)=16THENPRINT:GOTO17
21 IFM$<>CHR$(2Ø)ANDM$<>CHR$(148)THEN22
22 IFM$=CHR$(13)THENPRINTCHR$(13);:GOTO17
23 IFM$=CHR$(17)THENPRINTCHR$(17);:GOTO17
24 IFM$=CHR$(29)THENPRINTCHR$(29);:GOTO17
25 IFM$=CHR$(145)THENPRINTCHR$(145);:GOTO
  17
26 IFM$=CHR$(157)ANDPOS(M$)<>ØTHENPRINTCH
  R$(157);:GOTO17
27 IFM$=CHR$(32)THENPRINT"{RVS}{BLK}"CHR$
  (186);:GOTO17
28 IFM$=CHR$(133)THENPRINTCHR$(133):GOTO3
  1
29 PRINT"{RVS}{BLU}"CHR$(166);
3Ø GOTO17
31 B=Ø:L=77Ø2:FORY=1TO2:FORZ=LTOL+154STEP
  22:D=Ø:C=Ø:GOSUB56:PC=PC+1:NEXT:L=L+8:
  NEXT
32 L=7878:FORY=1TO2:FORZ=LTOL+154STEP22:D
  =Ø:C=Ø:GOSUB56:PC=PC+1:NEXT:L=L+8:NEXT
33 FORSC=8Ø76TO8186:POKESC,32:NEXT
34 PRINTLEFT$(P$,19)"{RVS}LIKE IT? ";:INP
  UT"{RVS}Y OR N";N$
35 IFN$="Y"THEN37
36 IFN$="N"THENPRINTLEFT$(P$,19)"{RVS}CUR
  SOR IS AT TOP LEFT":PC=PC-32:PRINT"
  {HOME}":GOTO17
37 PRINT"{CLR}":INPUT"{RVS}COLOR - 1 TO 8
  ";E
38 ONEGOTO39,4Ø,41,42,43,44,45,46
39 PRINT"{BLK}":GOSUB59:GOTO47
4Ø POKE36879,11Ø:PRINT"{WHT}":GOSUB59:PRI
  NT"{BLK}":GOTO47
41 PRINT"{RED}":GOSUB59:GOTO47
42 PRINT"{CYN}":GOSUB59:GOTO47
43 PRINT"{PUR}":GOSUB59:GOTO47
44 PRINT"{GRN}":GOSUB59:GOTO47
45 PRINT"{BLU}":GOSUB59:GOTO47
46 PRINT"{YEL}":GOSUB59:GOTO47
47 POKE36879,27:PRINT"{CLR}{DOWN}
  {8 RIGHT}"CR$:PRINTLEFT$(P$,8);
48 PC=PC-32:FORCH=1TO4:FORX=PCTOPC+7:PRIN
  T"{RVS}{BLK}"PEEK(X);:PC=PC+1:NEXT:PRI
  NT:NEXT
49 PRINT"{RVS}"PC:INPUT"{RVS}WANT TO SEE
  {SPACE}IT AGAIN";N$
5Ø IFN$="Y"THEN37
51 INPUT"{RVS}REVISE IT";Q$
52 IFQ$="Y"THEN65
53 INPUT"{RVS}DRAW ANOTHER";M$
54 IFM$="Y"THENJ=J+4:GOTO1Ø
55 END
56 FORX=ØTO7:CT%(B)=PEEK(Z+X):IFPEEK(Z+X)
  =23ØTHENC=VAL(A$(X))
57 IFPEEK(Z+X)=25ØTHENC=Ø
58 D=D+C:POKEPC,D:B=B+1:NEXT:RETURN
59 PRINT"{CLR}":FORX=1TO18:PRINTLEFT$(P$,
  X)" "CR$:FORT=1TO75:NEXT:NEXT
6Ø FORX=1TO18:PRINTLEFT$(P$,19)LEFT$(P1$,
  X)"{DOWN}{LEFT} {DOWN}{LEFT} {2 UP}"CR
  $:FORT=1TO75:NEXT:NEXT
61 FORX=18TO1STEP-1:PRINTLEFT$(P$,X)LEFT$
  (P1$,18)CR$:FORT=1TO75:NEXT:NEXT
62 FORX=18TO1STEP-1:PRINTLEFT$(P$,1)LEFT$
  (P1$,X)CR$"{2 UP} {DOWN}{LEFT} ":FORT=
  1TO75:NEXT:NEXT
63 RETURN
64 B=Ø:PRINT"{CLR}"
65 B=Ø:PRINT"{CLR}"
66 FORQ=1TO8:FORX=1TO8:PRINT"{RVS}{BLU}"C
  HR$(CT%(B));:B=B+1:NEXT:PRINT:NEXT:PRI
  NT"{HOME}{DOWN}{8 RIGHT}";
67 FORQ=1TO8:FORX=1TO8:PRINT"{RVS}"CHR$(C
  T%(B));:B=B+1:NEXT:PRINT"{DOWN}
  {8 LEFT}";:NEXT:PRINT
68 PRINT"{UP}";:FORQ=1TO8:FORX=1TO8:PRINT
  "{RVS}"CHR$(CT%(B));:B=B+1:NEXT:PRINT:
  NEXT
69 PRINT"{8 UP}{8 RIGHT}";:FORQ=1TO8:FORX
  =1TO8
7Ø PRINT"{RVS}"CHR$(CT%(B));:B=B+1:NEXT:P
  RINT"{DOWN}{8 LEFT}";:NEXT:PC=PC-32:GO
  TO13
```

Program 2:
Character Printer And Screen Dump

```
1 PRINT"{CLR}"TAB(48)"{RED}PEEK AND PRINT
8 FORT=1TO25ØØ:NEXT:DIMC%(3,8,8)
1Ø PC=7448:J=35:POKE36869,255
2Ø CR$=CHR$(J)+CHR$(J+1)+"{DOWN}{2 LEFT}"
  +CHR$(J+2)+CHR$(J+3)
3Ø PRINT"{CLR}{3 DOWN}{9 RIGHT}"CR$
4Ø PRINT"{3 DOWN}{RVS}"PC:FORA=1TO4:FORX=
  1TO8:PRINT"{RVS}"PEEK(PC);:PC=PC+1:NEX
  T:PRINT:NEXT
5Ø PRINT"{RVS}"PC-1
55 PRINT"{DOWN}":INPUT"{RVS}DESIGN NAME";
  DN$:GOSUB6ØØ:GOSUB6ØØØØ
6Ø PRINT"{2 DOWN}":INPUT"{RVS}REVERSE: Y
  {SPACE}OR N";AN$
7Ø IFAN$="Y"THEN9Ø
8Ø GOTO15Ø
9Ø RC$=CHR$(J+1)+CHR$(J)+"{DOWN}{2 LEFT}"
  +CHR$(J+3)+CHR$(J+2)
99 PRINT"{CLR}{3 DOWN}{8 RIGHT}"CR$:PRINT
  "{3 DOWN}{RVS}"
1ØØ L=7168:M=Ø:RC=PC-24:GOSUB2ØØ
11Ø RC=PC-32:GOSUB2ØØ
12Ø RC=PC-8:GOSUB2ØØ
13Ø RC=PC-16:GOSUB2ØØ
14Ø RC=PC-32:FORM=ØTO32:POKERC+M,PEEK(L+M
  ):NEXT
15Ø PRINT"{HOME}"TAB(242)TAB(22Ø);:INPUT"
  {RVS}DESIGN NAME";DN$:GOSUB6ØØ:GOSUB6
  ØØØØ
```

D.E.S.-SOFT ™
a division of
DES-Data Equipment Supply Corp.



ZARCON

By Steven Prentiss

Destined to be one of the best arcade style games on commodore computers.Can you in your Zarcon Fighter shoot all the alien formations while flying through an asteriod field!

5 Levels of excitement
Joystick required

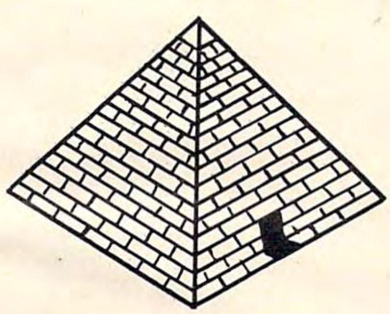
Vic-20 cassette	9.95
Commodore 64 cassette	9.95

SHROOMS

Subterranean Encounter
By Michael Koberstein

Inner Earth has invaded the surface world. The mutant spiders and the moles have taken humans captive and changed them into mussrooms. To rescue them you must enter the caverns. You are their only hope. Good Luck!

Vic-20 & 8K Expander,-Joystick-cass 9.95



SPHINX

3-D ADVENTURE

By Cory Christensen

You the Adventurous Explorer are looking for the lost treasure of the Sphinx. It is rumored to be in the Ancient Pyramid of RA. Beware of deadly traps and hidden passage ways.

Commodore 64-cassette 9.95

You are the commander of squadron of laser ships. It is your duty to defend the cities of earth against incoming alien attack. Spectacular Hi-Res graphics and machine code for super fast action.

Joystick Vic-20 cassette 9.95

SEE YOUR LOCAL DEALER TO SEE OUR FINE PRODUCTS

Dealer inquiries invited ***Software Distribution Available*** **Programs wanted**

(714) 778-5455
D E S
Data Equipment Supply Corp.
8315 Firestone Blvd., Downey, CA 90241
(213) 923-9361

```
155 PRINT"{HOME}"TAB(242)TAB(22Ø)" {RVS}H
    IT RETURN TO GO ON"
160 GETA$:IFA$=""THEN16Ø
170 J=J+4:GOTO2Ø
180 END
200 FORR=RCTORC+7:X=PEEK(R):B=Ø:C=Ø
210 FORA=7TOØSTEP-1:Y=INT(X/2↑A):IFY<=ØTH
    ENZ=Ø:GOTO23Ø
220 Z=2↑C
230 B=B+Z:C=C+1:IFZ=ØTHEN25Ø
240 X=X-2↑A
250 NEXT:PRINT"{RVS}"B;:POKEL+M,B:M=M+1:N
    EXT:PRINT:RETURN
600 FORA=ØTO3:FORB=ØTO7:FORE=ØTO7:C%(A,B,
    E)=Ø:NEXTE,B,A:DC=PC-32
605 FORA=ØTO3:B=Ø:FORD=DCTODC+6:X=PEEK(D)
    :Y=Ø:FORE=7TOØSTEP-1:Y=INT(X/2↑E):IFY
    <ØTHENY=Ø
608 IFY>ØTHENY=2
610 C%(A,B,E)=Y↑B:X=X-Y↑E:NEXT:B=B+1:NEXT
    :DC=DC+8:NEXT
620 FORA=ØTO3:FORB=ØTO6:FORE=7TOØSTEP-1:C
    %(A,7,E)=C%(A,7,E)+C%(A,B,E):NEXTE,B,
    A:RETURN
60000 REM SCREEN COPY
60010 R$=CHR$(145):V$=CHR$(146):OPEN4,4:P
      RINT#4:G=PEEK(648)*256:PRINT#4,R$;:
      A=Ø:FORP=GTOG+5Ø5
60020 C=PEEK(P):C$="":IF(P-G)/22=INT((P-G
      )/22)THENPRINT#4,CHR$(8)+CHR$(13)+C
      HR$(14);
60021 IFA>3THEN6ØØ3Ø
60022 IFC=32THEN6ØØ6Ø
60025 FORE=7TOØSTEP-1:IFC%(A,7,E)=ØTHENC$
      =C$+CHR$(C%(A,7,E):NEXT
60026 C$=C$+CHR$(C%(A,7,E)+128):NEXT:PRIN
      T#4,CHR$(8)C$;:A=A+1:NEXT
60030 IFC>128THENC=C-128
60040 IFC<32ORC>95THENC=C+64:GOTO6ØØ6Ø
60050 IFC>63ANDC<96THENC=C+128:C$=""
60060 C$=C$+CHR$(C):IFLEN(C$)>1THENC$=C$+
      V$+R$
60070 PRINT#4,C$;:NEXT:PRINT#4:CLOSE4:RET
      URN
```

©