grammer's work is ultimately judged on how effectively those rules are implemented in the game without sacrificing playability.

It's even possible to err on the side of authenticity. *Starbowl Football*, the previous effort from the same company that produced this entry, was *too* realistic: Tossing and receiving a pass required moving the receiver exactly under the flying ball and pressing the fire button at precisely the right instant—a near-impossible, frustrating maneuver that took even the most adept joystick maestros a long time to master. Fortunately for the sports-minded, the ball handling techniques in *Star League Baseball* are more accommodating.

Grandstand Viewpoint

In fact, it's one of the most enjoyable sports simulations ever, offering an unusual perspective on the diamond—the view you'd get if you were sitting up in the grandstands behind first base. Joysticks control the action with logically designed patterns. When you're in the field, the ball can be thrown to any of the infielders by pressing the fire button once and moving the stick in the direction of the base's actual position. The location of the man throwing the ball is irrelevant; this makes it easy to learn and execute the moves.

Hit the button twice to return the ball from any player to the pitcher. When he's got the ball, the same action puts him in pitching mode, and he crouches to look for the catcher's signal. Then you hold the button down and move the stick in one of eight directions, each indicating a different type of pitch, to send the ball flying across the plate. The pitcher has the option of changing his mind by releasing the button. This enables him to try to pick off a base runner who looks eager to steal second or third.

At the start of the game, you select from three pitchers, each with his own specialties, which include sinkers, curves, fast balls, and sliders of varying speeds and height. You're better off holding "Knuckles" in the bullpen as a relief pitcher, though. That's right, there's a seventh inning stretch that allows for this option.

Striking Out

To swing the bat, just press the fire button. In addition to visualizing the ball's trajectory, it helps if you glance at its shadow. The distance between the two provides a fair gauge of whether the ball's high, low, or in the strike zone. A batting practice option is convenient for honing this skill to perfection. You can also bunt, and then control the direction in which the ball travels. After each pitch, big block letters display the results (strike, out, ball, home run, etc.) at the top of the screen. When the catcher tosses the ball back to the pitcher, this display is replaced by the number of strikes, balls and outs, the current inning, and other vital information. A scoreboard also appears between innings, posting the runs scored in each inning.

The batter automatically runs to first upon hitting a fair ball, but you'll soon learn that placement—where the ball lands—makes a big difference in whether you get thrown out or not. Infield hits generally result in failure. Hit to the outfield, and you'll have more time to make it to first; the offense gets joystick control of the outfielder nearest the ball, and must race after the ball. He can snare a fly ball by watching its shadow to figure out where it will land.

Stealing Bases

A runner won't advance to the next base unless you move the stick to the right. This allows you to lead off the base, or even steal. But watch out, because it's easy to get caught in a rundown between a pair of infielders. Episodes like this spark genuine excitement when you're playing the computer or a friend, but the two-player games are definitely more fun. Strategy is as important a role as eye-hand coordination, because it pays to figure out the pitcher's pattern. If he just tossed a ball right down the middle and the count's now three and two, will he repeat himself, or try to fake you out with a high slider? You have only split seconds to make the same decisions you would in the batter's box.

The SID chip recreates the smack of a ball connecting with a piece of ash, or plopping into a leather glove. And you'll hear some familiar ballpark sounds when the bases are loaded or one of the heavy hitters approaches the plate. The crisply defined characters wear clearly recognizable hats, and are well animated when you put them through their paces. It's impossible to forget which player has the ball, because he's always black instead of his team's color of white or yellow. Until you've learned the ropes, taking on the computer is only good for humiliation, but the satisfaction of pulling off a successful double play or hitting a grand slam against a human opponent is infinitely more exhilarating than shooting down a thousand flying saucers from the planet Mongo.

Star League Baseball
Gamestar, Inc.
1302 State Street
Santa Barbara, CA 93101
Disk or Tape
$31.95 Atari; $29.95 Commodore 64 ©

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 914
Farmingdale, NY 11737

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 $24.00 (2 years, $45.00; 3 years, $65.00. 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

INSIGHT: Atari

Bill Wilkinson

The assembler listing which accompanies this article is a set of patches to Atari DOS 2.0s. If you own an Atari 1050 drive, these patches will allow you to use it in "enhanced density" mode.

Before we get started with the listing and its explanation, though, let's look at a new tidbit.

Bye-Bye BASIC

Are you an 800XL owner? Do you have an unprotected diskette which boots a machine language program via an AUTORUN.SYS file? Would you like to avoid pushing the OPTION button? Are you willing to follow a few simple steps to do so?

Your 800XL enables and disables the built-in BASIC by changing the contents of location $D301 (54017). In Atari 400s and 800s, this location is usually used to input the state of joysticks 3 and 4. In an 800XL, this port controls various system hardware configurations.

For example, bit 0 of $D301 controls whether the OS ROM is active or whether you are using the RAM underneath it. And—guess what—bit 1 of $D301 controls whether the built-in BASIC is active or not. Specifically, the following table applies:

Bit 0 = 1		**OS ROM enabled**
	0	**OS ROM disabled, RAM enabled**
Bit 1 = 1		**Atari BASIC disabled, RAM enabled**
	0	**Atari BASIC enabled**

At least one of the other bits in $D301 is used (to control whether or not the diagnostic ROM is enabled), but the "normal" values for $D301 are either $FF (BASIC disabled) or $FD (BASIC enabled).

No Option Button

So all we need to do is add a couple of instructions to our AUTORUN.SYS file, to select RAM instead of BASIC, and we will no longer have to hold down the OPTION button. For example, we might add:

```
LDA #$FF
STA $D301
```

And, yet, there is an easier way. Remember, Atari LOAD files may consist of multiple segments, each of which starts with a start address and an end address. The entire file starts with a pair of $FF bytes, but it doesn't hurt if there are extra $FF header bytes in front of other segments.

So consider: If we specify that we have a LOAD file which starts at location $D301 *and* ends at location $D301, the DOS file loader will try to load (and thereby *store*) a single byte at location $D301. This is equivalent to storing a byte via our program.

Disabling BASIC

So simply use the following steps to modify your AUTORUN.SYS to disable the built-in BASIC:

Under Atari DOS 2.0s:

1. Boot your DOS disk while holding down the OPTION button.
2. Put the disk containing the AUTORUN.SYS you want to modify into drive 1.
3. Use the E option from the DOS menu. When prompted for old and new filenames, respond:

   D:AUTORUN.SYS,AUTORUN.OLD
4. Use the K option from the DOS menu. When prompted for filename, starting address, etc., respond:

   D:AUTORUN.SYS,D301,D301
5. Use the C option from the DOS menu. When prompted for from and to filenames, respond:

   D:AUTORUN.OLD,AUTORUN.SYS/A

Under OS/A+ or DOS XL:

1. Boot your DOS disk while holding down the OPTION button. If the DOS XL menu appears, use the Q option.
2. Put the disk containing the AUTORUN.SYS you want to modify into drive 1.
3. Type the command:

   RENAME AUTORUN.SYS AUTORUN.OLD
4. Type the command:

   SAVE AUTORUN.SYS D301 D301
5. Type the command:

   COPY −AF AUTORUN.OLD AUTORUN.SYS

And that's it. Your AUTORUN.SYS file should now be ready to use.

ATARI

ATARI is a trademark of ATARI, INC.

Atari 800XL	Call
1010 Program Recorder	77.00
1027 Letter Quality Printer	Call
1050 Disk Drive	Call
Atari 850 Interface	Call
Atari Light Pen	82.95
Numerical Keypad	104.95

INTERFACES

Aid Interfast I	Call
Ape Face	Call
R-Verter Modem Adaptor	39.95

DISK DRIVES

Indus GT	Call
Percom	Call
Rana 1000	Call
Trak	Call

MOSAIC

48K RAM	94.00
64K RAM/400	149.00
64K RAM/800 + Cable Kit #1	169.00
Handyman-D	23.95
Cable Kit #2	14.95
Superdrive-D	23.95

MODEMS

Hayes Smart Modem 300	Call
Mark II	79.00
Mark VII/Auto Ans/ Auto Dial	Call
Mark XII/1200 Baud	Call
MPP 1000 C	Call
Novation	Call
R-Verter Modem Adaptor	39.95

MONITORS

AMDEK Color 1 +	Call
USI 12" Amber	149.00
USI 12" Green	139.00
USI 14" Color	269.00
BMC 12" Color	229.00
BMC 12" Amber	99.00
BMC 12" Green	95.00
Phillips 12" Green w/Sound	129.00
Phillips 12" Amber w/Sound	139.00

Triangle Replacement Keyboard for 400	54.95
Alien Voice Box II	99.95
Expansion Memory for 600XL	Call

COMMODORE 64

CBM 64	Call
SX-64 Portable	Call
1541 Disk Drive	Call
1526 Printer	279.00
1530 Datasette	66.00
1702 Monitor	Call
1650 AD/AA Modem	89.00
RS 232 Interface	Call

TOUCH TABLETS

Koala Pad-D	69.95
Koala Pad-Cart	74.95

PRINTERS

Alphacom 40C w/Interface	99.95
Alphacom 80C w/Interface	189.95
Axiom AT-550	329.00
Epson	Call
Gemini 10X	269.00
Okidata	Call
Prowriter I	Call
Riteman	Call

SUPER PRINTER PACKAGES

Gemini 10X and Apeface	339.00
Prowriter and Apeface	409.00
Prowriter + Aid Interf. + Cable	485.00
Gemini 10X and Cardco + G	349.00
Prowriter and Cardco + G	419.00

No additional ship. charges on printer packages in Continental USA

☆ Call for prices on joysticks, printer cables, blank floppy disks, books and other computer accessories. ☆

ATARI SOFTWARE

ATARI

Caverns of Mars-Cart	32.95
Donkey Kong-Cart	37.95
Donkey Kong Jr.-Cart	39.95
Eastern Front-Cart	32.95
Joust-Cart	39.95
Jungle Hunt-Cart	35.95
Mario Bros.-Cart	35.95
Ms. Pac-Man-Cart	39.95
Pengo-Cart	35.95
Pole Position-Cart	39.95
Qix-Cart	32.95
Realsports Football-Cart	35.95
Robotron: 2084-Cart	35.95
Star Raiders-Cart	32.95
Tennis-Cart	35.95
Atariwriter-Cart	74.95
Family Finances-D	37.95
Home Filing Manager-D	37.95
Mailing List-T	19.95
Syncalc-D	74.95
Synfile-D	74.95
Syntrend-D	74.95
Logo-Cart	79.95
My First Alphabet-D	26.95
Atarimusic I or II-D	29.95
Conversational Lang.-T	44.95
Final Legacy-Cart	Call
Ballblazer-Cart	Call
Plato-Cart	Call
Paint-D	33.95
Microsoft Basic II-Cart	67.95
Atari Macro-Assembler-Cart	67.95
Atari Basic-Cart	41.95
Assembler Editor-Cart	46.95

BRODERBUND

AE-D	23.95
Arcade Machine-D	41.95
Bank Street Writer-D	49.95
Choplifter-D	23.95
Drol-D	23.95
Gumball-D	20.95
Loderunner-D	23.95
Mask of the Sun-D	27.95
Operation Whirlwind-D	27.95

DATASOFT

Basic Compiler-D	55.95
Bruce Lee-D/T	23.95
Dallas-D	23.95
Heathcliff-D/T	23.95
Letter Wizard	Call
Micropainter-D	23.95
Money Wizard	Call

DATASOFT (cont'd.)

Nibbler-D/T	20.95
Spell Wizard-D	34.95
Teletalk-D	34.95
Zaxxon-D/T	27.95

EPYX

Dragons/Pern-D/T	27.95
Fun With Art-Cart	27.95
Gateway to Apshai-Cart	27.95
Jumpman Jr.-Cart	27.95
Jumpman-D/T	27.95
Pitstop-Cart	27.95
Puzzlemania	Call
Summer Games	Call
Temple of Apshai-D/T	27.95

INFOCOM

Deadline-D	34.95
Enchanter-D	34.95
Infidel-D	34.95
Planetfall-D	34.95
Sorcerer-D	34.95
Starcross-D	27.95
Suspended-D	34.95
Witness-D	34.95
Zork I, II, or III-D	27.95
Sea Stalker-D	34.95

MISCELLANEOUS ATARI

Omnimon	82.95
Pitfall-Cart	31.95
River Raid-Cart	31.95
Diskey-D	34.95
Ultra Disassembler-D	34.95
Zombies-D/T	23.95
Codewriter-D	69.95
Star League Baseball-D/T	22.95
Star Bowl Football-D/T	22.95
Death in the Caribbean-D	27.95
Dino Eggs-D	27.95
The Heist-D	23.95
Gyruss-Cart	34.95
James Bond-Cart	34.95
Popeye-Cart	34.95
Q-Bert-Cart	34.95
Star Wars-Cart	34.95
Return of Heracles-D	26.50
Master Type-D/Cart	27.95
Songwriter-D	27.95
Flight Simulator II-D	37.95
Chatterbee-D	27.95
S.A.M.-D	41.95
Castle Wolfenstein-D	20.95
Compuserve Starter Kit	34.95
Home Accountant-D	52.95
Ken Uston's Blackjack-D	49.95

MISC. ATARI (cont'd.)

Megafont-D	19.95
Monkey Wrench II-Cart	49.95
Movie Maker-D	41.95
Pogo Joe-D	20.95
Spelunker-D	27.95
Ultima III-D	41.95
Jupiter Mission-D	34.95
Dreadnaught Factor-Cart	31.95
Keystone Kapers-Cart	31.95
Antica 4-D	37.95
Boulder Dash-D/T	20.95
Scraper Caper-Cart	34.95
Miner 2049'er-Cart	34.95
Beachead-D/T	23.95

OPTIMIZED SYSTEMS

Action-Cart	74.95
Basic XL-Cart	74.95
MAC/65-Cart	74.95
DOS XL-D	27.95
Action Aid-D	27.95

SIERRA ON-LINE

Aquatron-D	20.95
Dark Crystal-D	27.95
Homeword Speller-D	34.95
Homeword-D	49.95
Oil's Well-D	23.95
Prisoner-D	23.95
Quest for Tires-D	23.95
Ultima II-D	41.95
Ultima I-D	23.95
Wiztype-D	23.95

SSI

Carrier Force-D	41.95
Combat Leader-D/T	27.95
Cosmic Balance II-D	27.95
Cosmic Balance-D	27.95
Broadsides-D	27.95
War in Russia-D	55.95
50 Mission Crush-D	27.95
Questron-D	34.95
Rails West-D	27.95

To Order Call Toll Free

800-558-0003

For Technical Info, Order Inquiries, or for Wisc. Orders ·

414-351-2007

COMMODORE 64 SOFTWARE

ACCESS

Neutral Zone-D/T	23.95
Spritemaster-D/T	23.95
Beachhead-D/T	23.95
Master Composer-D	27.95

BATTERIES INCLUDED

Consultant-D	69.95
Paperclip w/Spellpak-D	84.95
Super Busscard II	Call
Home Inventory-D	23.95
Recipe-D	23.95
Audio/Video Cat-D	23.95
Mail List-D	23.95
Stamps-D	23.95

CARDCO

Cardprint/B	47.95
Cardco + G	69.95
Cardboard/5	59.95
Cardkey	39.95
Cassette Recorder	47.95
Printer Utility-D/T	19.95
Write Now-Cart	34.95
Mail Now-D	29.95

COMMODORE

Program Ref. Guide	19.95
Assembler-D	39.95
Easy Finance I, II, III, IV-D	19.95
Easy Calc-D	64.95
Easy Mail-D	19.95
Easy Script-D	44.95
Easy Spell-D	19.95
Logo-D	57.95
The Manager-D	39.95
General Ledger-D	39.95
Accts. Rec.-D	39.95
Accts. Pay.-D	39.95
Magic Desk-D	52.95
Zork I, II or III-D	32.95
Suspended-D	32.95
Starcross-D	32.95
Deadline-D	32.95
Soccer-Cart	29.95

HANDIC

64 Forth-Cart	29.95
64 Graf-Cart	23.95
Stat 64-Cart	23.95
Calc Result Easy-Cart	34.95
Calc Result Advanced-D	74.95
Calc Result Adv.-Cart	69.95
The Diary-Cart	23.95
The Tool-Cart	29.95

INSTA (CIMMARON)

Insta-Writer-Cart	39.95
Insta-Mail-D	24.95
Insta-File-D	49.95
Management Combo	64.95
Insta-Calc-Cart/D	31.95
Insta-Graph-D	24.95
Insta-Vestor-D	31.95
Insta-Speed-D	99.95
Insta Sched-Cart/D	49.95
Insta-Music-Cart/D	79.95
Invest Combo	74.95

MISC. COMMODORE

Ken Uston's Blackjack-D	49.95
Quick Brown Fox-D/Cart	34.95
Ultima III-D	41.95
Flight Simulator II-D	37.95
Night Mission/ Pinball-D/T	20.95
Praticaic PS-D	59.95
M-File-D	64.95
Word Pro 3 + /Spell-D	74.95
Home Accountant-D	52.95
Step By Step-D/T	44.95
Barron's Sat.-D	67.95
Bristles-D/T	20.95
Telestar 64-Cart	37.95
Star League Baseball-D/T	20.95
Castle Wolfenstein-D	20.95
Mastertype-D/Cart	27.95
Vic Switch	124.95
First Class Mail-D	34.95
Aztec-D	27.95
Miner 2049er-Cart	27.95
Sea Dragon-D/T	23.95
Diskey-D	34.95
Hodge Podge-D/T	19.95
Strip Poker-D	23.95
Mr. Robot-D	23.95
Paint Magic-D	34.95
Pooyan-D/T	20.95
Astro Chase-D/T	20.95
Flip Flop-D/T	20.95

MISC. COMM. (cont'd.)

Basic Building Blks-D	54.95
Critical Mass-D	27.95
Rescue Squad-D	20.95
Super Text Word Pro.-D	69.95
Beyond Wolfenstein-D	23.95
Sam-D	41.95
Chatterbee-D	27.95

SPINNAKER

Adventure Creator-Cart	27.95
Aerobics-D	30.95
Aegean Voyage-Cart	27.95
Alf in the Color Caves-C	27.95
Alphabet Zoo-Cart	23.95
Bubble Burst-Cart	27.95
Cosmic Life-Cart	23.95
Delta Drawing-Cart	27.95
Facemaker-Cart	23.95
Fraction Fever-Cart	23.95
Grandma's House-D	23.95
Jukebox-Cart	27.95
Kids on Keys-Cart	23.95
Kidwriter-D	23.95
Kindercomp-Cart	20.95
Ranch-Cart	27.95
Rhymes/Riddles-D	20.95
Search/ Amazing Thing-D	27.95
Snooper #1-D	30.95
Snooper #2-D	30.95
Story Machine-Cart	27.95
Trains-D	27.95
Up For Grabs-Cart	27.95

SYNAPSE

Blue Max-D/T	23.95
Drelbs-D/T	23.95
Fort Apocalypse-D/T	23.95
Necromancer-D/T	23.95
New York City-D/T	23.95
Pharoah's Curse-D/T	23.95
Protector II-D/T	23.95
Quasimodo-D/T	23.95
Rainbow Walker-D/T	23.95
Relax Stress Reduction Sys.	64.95
Shamus Case II-D/T	23.95
Shamus-D/T	23.95
Slam-Ball-D/T	23.95
Survivor-D/T	23.95
Zaxxon-D/T	27.95
Zepplin-D/T	23.95

D-Disk T-Cassette Cart-Cartridge

NEW LOWER PRICES

ComputAbility™

NEW PRODUCTS

NO SURCHARGE FOR MASTERCARD OR VISA

ORDERING INFORMATION. Please specify system. For fast delivery send cashier's check, money order or direct bank transfers. Personal and company checks allow 2 weeks to clear. Charges for COD are $3.00. School Purchase Orders welcome. In CONTINENTAL USA, include $3.00 shipping per software order. Include 3% shipping on all Hardware orders, minimum $3.00. Mastercard & Visa please include card # and expiration date. WI residents please add 5% sales tax. HI, AK, FPO, APO, Canadian orders — add 5% shipping, minimum $5.00. All other foreign orders, please add 15% shipping, minimum $10.00. All goods are new and include factory warranty. Due to our low prices, all sales are final. All defective returns must have a return authorization number. Please call 414-351-2007 to obtain an RA# or your return will NOT be accepted for replacement or repair. Prices and availability are subject to change without notice.

COMPUTABILITY
P.O. Box 17882
Milwaukee, WI 53217

ORDER LINES OPEN
Mon-Fri 11 AM - 7 PM CST
Sat 12 PM - 5 PM CST

Check The Pointers

Caution! Even though the built-in BASIC is now disabled, HIMEM (the contents of location $2E5) and RAMTOP (contents of location $6A) will still reflect the 40K byte configuration where BASIC is present. If your program pays attention to one or both of these two values, it would also be worth performing the following steps:

1. Change RAMTOP to reflect the full 48K bytes.
2. Close channel zero (the screen editor).
3. Open channel zero for the E: device.

These steps will insure that all 48K bytes of accessible RAM are in use by your program. I won't go into how to accomplish these here and now. Write if you would like me to show how to code those steps in machine language.

Coming Attractions

A project related to this, which I hope to implement in an upcoming column, would be an "M:" device driver. Once upon a lifetime ago, in this column, I presented such a driver. It used the "excess" memory (between the top of a BASIC program and the bottom of the graphics screen) as a pseudodevice.

I would like to do the same thing again, but this time use the extra memory under the OS ROMs or under the built-in BASIC as a superfast RAM disk. Stay tuned for further developments.

DOS 2.0s For Enhanced Density 1050s

First, I would like to point out that the task of reconfiguring Atari DOS 2.0s for an enhanced density 1050 is difficult. I would also like to note that it is *extremely* difficult (if not impossible) to finish the task if you have only one drive.

So, may I suggest that you cooperate with a friend and his drive if you have only one of your own. If your friend's drive is an 810 or a non-Atari drive, it should be set up as drive 1. Your 1050 should be set up as drive 2.

Also, you should use an assembler capable of placing its object code directly in memory. (For example, AMAC—the Atari Macro Assembler—cannot be used for this job.) This is because loading the DOS-modifier code from a disk will use DOS itself, and you are almost guaranteed to run into conflicts. Atari's Assembler Editor cartridge, the old OSS EASMD, OSS's MAC/65, and (I believe) SYNASSEMBLER will all work properly (though the syntax for SYNASSEMBLER may vary a bit from what I show here).

You should boot a normal Atari DOS 2.0s disk, making sure that you can access a normal single diskette in drive 2 (at least to the point of making sure you can list its directory). Be sure you have at least two (2) blank or junk disks ready and at hand. Then begin.

Patching DOS

Type in the program, as shown herein. You may use automatic line numbering if you wish. Type in just the part from the right of the line numbers. LIST or SAVE the source code to disk and then assemble it. Check it against the listing given here. Do not proceed until you are reasonably sure that you have typed it in and assembled it correctly.

Then change line 1000 to read:

```
1000  .OPT NOLIST,OBJ
```

and assemble the code once more.

Voilà! DOS has been patched!

But, because DOS's DRVTBL has changed format, you *must* now hit the SYSTEM RESET key. Then give the DOS command from your assembler. Assuming that you get to the DOS menu (and if you don't, you did something wrong), it would probably be a good idea to immediately format (menu option I) a blank disk in drive 1 and write the DOS files (option H).

Implementing Enhanced Density

Now comes the tricky part. The way we have patched DOS 2.0s, DOS automatically checks each drive at power-on (or SYSTEM RESET) time to find its current configuration (single density, double density, or enhanced density). But the 1050 assumes it is in single-density mode unless you have inserted an enhanced-density diskette. So, up until now, DOS thinks it is working with all single-density disk drives. How do we change its mind?

The easy way: Turn your power off, put your BASIC (or BASIC XL) cartridge into your machine, and turn the power back on, thus booting the disk we just formatted and wrote DOS files to. Insert a blank disk into the second drive (your 1050). From BASIC, give the following command:

```
XIO 254,#1,0,34,"D2:"
```

If you are a faithful reader, you will recognize that as the format command, given from BASIC. But the 34 in the next-to-last position is new! That's right. As we have patched DOS, a nonzero value given in AUX2 is assumed to be the format command value to be sent to the disk drive. The *only legal values* here are 33 (for single density, à la 810 drives) and 34 (for 1050 enhanced density)!

Now drive 2 contains what we hope is an enhanced-density diskette. Once more, hit SYSTEM RESET so that DOS will recognize the new density. Then give the DOS command from BASIC. Once in DOS, use the H menu option to write the DOS files to drive 2.

If you have performed all these steps correctly, you should now have a bootable enhanced-density diskette in drive 2. You might wish to change your 1050 back to being drive 1 and try to boot from it with this new diskette.

Simpler Commands

The beauty of this system is that, once you have created this one enhanced-density master, you may make new enhanced-density masters by using just the I and H commands from the DOS menu.

There is, however, one potential problem. How do you copy files from an old single-density disk to a new enhanced-density disk? For now, the only practical way is to borrow a second drive and have one of each type of disk on your system. There may be ways around even this problem. We'll see.

Patching Other DOS Versions

The patch program given here will also work on all versions of OS/A+ and DOS XL from 1.2 to 2.3 (except that it will *not* patch the DOSXL.SYS versions).

The procedures are almost the same, but it is significantly easier to use a single drive. Try the following if you have only a single disk, on which you boot OS/A+ or DOS XL:

1. Type in, save, and check out the patch listing as described above.
2. Hit SYSTEM RESET. If you end up back in an assembler cartridge, type a DOS command.
3. From the D1: prompt, use an INIT command. Or use the I option from the DOS XL menu.
4. Use Option 1 (on a blank disk) or 3 (on an existing disk) of INIT. Use Option 4 to return to DOS.
5. Insert a BASIC cartridge. Reboot from the disk you just INITed.
6. Type the following BASIC command:
   XIO 254,#1,0,34,"D1:"
7. Hit SYSTEM RESET after the formatting is finished. If you are not then in the BASIC cartridge, use the CAR command.
8. Type the following BASIC command line:
   OPEN #1,8,0,"D1:DOS.SYS" : CLOSE #1

The reason the procedure works on a single drive is that neither OS/A+ nor DOS XL requires the DUP.SYS file of Atari DOS. The disk initialization can thus be performed entirely from BASIC.

Patches To Atari DOS 2.0s

```
0000           1000      .OPT LIST,NO OBJ
               1010 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               1030 ; PATCHES TO ATARI DOS 2.0S
               1040 ;
               1050 ; THESE PATCHES ALLOW AN ATARI 1050 DRIVE
               1060 ;   TO UTILIZE ENHANCED DENSITY UNDER
               1070 ;   DOS 2.0S, TO A MAXIMUM OF 965 FREE SECTORS
               1100 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
               1110 ;
               1120 ; EQUATES -- TAKEN FROM THE LISTING OF
               1130 ;            ATARI DOS AS PUBLISHED IN
               1140 ;              "INSIDE ATARI DOS"
               1150 ;            FROM COMPUTE! BOOKS
               1160 ;
    =1311      1170 DRVTBL = $1311
    =1301      1180 CURFCB = $1301
    =0048      1190 ZSBA =  $48
    =11DB      1200 DERR1 = $11DB
    =12FE      1210 DRVTYP = $12FE
    =0021      1220 DCBCFD = '!
    =0302      1230 DCBCMD = $0302
    =0045      1240 ZDRVA = $45
    =0A4A      1250 NOBURST = $0A4A
    =0A4C      1260 WRBUR = $0A4C
    =0D18      1270 XFORMAT = $0D18
    =0BD6      1280 XFV =    $0BD6
    =1372      1290 Z   =    $1372
    =034B      1300 ICAUX2 = $034B
    =1382      1310 FCBOTC = $1382
               1340 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

The rest of this listing will appear in "Insight: Atari" next month. ©

ML Tracer

Thomas G. Gordon

Attempting to debug a machine language program can sometimes be a trying experience, especially when the program always seems to exit into the twilight zone. And trying to study a program in ROM can be just as frustrating, even with a disassembler (where do branch instructions go?). Here's an excellent programming utility: a single-stepper for Atari, Apple, and all Commodore computers.

Anyone who has ever worked with machine language knows how helpful it can be to be able to single-step through a program. "ML Tracer" allows you to step through a machine language routine one event at a time and print out the contents of all of the microprocessor registers after each instruction. It also allows you to follow all branches, jumps, and returns. The program will display the address, opcode, mnemonic, and operand of each instruction.

Three versions are included. Program 1 runs on all Commodore computers (for the VIC, 8K or more expansion memory is required). Program 2, for the Apple II, is only slightly different from the Commodore version. The Atari version, Program 3, has more substantial changes, but its structure is still quite similar. Since all the versions have the same line numbers, references in this article apply to all versions unless otherwise stated.

When Tracer is run, there will be a ten-second delay while the DATA statements are read. You'll then be asked for the hex address of the ML program you wish to examine. You can change the contents of any register, before each instruction is executed. Press *a* for the accumulator, *x* for the x register, *y* for the y register, *s* for the stack pointer, *p* for the processor status, or *i* for the instruction pointer (program counter). On the Atari, also press RETURN. When you're through loading registers, press RETURN once more to execute the next instruction.

Hexadecimal numbers are used for all input and output. If you enter an address as a one-, two-, or three-digit hexadecimal number, zeros will be added on the left to make a four-digit number. If too many digits are entered, the rightmost four digits will be used. The same applies to changing the value in a register. The number that you enter will be converted to a two-digit hexadecimal number using the same rules.

The Execution Subroutine

The program is written mostly in BASIC, but contains two machine language subroutines. The first, the initialization subroutine, copies the lowest three pages (768 bytes) of RAM, which are used by BASIC, to a location above the BASIC program. The other, the execution subroutine, exchanges the two three-page blocks of data and loads all the registers with their saved values, then executes one instruction (which has been POKEd in from BASIC). When the instruction has been executed, the registers are saved and BASIC's original lower three pages of memory are restored.

The same technique was used to identify addressing modes as in my disassembler ("A 6502 Disassembler," COMPUTE!, January 1981, p. 81). Lines 10000–10031 contain four-character extended mnemonics for the 6502's instruction set. The fourth character is a tag code identifying the addressing mode of the instruction. In lines 110–120, the mode is identified and the proper subroutine is called.

There are several instructions which cannot be allowed to actually execute in the machine language subroutine. If any control transfer instructions (JMP, JSR, RTS, RTI, or a conditional branch) were executed, control would not be returned properly to the BASIC program. These instructions are simulated in BASIC instead, so that they appear to execute successfully. The SEI and CLI instructions are ignored, since interrupts are always disabled during the execution subroutine.

ATARI

600XL ... CALL
800XL ... CALL

DISK DRIVES
Indus GT ... Call
Trak AT-D2 ... $388
Trak AT-1 ... $319
Trak AT-D4 ... Call
Astra 1620 (Dual) ... $499
Percom ... Call
Atari 1050 ... $349
Rana 1000 ... Call

INTERFACES
Axiom 846 ... Call
Ape Face ... Call
Atari 850 (In Stock) ... $169
Interfast 1 ... $150
Microbits 1150 ... Call
Axiom Buffer ... Call

MEMORIES
Microbits 64K (XL) ... $126
Mosaic 48K (400) ... $98
Mosaic 64K (400/800) ... Call
Mosaic 32K ... $68
Atari 64K (600XL) ... Call

DIRECT PRINTERS
Axiom AT-100 ... $219
Atari 1027 ... $285
Axiom 550 AT ... $319
Axiom 700 AT ... $469
Atari 1025 ... $395

DIRECT MODEMS
Microbits 1000C ... $128
Atari 1030 ... $114

OTHER ATARI
400 Keyboard ... Call
Koala Pad ... $67
Chalkboard Pad ... $75
Bit-3 80 Column ... $228
ATR-8000 (64K) ... $448
ATR-8000 (16K) ... $295
Alien Voice Box ... $98
1010 Recorder ... $74

SPECIALS
Wico Joystick ... $21
Wico Trackball ... $36
Compuserve ... $26
Covers ... $7
Maxells ... $25
Memorex ... $24
Elephants ... $18
Flip 'N File ... $20

CBM 64 ... CALL
SX64 ... CALL
1701 MONITOR ... $249

ASTRA 1620
DISK DRIVE SYSTEM

MORE FOR YOUR MONEY
DOUBLE OR SINGLE DENSITY
TWO DRIVES
FOR ATARI

SPECIAL ... $499

Printers/Etc.

STAR
Gemini 10X ... $268
Gemini 15X ... $378
Delta 10 ... $378
Star Letter Qual. ... Call

CITOH
Prowriter ... $335
Prowriter II ... $648
Starwriter ... $1098
Printmaster ... $1448

EPSON
FX80 ... Call
MX100 ... Call
RX 80 F/T ... Call

OKIDATA
82A ... Call
84P ... Call
92 ... Call
93 ... Call

SMITH CORONA
TP II ... $438

MANNESMAN
160L ... $588
Spirit ... Call

JUKI ... Call

MONITORS

AMDEK
Color I ... $274
V300 ... $139
V300A ... $149
Color II ... $449

SAKATA ... Call

NEC
GRN (JB1260) ... $115
GRN (JB1201) ... $149
Color Composite ... $298
RGB Color ... $598

PANASONIC ... Call

MODEMS

NOVATION
J-Cat ... $99
Apple Cat II ... $259
D-Cat ... $149

HAYES
Smartmodem ... $209
Smartmodem 1200 ... $498
Micromodem II ... $259
Micromodem IIe ... $239

D = Disk
R = ROM (Cartridge)
C = Cassette

COSMIC COMPUTERS
UNLIMITED

727 BREA CANYON RD., SUITE 16
WALNUT, CA 91789

ORDER LINES OPEN MON-SAT 8 am - 8 pm

(800) 626-7642

PLEASE FOR ORDERS ONLY
SORRY, NO COD'S

IN CALIF. **(714) 594-5204**

FOR TECHNICAL INFO, ORDER INQUIRIES, OR FOR CALIFORNIA ORDERS

Add $2.50 shipping per software order in continental U.S. Add $5.00 shipping per software order for AK, HI, FPO-APO. Add $10.00 or 15% (whichever is greater) per software order for non-U.S. Call for cost of hardware shipping. Calif. residents add 6½% sales tax. Cashiers checks or money orders filled within 24 hours for items in stock. Personal checks require 4 weeks to clear. MasterCard and Visa OK for software only within continental U.S., add 3% surcharge. Include card no., expiration date and signature. Due to our low prices, all sales are final. All defective returns must have a return authorization number. Please call to obtain one before returning goods for replacement or repair. Prices & availability subject to change.

ATARI SOFTWARE

ADVENTURE INT'L
Ultra Disassembler (D) ... $33
Diskey (D) ... $33
Adv. 1-12 (each) (C) ... $18
Saga 1-12 (each) (D) ... $27

ATARI
Atari Writer (R) ... $68
Paint (D) ... $30
Microsoft Basic II (R) ... $64
Visicalc (D) ... $139
Home File Mgr (D) ... $36
Assembler Editor (R) ... $44
Qix (R) ... $32
Dig Dug (R) ... $32
Atari Logo (R) ... $72
Robotron (R) ... $35

AVALON HILL
Telengard (C) $16 (D) $19
Close Assault (C) 20 (D) 23
TAC (D) ... $27

BRODERBUND
Arcade Machine (D) ... $39
Bank St. Writer (D) ... $46
Oper. Whirlwind (D) ... $27
Choplifter (D) $23 (R) $29
David's Midnight (D) ... $23

CBS SOFTWARE ... Call

CONTINENTAL
Home Accountant (D) ... $48
Tax Advantage (D) ... $45

COUNTERPOINT SW ... Call

DATASOFT
Pooyan (C/D) ... $20
Teletalk (D) ... $33
Basic Compiler (D) ... $55
Graphic Gen (D) ... $17
Graphic Master (D) ... $23
Micropainter (D) ... $23
Zaxxon (C/D) ... $27
Text Wizard/Spell (D) ... $55

EASTERN HOUSE
Monkey Wrench II ... $51

EDUCATIONAL SW
Tricky 1,2,3 or 4 ... $15
Tricky 5-13 ... $22

EPYX
Dragon Riders (C/D) ... $27
Temple APS (C/D) ... $27
Jumpman (C/D) ... $27
Pit Stop (R) ... $27

FIRST STAR
Boulder Dsh (C/D) 20 (R) 27
Bristles (C/D) ... $20
Flip Flop (C/D) ... $20

GAMESTAR
Football (C/D) ... $21
Baseball (C/D) ... $21

INFOCOM
Zork I, II or III (D) ... $27
Deadline (D) ... $34
Starcross (D) ... $27
Suspended (D) ... $34
Witness (D) ... $34
Planetfall (D) ... $34
Enchanter (D) ... $34
Infidel (D) ... $34

KRELL SAT ... Call

INTELL. STATEMENTS
Prof. Blackjack (D) ... $46

LJK
Letter Perfect (D) ... $74
Data Perfect (D) ... $74
Spell Perfect (D) ... $56
Letter Perfect (R) ... $74

MICROPROSE
Solo Flight (D) ... $26
Hellcat Ace (C/D) ... $23

MONARCH
ABC Compiler (D) ... $55

OPTIMIZED SYSTEMS
Action (R) ... $65
Basic XL (R) ... $65
Mac 65 (D) ... $58
C-65 (D) ... $58
Bug 65 (D) ... $23

PARKER BROS
Astrochase (R) ... $33
Death Star (R) ... $33
Q-Bert (R) ... $33
Popeye (R) ... $33

PENGUIN
Graphics Magician (D) ... $39
Spy's Demise (D) ... $21

QUALITY SW
Return of Hercules (D) ... $22
Ali Baba (D) ... $22

RESTON
Moviemaker (D) ... $45

SCARBOROUGH SYS.
Mastertype (D/R) ... $27
Songwriter (D) ... $27

SCHOOL WIZWARE ... Call

SEGA
Star Trek (R) ... $27
Buck Rogers (R) ... $27
Congo Bongo (R) ... $27

SIERRA ON-LINE
Homeword (D) ... $46
Ultima II (D) ... $39
Dark Crystal (D) ... $26
Wiz. & Princess (D) ... $22

SPINNAKER
Snooper Troop 1,2 (D) ... $30
Most Amazing (D) ... $27
Kids on Keys (D) ... $20
Trains (D) ... $27
Delta Drawing (R) ... $27
Aerobics (D) ... $34

STRATEGIC SIM.
Broadsides (D) ... $27
Carrier Force (D) ... $39
Combat Leader (D) ... $27
Rails West (D) ... $27
Epidemic (D) ... $23
Eagles (D) ... $27
Cosmic Bal I or II (D) ... $27

SUBLOGIC
Flight Simulator II (D) ... $36
Pinball (C/D) ... $20

SYNAPSE
Syn File + (D) ... $65
Syn Text (D) ... $65
File Manager (R) ... $54
Fort Apocalypse (C/D) ... $23
Dimension X (C/D) ... $23
Blue Max (C/D) ... $23
Encounter (D/R) ... $23
Zepplin (C/D) ... $23
Pharoah's Curse (C/D) ... $23

TRONIX
S.A.M. (D) ... $39
P.M. Animator (D) ... $29
Juice (C/D) ... $20
Chatterbee (D) ... $27

MISCELLANEOUS
Miner 2049 (R) ... $34
Millionaire (C/D) ... $45
Zombies (C/D) ... $23
Prisoner 2 (D) ... $27
Sargon II (D) ... $23
Gridrunner (R) ... $20
B-Graph (D) ... $65
Miles Invoices (D) ... $57
Castle Wolfenstein (D) ... $20
Odesta Chess (D) ... $46
Financial Wizard (D) ... $41
Ultima III (D) ... $39

COMMODORE

DISK DRIVES
MSD (170K) ... $349
MSD (Dual) (170Kx2) ... $539
Laser (170K) ... $325
Commodore 1541 ... $239
Concord (170K) ... Call

INTERFACES
The Connection ... $85
Bus Card ... $149
Cardco G + ... $69
Cardco B ... Call
MSD (IEEE) ... $98
Cardco 5 Slot ... $48
RS-232 ... Call

80 COLUMN BDS
Batteries 80 Col ... $149
Video Pak 80 ... $129
Z80 Video Pak ... $209

DIRECT PRINTERS
MPS 801 ... $219
Commodore 1526 ... $288
Cardco LQ/I ... $498
1520 Color Printer ... $129

DIRECT MODEMS
Hesmodem ... $53
1650 Automodem ... $99
1600 Modem ... Call

RECORDERS
Cardco Recorder ... $48
1530 Commodore ... Call
Cassette Interface ... $29

64 SOFTWARE 64

ACCESS SOFTWARE
Beached (D) ... $24
Neutral Zone (C/D) ... $24
Spritemaster (D) ... $25

AVALON HILL
Nuke War (C) ... $12
Androm. Conquest (C) ... $14
Midway Campaign (C) ... $13
Computer Football (C) ... $13
Telengard (C) $16 (D) $19
Flying Ace (C) ... $15
Moon Patrol (C) ... $17

BATTERIES INCLUDED
Paper Clip (D) ... $69
Consultant (D) ... $75
Bus Card ... $138
80 Column BD ... $138
Paper Clip w/Spell (D) ... $85
Spell Pack (D) ... $37
Organizer Series (Ea) ... $22

BLUE SKY
Calc Result Adv. ... $99
Calc Result Easy ... $57

BRODERBUND
Bank St. Writer (D) ... $46
Operat. Whrlwnd (D) ... $27
Choplifter (R) ... $27
Lode Runner (D) 23 (R) 27

CBS SOFTWARE
Success with Math (D) ... $17
Wbstr Word Game (D) ... $20
Learning Bridge (D) ... $55

COMMODORE
Simon's Basic (R) ... CALL
Magic Desk (R) ... $48
Logo (D) ... $45
Assembler 64 (D) ... $15
Easy Script 64 (D) ... $32
Accounting Pkgs (Ea) ... $32

COMPUTER ADV. IDEAS
Game Show (D) ... $27
Win W/Words I or II (D) ... $15
Sports Facts (D) ... $15

CONTINENTAL S.W.
Home Accountant (D) ... $48
Tax Advantage (D) ... $45
FCM (D) ... $34

COUNTER POINT SW
Early Games (Ea) ... $20

CREATIVE SOFTWARE
Moondust (R) ... $23
Save New York (R) ... $23
Pipes (R) ... $23
Househld Finance (C) ... $20

DATASOFT
Pooyan (C/D) ... $20
Moon Shuttle (D) ... $20

ENTECH
Studio 64 (C/D) ... $28
Database 64 (D) ... $45

EPYX
Temple of APS (C/D) ... $27
Jumpman (C/D) ... $27
Dragonriders (C/D) ... $27
Pitstop (R) ... $27
Gateway to APS (R) ... $27

FIRST STAR
Astrochase (C/D) ... $20
Bristle (C/D) ... $20
Flip Flop (C/D) ... $20

FUTURE HOUSE
Comp. Pers. Account. ... $56

HES
Synthesound 64 (D) ... $23
Omnicalc (D) ... $34
Omniwriter (D) ... $45
Hesmon (D) ... $27
64 Forth (R) ... $39
Multiplan (D) ... $65
Turtle Graphics (R) ... $39

INFOCOM
Planet Fall (D) ... $34
Enchanter (D) ... $34
Infidel (D) ... $34

JINSAM
Mini Jini (R) ... $75

MICROSOFTWARE INT'L
Practicalc 64 (C) 34 (D) 36
Spreadsheet (C) 49 (D) 52
Practifile 64 (D) ... $36

MIRAGE CONCEPTS
Data Base (D) ... $68
Word Processor (D) ... $68

MICROSPEC
Database (D) ... $44
Mailing List (D) ... $32
Checkbook Mgr (D) ... $47
G/L (D) ... $44
A/P (D) ... $47
A/R (D) ... $47

PARKER BROS
Q-Bert (R) ... $33
Popeye (R) ... $33
Frogger (R) ... $33

PROFESSIONAL SW
Wordpro W/Spell (D) ... $68
Spellright (D) ... $45

QUIKTEX
Quick Br. Fox (R) ... $49

RAINBOW
File Assistant (D) ... $46
Writers Assistant (D) ... $46
Spreadsht Assist. (D) ... $56

SCARBOROUGH
Mastertype (D/R) ... $27
Song Writer (D) ... $27

SEGA
Star Trek (R) ... $27
Buck Rogers (R) ... $27
Congo Bongo (R) ... $27

SOFTSMITH
Touch Typing (C/D) ... $21

SOUTHERN SOLUTIONS
Businessman (D) ... $48
Bill Payer (D) ... $48
Bill Collector (D) ... $48
Paymaster (D) ... $48

SPINNAKER
Snooper 1 or 2 (D) ... $27
Aerobics (D) ... $34
Kids on Keys (D) ... $20
Most Amazing (D) ... $27
Kindercomp (D) ... $20
Alphabet Zoo (D) ... 20
Trains (D) ... $27
Delta Drawing (R) ... $27
Delta Music (R) ... $27

SUBLOGIC
Flight Simulator II (D) ... $36
Pinball (C/D) ... $22

SYNAPSE
Ft. Apocalypse (C/D) ... $23
Necromancer (C/D) ... $23
Zaxxon (D) ... $23
Blue Max (D) ... $23
Pharoah's Curse (C/D) ... $23
Zepplin (D) ... $23
Quasimodo (D) ... $23

SSI
Combat Leader (C/D) ... $27
Computer Baseball (D) ... $27
Eagles (D) ... $27
Ringside Seat (D) ... $27
Tigers In Snow (C/D) ... $27
Battle Normandy (C/D) ... $27

TIMEWORKS
Dungeons of Alg. (C/D) ... $17
Robbers Lost (C/D) ... $17
Money Mgr. (C/D) ... $17
Wall Street (C/D) ... $17
Data Manager (C/D) ... $17
Elec. Checkbook (C/D) ... $17

TOTL
Totl Text (C) $32 (D) $34
Label (C) $15 (D) $17
Time Mgr. (C) $24 (D) $27
Rsrch Asst. (C) $24 (D) $27

TRONIX
S.A.M. (D) ... $39
Juice (D) ... $23
Chatterbee (D) ... $27

MISCELLANEOUS
Diskey (D) ... $33
Barrons Sat (D) ... $59
Millionaire (D) ... $39
Sargon II (D) ... $23
B-Graph (D) ... $59
Castle Wolfenstein (D) ... $20
Odesta Chess (D) ... $46
Ultima III (D) ... $39
Prof. Blackjack (D) ... $46
Homeword (D) ... $46
Pers. Accountant (D) ... $23
Karate Devils (D) ... $27
Final Flight (D) ... $22
MAE Assembler (D) ... $47

How Does It Work?

The simplest way to see how the program works is to trace through an example. Suppose the instruction LDA #$20 resides at addresses $03C0–$03C1. For this instruction, the extended mnemonic is LDAB, where LDA stands for LoaD Accumulator, and B is the tag code for immediate addressing. The hexadecimal representation for LDA immediate is $A9, which is equivalent to decimal 169.

Line 50, the top of the main loop, calls the keyboard pause routine at line 7000, which also handles changing registers. In line 55, the variable C is loaded with 169 by PEEKing the memory addressed by B, the instruction pointer. The value of B, 960 in this example, is then converted to hexadecimal characters in line 2000 and PRINTed.

In line 60, NOP instructions are POKEd into the execution routine to take up space after one- or two-byte instructions. The hexadecimal value of the opcode is printed next, and then the mnemonic is retrieved from the array R$(). (In the Atari version, mnemonics are stored in the string R$.) If the mnemonic is a blank, this instruction is undefined and an error message is displayed. Otherwise, the standard (three-character) mnemonic is PRINTed, the opcode is POKEd into the execution routine at OP, and the program counter is incremented to 961.

The ASCII code for B is 66, so the ON GOSUB in line 120 transfers control to line 400. Here, the symbol for the addressing mode, #$ is printed. The one-byte operand routine, at line 3000, PEEKs location 961, pointed to by the program counter. This number is POKEd into OP+1, then converted to hexadecimal and PRINTed. After incrementing the program counter to point to the start of the next instruction, a RETURN is executed at line 3000.

At line 5000, the execution routine is SYSed, CALLed, or USRed depending on which computer you have. The contents of the registers are displayed, and control passes back to line 120. Here, a GOTO 50 takes us back to the top of the loop, where the instruction at $3C2 will be executed.

Tracing Is Educational Too

You will find that this program is most useful for testing small ML programs, such as those called as subroutines from BASIC. It's also good for examining sections of larger programs when you're not sure how a particular routine works. If you're learning machine language, you'll find that the register display is an enormous help in understanding the effects and side effects of each instruction, especially the bits (flags) of the processor status register.

Do be careful, though. Any program is vulnerable when dealing with something as powerful as machine language, and this one is no exception. There are more ways to kill a BASIC program from ML than anyone can name in one sitting, so always be conscientious about saving your programs. After you type this one in, SAVE it before you even think about running it. One typographical error could cause the program to erase itself, or at least lock up the computer.

There are also some ML programs that this tracer can't follow, such as those which disconnect the keyboard or video display (whether intentionally or accidentally). If everything is saved on disk or tape (for real security, take the diskette or cassette out of the drive), you can experiment as much as you want, and then if disaster struck all you'd have to do is just turn the computer off and reload the program.

Program 1: Commodore ML Tracer

Refer to the "Automatic Proofreader" article before typing this program in.

```
10 GOSUB6000                            :rem 167
35 POKEA,0:POKEX,0:POKEY,0:POKEP,52:POKES
   ,255                                 :rem 63
40 PRINT"START ADDRESS (HEX)";:H$="C000":
   INPUTH$                              :rem 106
45 H$=RIGHT$(H$,4):GOSUB1500:B=D:PRINT"AN
   Y KEY TO STEP"                       :rem 9
50 GOSUB7000:D=FRE(0)                   :rem 197
55 PRINT:C=PEEK(B):D=B:GOSUB2000:PRINTH$"
   ";                                   :rem 148
60 POKEOP+1,234:POKEOP+2,234            :rem 127
70 D=C:GOSUB2000:PRINTRIGHT$(H$,2)" ";
                                        :rem 170
80 IFR$(C)=""THENPRINT"INVALID OPCODE":PR
   INT:GOTO35                           :rem 229
90 R$=LEFT$(R$(C),3):PRINTR$" ";:POKEOP,C
   :B=B+1                               :rem 175
100 IFR$="BRK"THENPRINT:GOTO35          :rem 141
110 U$=RIGHT$(R$(C),1):IFU$=" "THENGOSUB2
    00:GOTO50                           :rem 126
120 ONASC(U$)-64GOSUB300,400,500,600,700,
    800,900,1000,1100,1200,1300:GOTO50
                                        :rem 156
199 REM{4 SPACES}>IMPLIED MODE<         :rem 42
200 IFR$="RTS"THENGOSUB4000:B=D:GOSUB4000
    :B=D*256+B+1:GOSUB5005:RETURN       :rem 42
203 IFR$<>"RTI"THEN208                  :rem 16
205 GOSUB4000:POKEP,D:GOSUB4000:B=D:GOSUB
    4000:B=D*256+B:GOSUB5005:RETURN
                                        :rem 204
208 IFR$="SEI"ORR$="CLI"THENGOSUB5005:RET
    URN                                 :rem 4
210 GOSUB5000:RETURN                    :rem 242
299 REM{4 SPACES}>ABSOLUTE MODE<        :rem 134
300 PRINT"$";:GOSUB2500                 :rem 68
310 IFR$="JMP"THENB=PEEK(OP+1)+PEEK(OP+2)
    *256:GOSUB5005:RETURN               :rem 34
320 IFR$<>"JSR"THEN340                  :rem 13
330 B=B-1:D=INT(B/256):GOSUB3500:D=B-INT(
    B/256)*256:GOSUB3500                :rem 249
335 B=PEEK(OP+1)+PEEK(OP+2)*256:GOSUB5005
    :RETURN                             :rem 141
340 GOSUB5000:RETURN                    :rem 246
```

```
399 REM{4 SPACES}>IMMEDIATE MODE<:rem 183
400 PRINT"#$";:GOSUB3000:GOSUB5000:RETURN
                                   :rem 253
499 REM{4 SPACES}>ZERO PAGE MODE<:rem 134
500 PRINT"$";:GOSUB3000:GOSUB5000:RETURN
                                   :rem 219
599 REM{4 SPACES}>ABSOLUTE,X<      :rem 232
600 PRINT"$";:GOSUB2500:PRINT",X";:GOSUB5
    000:RETURN                     :rem 170
699 REM{4 SPACES}>ABSOLUTE,Y<      :rem 234
700 PRINT"$";:GOSUB2500:PRINT",Y";:GOSUB5
    000:RETURN                     :rem 172
799 REM{4 SPACES}>(INDIRECT,X)<     :rem 46
800 PRINT"($";:GOSUB3000:PRINT",X)";:GOSU
    B5000:RETURN                   :rem 249
899 REM{4 SPACES}>(INDIRECT),Y<     :rem 48
900 PRINT"($";:GOSUB3000:PRINT"),Y";:GOSU
    B5000:RETURN                   :rem 251
999 REM{4 SPACES}>ZERO PAGE,X<     :rem 234
1000 PRINT"$";:GOSUB3000:PRINT",X";:GOSUB
     5000:RETURN                   :rem 209
1099 REM{3 SPACES}>ZERO PAGE,Y<     :rem 19
1100 PRINT"$";:GOSUB3000:PRINT",Y";:GOSUB
     5000:RETURN                   :rem 211
1199 REM{3 SPACES}>RELATIVE JUMP<:rem 202
1200 PRINT"TO ";:D=PEEK(B):B=B+1:D=D+(D>1
     27)*256:D=B+D:B1=D             :rem 52
1210 GOSUB2000:PRINT"$"H$;:BM=BM(INT(C/64
     )):BC=BMANDPEEK(P)            :rem 254
1220 IFBC=(INT(C/32)AND1)*BMTHENB=B1
                                    :rem 88
1230 GOSUB5005:RETURN               :rem 42
1299 REM{3 SPACES}>INDIRECT JUMP<:rem 193
1300 PRINT"(";:GOSUB2500:PRINT")";:B=PEEK
     (OP+1)+PEEK(OP+2)*256         :rem 118
1310 B=PEEK(B)+PEEK(B+1)*256:GOSUB5005:RE
     TURN                          :rem 160
1499 REM{3 SPACES}> HEX TO DEC < :rem 137
1500 D=0:FORI=1TOLEN(H$):J=ASC(MID$(H$,I,
     1))-48:D=D*H+J+7*(J>9):NEXT:RETURN
                                   :rem 180
1999 REM{3 SPACES}> DEC TO HEX < :rem 142
2000 H$="":FORI=1TO4:E=INT(D/H):J=D-E*H:H
     $=CHR$(J+48-7*(J>9))+H$:D=E:NEXT
                                   :rem 192
2005 RETURN                        :rem 167
2499 REM{3 SPACES}> 2BYTE OPERAND <
                                   :rem 165
2500 D=PEEK(B+1):POKEOP+2,D:GOSUB2000:PRI
     NTRIGHT$(H$,2);:GOSUB3000:B=B+1:RETU
     RN                             :rem 90
2999 REM{3 SPACES}> 1BYTE OPERAND <
                                   :rem 169
3000 D=PEEK(B):POKEOP+1,D:GOSUB2000:PRINT
     RIGHT$(H$,2);:B=B+1:RETURN    :rem 124
3499 REM{3 SPACES}> PUSH <         :rem 119
3500 J=PEEK(S):POKEML+512+J,D      :rem 194
3505 IFJ=0THENPRINT:PRINT"WARNING: STACK
     {SPACE}OVERFLOW":J=256        :rem 114
3510 POKES,J-1:RETURN               :rem 57
3999 REM{3 SPACES}> POP <           :rem 43
4000 J=PEEK(S):D=PEEK(ML+513+J)     :rem 23
4005 IFJ=255THENPRINT:PRINT"WARNING: STAC
     K UNDERFLOW":J=-1             :rem 221
4010 POKES,J+1:RETURN               :rem 51
4999 REM{3 SPACES}> EXECUTE ONE INSTRUCTI
     ON <                          :rem 148
5000 SYSML+23                      :rem 237
5005 PRINT:FORK=0TO4:D=PEEK(A+K):GOSUB200
     0                             :rem 107
5010 PRINTMID$(" A= X= Y= S= P=",3*K+1,3)
     ;:PRINTRIGHT$(H$,2);:NEXT:PRINT:RETU
     RN                            :rem 143
5999 REM{3 SPACES}> INITIAL STUFF <
                                   :rem 208
6000 ML=2*4096+8*256               :rem 245
6001 A=ML+240:X=A+1:Y=X+1:S=Y+1:P=S+1:H=1
     6:OP=ML+92                    :rem 239
6002 DIMR$(255):DIMBM(3):FORI=0TO3:READB:
     BM(I)=B:NEXT                  :rem 204
6003 FORT=0TO255:READR$(T):NEXT   :rem 154
6004 READR$:IFR$<>"END"THENPRINT"ERROR IN
      OPCODES":PRINT"CHECK FOR TYPO'S":EN
     D                             :rem 133
6005 I=0:FORT=MLTOML+164:READB:POKET,B:I=
     I+B:NEXT                      :rem 128
6008 IFI<>17737THENPRINT"ERROR IN ML DATA
     ":PRINT"CHECK FOR TYPO'S":END:rem 36
6010 SYSML                          :rem 95
6015 PRINT"{CLR}{7 DOWN}{5 RIGHT}6502 ML
     {SPACE}TRACER{4 DOWN}"        :rem 163
6020 RETURN                        :rem 168
6999 REM{2 SPACES}> PAUSE <        :rem 189
7000 GETA$:IFA$=""THEN7000         :rem 177
7010 IFA$="I"THEND=B:L=4:GOSUB7100:B=D:GO
     TO7000                         :rem 40
7020 IFA$="A"THEND=PEEK(A):L=2:GOSUB7100:
     POKEA,D:GOTO7000              :rem 177
7030 IFA$="X"THEND=PEEK(X):L=2:GOSUB7100:
     POKEX,D:GOTO7000              :rem 247
7040 IFA$="Y"THEND=PEEK(Y):L=2:GOSUB7100:
     POKEY,D:GOTO7000              :rem 251
7050 IFA$="S"THEND=PEEK(S):L=2:GOSUB7100:
     POKES,D:GOTO7000              :rem 234
7060 IFA$="P"THEND=PEEK(P):L=2:GOSUB7100:
     POKEP,D:GOTO7000              :rem 226
7070 RETURN                        :rem 174
7100 PRINTA$"=";:GOSUB2000:INPUTH$:H$=RIG
     HT$(H$,L):GOSUB1500:RETURN   :rem 124
9000 DATA128,64,1,2                :rem 207
10000 DATABRK ,ORAF,,,,ORAC,ASLC,:rem 142
10001 DATAPHP ,ORAB,ASL ,,,ORAA,ASLA,
                                   :rem 112
10002 DATABPLJ,ORAG,,,,ORAH,ASLH,:rem 228
10003 DATACLC ,ORAE,,,,ORAD,ASLD,:rem 133
10004 DATAJSRA,ANDF,,,BITC,ANDC,ROLC,
                                   :rem 244
10005 DATAPLP ,ANDB,ROL ,,BITA,ANDA,ROLA,
                                   :rem 148
10006 DATABMIJ,ANDG,,,,ANDH,ROLH,:rem 209
10007 DATASEC ,ANDE,,,,AMDD,ROLD,:rem 128
10008 DATARTI ,EORF,,,,EORC,LSRC,:rem 191
10009 DATAPHA ,EORB,LSR ,,JMPA,EORA,LSRA,
                                   :rem 187
10010 DATABVCJ,EORG,,,,EORH,LSRH,:rem 249
10011 DATACLI ,EORE,,,,EORD,LSRD,:rem 163
10012 DATARTS ,ADCF,,,,ADCC,RORC,:rem 138
10013 DATAPLA ,ADCB,ROR ,,JMPK,ADCA,RORA,
                                   :rem 140
10014 DATABVSJ,ADCG,,,,ADCH,RORH,:rem 211
10015 DATASEI ,ADCE,,,,ADCD,RORD,:rem 118
10016 DATA,STAF,,,STYC,STAC,STXC, :rem 36
10017 DATADEY ,,TXA ,,STYA,STAA,STXA,
                                   :rem 192
10018 DATABCCJ,STAG,,,STYH,STAH,STXI,
                                    :rem 73
10019 DATATYA ,STAE,TXS ,,,STAD,,:rem 143
10020 DATALDYB,LDAF,LDXB,,LDYC,LDAC,LDXC,
                                    :rem 24
10021 DATATAY ,LDAB,TAX ,,LDYA,LDAA,LDXA,
                                   :rem 149
10022 DATABCSJ,LDAG,,,LDYH,LDAH,LDXI,
                                   :rem 248
```

```
10023 DATACLV ,LDAE,TSX ,,LDYD,LDAD,LDXE,
                                    :rem 173
10024 DATACPYB,CMPF,,,CPYC,CMPC,DECC,
                                    :rem 250
10025 DATAINY ,CMPB,DEX ,,CPYA,CMPA,DECA,
                                    :rem 148
10026 DATABNEJ,CMPG,,,,CMPH,DECH,:rem 201
10027 DATACLD ,CMPE,,,,CMPD,DECD,:rem 116
10028 DATACPXB,SBCF,,,CPXC,SBCC,INCC,
                                    :rem 250
10029 DATAINX ,SBCB,NOP ,,CPXA,SBCA,INCA,
                                    :rem 160
10030 DATABEQJ,SBCG,,,,SBCI,INCI,:rem 199
10031 DATASED ,SBCE,,,,SBCD,INCD,:rem 118
10032 DATAEND                      :rem 231
20000 DATA162,0,181,0,157,0,41,189
                                    :rem 167
20001 DATA0,1,157,0,42,189,0,2    :rem 217
20002 DATA157,0,43,232,208,236,96,120
                                    :rem 68
20003 DATA162,0,181,0,168,189,0,41
                                    :rem 172
20004 DATA149,0,152,157,0,41,189,0
                                    :rem 174
20005 DATA1,168,189,0,42,157,0,1  :rem 75
20006 DATA152,157,0,42,189,0,2,168
                                    :rem 180
20007 DATA189,0,43,157,0,2,152,157
                                    :rem 180
20008 DATA0,43,232,208,213,186,138,174
                                    :rem 125
20009 DATA243,40,154,141,243,40,172,242
                                    :rem 165
20010 DATA40,174,241,40,173,244,40,72
                                    :rem 62
20011 DATA173,240,40,40,234,234,234,8
                                    :rem 62
20012 DATA141,240,40,104,141,244,40,142
                                    :rem 147
20013 DATA241,40,140,242,40,186,138,174
                                    :rem 167
20014 DATA243,40,154,141,243,40,162,0
                                    :rem 56
20015 DATA181,0,168,189,0,41,149,0
                                    :rem 180
20016 DATA152,157,0,41,189,0,1,168
                                    :rem 179
20017 DATA189,0,42,157,0,1,152,157
                                    :rem 179
20018 DATA0,42,189,0,2,168,189,0  :rem 84
20019 DATA43,157,0,2,152,157,0,43:rem 124
20020 DATA232,208,213,88,96       :rem 100
```

Program 2: Apple ML Tracer

```
10  GOSUB 6000
35  POKE A,0: POKE X,0: POKE Y,0:
      POKE P,52: POKE S,255
40  PRINT "START ADDRESS (HEX)";:
      INPUT H$
42  IF H$ = "" THEN H$ = "C000"
45 H$ =  RIGHT$ (H$,4): GOSUB 150
      0:B = D: PRINT "ANY KEY TO S
      TEP"
50  GOSUB 7000:D =  FRE (0)
55  PRINT :C =  PEEK (B):D = B: GOSUB
      2000: PRINT H$" ";
60  POKE OP + 1,234: POKE OP + 2,
      234
70 D = C: GOSUB 2000: PRINT  RIGHT$
      (H$,2)" ";
80  IF R$(C) = "" THEN  PRINT "IN
      VALID OPCODE": PRINT : GOTO
      35
90 R$ =  LEFT$ (R$(C),3): PRINT R
      $" ";: POKE OP,C:B = B + 1
100  IF R$ = "BRK" THEN  PRINT : GOTO
      35
110 U$ =  RIGHT$ (R$(C),1): IF U$
      = " " THEN  GOSUB 200: GOTO
      50
120  ON  ASC (U$) - 64 GOSUB 300,
      400,500,600,700,800,900,1000
      ,1100,1200,1300: GOTO 50
199  REM    >IMPLIED MODE<
200  IF R$ = "RTS" THEN  GOSUB 40
      00:B = D: GOSUB 4000:B = D *
      256 + B + 1: GOSUB 5005: RETURN
203  IF R$ <  > "RTI" THEN 208
205  GOSUB 4000: POKE P,D: GOSUB
      4000:B = D: GOSUB 4000:B = D
       * 256 + B: GOSUB 5005: RETURN
208  IF R$ = "SEI" OR R$ = "CLI" THEN
      GOSUB 5005: RETURN
210  GOSUB 5000: RETURN
299  REM    >ABSOLUTE MODE<
300  PRINT "$";: GOSUB 2500
310  IF R$ = "JMP" THEN B =  PEEK
      (OP + 1) +  PEEK (OP + 2) *
      256: GOSUB 5005: RETURN
320  IF R$ <  > "JSR" THEN 340
330 B = B - 1:D =  INT (B / 256):
      GOSUB 3500:D = B -  INT (B /
      256) * 256: GOSUB 3500
335 B =  PEEK (OP + 1) +  PEEK (O
      P + 2) * 256: GOSUB 5005: RETURN
340  GOSUB 5000: RETURN
399  REM    >IMMEDIATE MODE<
400  PRINT "#$";: GOSUB 3000: GOSUB
      5000: RETURN
499  REM    >ZERO PAGE MODE<
500  PRINT "$";: GOSUB 3000: GOSUB
      5000: RETURN
599  REM    >ABSOLUTE,X<
600  PRINT "$";: GOSUB 2500: PRINT
      ",X";: GOSUB 5000: RETURN
699  REM    >ABSOLUTE,Y<
700  PRINT "$";: GOSUB 2500: PRINT
      ",Y";: GOSUB 5000: RETURN
799  REM    >(INDIRECT,X)<
800  PRINT "($";: GOSUB 3000: PRINT
      ",X)";: GOSUB 5000: RETURN
899  REM    >(INDIRECT),Y<
900  PRINT "($";: GOSUB 3000: PRINT
      "),Y";: GOSUB 5000: RETURN
999  REM    >ZERO PAGE,X<
1000  PRINT "$";: GOSUB 3000: PRINT
      ",X";: GOSUB 5000: RETURN
1099  REM    >ZERO PAGE,Y<
1100  PRINT "$";: GOSUB 3000: PRINT
      ",Y";: GOSUB 5000: RETURN
1199  REM    >RELATIVE JUMP<
1200  PRINT "TO ";:D =  PEEK (B):
      B = B + 1:D = D - (D > 127) *
      256:D = B + D:B1 = D
1210  GOSUB 2000: PRINT "$"H$;:BM
      = BM( INT (C / 64)):BC =  INT
      ( PEEK (P) / BM):BC = BC - 2
      *  INT (BC / 2)
1220  IF BC = ( INT (C / 32) - 2 *
      INT (C / 64)) THEN B = B1
```

```
1230  GOSUB 5005: RETURN
1299  REM    >INDIRECT JUMP<
1300  PRINT "(";: GOSUB 2500: PRINT
      ")";:B =  PEEK (OP + 1) +  PEEK
      (OP + 2) * 256
1310 B =  PEEK (B) +  PEEK (B + 1
      ) * 256: GOSUB 5005: RETURN
1499  REM    > HEX TO DEC <
1500 D = 0: FOR I = 1 TO  LEN (H$
      ):J =  ASC ( MID$ (H$,I,1)) -
      48:D = D * H + J - 7 * (J >
      9): NEXT : RETURN
1999  REM    > DEC TO HEX <
2000 H$ = "": FOR I = 1 TO 4:E =
      INT (D / H):J = D - E * H:H
      $ =  CHR$ (J + 48 + 7 * (J >
      9)) + H$:D = E: NEXT
2005 RETURN
2499  REM    > 2BYTE OPERAND <
2500 D =  PEEK (B + 1): POKE OP +
      2,D: GOSUB 2000: PRINT  RIGHT$
      (H$,2);: GOSUB 3000:B = B +
      1: RETURN
2999  REM    > 1BYTE OPERAND <
3000 D =  PEEK (B): POKE OP + 1,D
      : GOSUB 2000: PRINT  RIGHT$
      (H$,2);:B = B + 1: RETURN
3499  REM    > PUSH <
3500 J =  PEEK (S): POKE ML + 512
      + J,D
3505  IF J = 0 THEN  PRINT : PRINT
      "WARNING: STACK OVERFLOW":J =
      256
3510  POKE S,J - 1: RETURN
3999  REM    > POP <
4000 J =  PEEK (S):D =  PEEK (ML +
      513 + J)
4005  IF J = 255 THEN  PRINT : PRINT
      "WARNING: STACK UNDERFLOW":J
      =  - 1
4010  POKE S,J + 1: RETURN
4999  REM    > EXECUTE ONE INSTRU
      CTION <
5000  CALL (ML + 23)
5005  PRINT : FOR K = 0 TO 4:D =
      PEEK (A + K): GOSUB 2000
5010  PRINT  MID$ (" A= X= Y= S=
      P=",3 * K + 1,3);: PRINT  RIGHT$
      (H$,2);: NEXT : PRINT : RETURN
5999  REM    > INITIAL STUFF <
6000 ML = 2 * 4096 + 8 * 256
6001 A = ML + 240:X = A + 1:Y = X
      + 1:S = Y + 1:P = S + 1:H =
      16:OP = ML + 92
6002  DIM R$(255): DIM BM(3): FOR
      I = 0 TO 3: READ B:BM(I) = B
      : NEXT
6003  FOR T = 0 TO 255: READ R$(T
      ): NEXT
6004  READ R$: IF R$ <  > "END" THEN
      PRINT "ERROR IN OPCODES": PRINT
      "CHECK FOR TYPO'S": END
6005 I = 0: FOR T = ML TO ML + 16
      4: READ B: POKE T,B:I = I +
      B: NEXT
6008  IF I <  > 17737 THEN  PRINT
      "ERROR IN ML DATA": PRINT "C
      HECK FOR TYPO'S": END
6010  CALL ML
6015  HOME : PRINT "6502 ML TRACE
      R"
6020  RETURN
6999  REM    > PAUSE <
7000  GET A$: IF A$ = "" THEN 700
      0
7010  IF A$ = "I" THEN D = B:L =
      4: GOSUB 7100:B = D: GOTO 70
      00
7020  IF A$ = "A" THEN D =  PEEK
      (A):L = 2: GOSUB 7100: POKE
      A,D: GOTO 7000
7030  IF A$ = "X" THEN D =  PEEK
      (X):L = 2: GOSUB 7100: POKE
      X,D: GOTO 7000
7040  IF A$ = "Y" THEN D =  PEEK
      (Y):L = 2: GOSUB 7100: POKE
      Y,D: GOTO 7000
7050  IF A$ = "S" THEN D =  PEEK (S)
      :L = 2: GOSUB 7100: POKE S,D: GOTO
      7000
7060  IF A$ = "P" THEN D =  PEEK
      (P):L = 2: GOSUB 7100: POKE
      P,D: GOTO 7000
7065  IF A$ =  CHR$ (3) THEN  STOP

7070  RETURN
7100  PRINT A$"=";: GOSUB 2000:A$
      = H$: INPUT H$: IF H$ = "" THEN
      H$ = A$
7110 H$ =  RIGHT$ (H$,L): GOSUB 1
      500: RETURN
9000  DATA 128,64,1,2
10000  DATA BRK ,ORAF,,,,ORAC,ASL
      C,
10001  DATA PHP ,ORAB,ASL ,,,ORAA
      ,ASLA,
10002  DATA BPLJ,ORAG,,,,ORAH,ASL
      H,
10003  DATA CLC ,ORAE,,,,ORAD,ASL
      D,
10004  DATA JSRA,ANDF,,,BITC,ANDC
      ,ROLC,
10005  DATA PLP ,ANDB,ROL ,,BITA,
      ANDA,ROLA,
10006  DATA BMIJ,ANDG,,,,ANDH,ROL
      H,
10007  DATA SEC ,ANDE,,,,AMDD,ROL
      D,
10008  DATA RTI ,EORF,,,,EORC,LSR
      C,
10009  DATA PHA ,EORB,LSR ,,JMPA,
      EORA,LSRA,
10010  DATA BVCJ,EORG,,,,EORH,LSR
      H,
10011  DATA CLI ,EORE,,,,EORD,LSR
      D,
10012  DATA RTS ,ADCF,,,,ADCC,ROR
      C,
10013  DATA PLA ,ADCB,ROR ,,JMPK,
      ADCA,RORA,
10014  DATA BVSJ,ADCG,,,,ADCH,ROR
      H,
10015  DATA SEI ,ADCE,,,,ADCD,ROR
      D,
10016  DATA ,STAF,,,STYC,STAC,STX
      C,
10017  DATA DEY ,,TXA ,,STYA,STAA
      ,STXA,
```

```
10018  DATA BCCJ,STAG,,,STYH,STAH
     ,STXI,
10019  DATA TYA ,STAE,TXS ,,,STAD
     ,,
10020  DATA LDYB,LDAF,LDXB,,LDYC,
     LDAC,LDXC,
10021  DATA TAY ,LDAB,TAX ,,LDYA,
     LDAA,LDXA,
10022  DATA BCSJ,LDAG,,,LDYH,LDAH
     ,LDXI,
10023  DATA CLV ,LDAE,TSX ,,LDYD,
     LDAD,LDXE,
10024  DATA CPYB,CMPF,,,CPYC,CMPC
     ,DECC,
10025  DATA INY ,CMPB,DEX ,,CPYA,
     CMPA,DECA,
10026  DATA BNEJ,CMPG,,,,CMPH,DEC
     H,
10027  DATA CLD ,CMPE,,,,CMPD,DEC
     D,
10028  DATA CPXB,SBCF,,,CPXC,SBCC
     ,INCC,
10029  DATA INX ,SBCB,NOP ,,CPXA,
     SBCA,INCA,
10030  DATA BEQJ,SBCG,,,,SBCI,INC
     I,
10031  DATA SED ,SBCE,,,,SBCD,INC
     D,
10032  DATA END
20000  DATA 162,0,181,0,157,0,41,
     189
20001  DATA 0,1,157,0,42,189,0,2

20002  DATA 157,0,43,232,208,236,
     96,120
20003  DATA  162,0,181,0,168,189,
     0,41
20004  DATA 149,0,152,157,0,41,18
     9,0
20005  DATA 1,168,189,0,42,157,0,
     1
20006  DATA 152,157,0,42,189,0,2,
     168
20007  DATA 189,0,43,157,0,2,152,
     157
20008  DATA 0,43,232,208,213,186,
     138,174
20009  DATA 243,40,154,141,243,40
     ,172,242
20010  DATA 40,174,241,40,173,244
     ,40,72
20011  DATA 173,240,40,40,234,234
     ,234,8
20012  DATA 141,240,40,104,141,24
     4,40,142
20013  DATA 241,40,140,242,40,186
     ,138,174
20014  DATA  243,40,154,141,243,4
     0,162,0
20015  DATA 181,0,168,189,0,41,14
     9,0
20016  DATA 152,157,0,41,189,0,1,
     168
20017  DATA 189,0,42,157,0,1,152,
     157
20018  DATA 0,42,189,0,2,168,189,
     0
20019  DATA 43,157,0,2,152,157,0,
     43
20020  DATA 232,208,213,88,96
```

Program 3: Atari ML Tracer

Refer to the "Automatic Proofreader" article before typing this program in.

```
KH 10 GOSUB 6000
DP 35 POKE A,0:POKE X,0:POKE Y,0:POK
      E P,52:POKE S,255
HA 40 PRINT "START ADDRESS (HEX)";:I
      NPUT H$
JC 42 I=LEN(H$)-3:IF I<1 THEN I=1:IF
       NOT LEN(H$) THEN H$="C000"
EE 45 H$=H$(I):GOSUB 1500:B=D:PRINT
      "HIT RETURN TO STEP"
MF 50 GOSUB 7000:D=FRE(0)
MP 55 PRINT :C=PEEK(B):D=B:GOSUB 200
      0:PRINT H$;" ";
HP 60 POKE OP+1,234:POKE OP+2,234
BI 70 D=C:GOSUB 2000:PRINT H$(3);" "
      ;
OO 75 O$=R$(C*4+1,C*4+3):U$=R$(C*4+4
      ,C*4+4)
EO 80 IF O$="{3 SPACES}" THEN PRINT
      "INVALID OPCODE":PRINT :GOTO 3
      5
PB 90 PRINT O$;" ";:POKE OP,C:B=B+1
IK 100 IF O$="BRK" THEN PRINT :GOTO
       35
DE 110 IF U$=" " THEN GOSUB 200:GOTO
        50
JM 120 ON ASC(U$)-64 GOSUB 300,400,5
       00,600,700,800,900,1000,1100,
       1200,1300:GOTO 50
CK 199 REM >IMPLIED MODE<
CH 200 IF O$="RTS" THEN GOSUB 4000:B
       =D:GOSUB 4000:B=D*256+B+1:GOS
       UB 5005:RETURN
AN 203 IF O$<>"RTI" THEN 208
MM 205 GOSUB 4000:POKE P,D:GOSUB 400
       0:B=D:GOSUB 4000:B=B*256+D:GO
       SUB 5005:RETURN
PO 208 IF O$="SEI" OR O$="CLI" THEN
       GOSUB 5005:RETURN
PC 210 GOSUB 5000:RETURN
IG 299 REM >ABSOLUTE MODE<
EE 300 PRINT "$";:GOSUB 2500
BP 310 IF O$="JMP" THEN B=PEEK(OP+1)
       +PEEK(OP+2)*256:GOSUB 5005:RE
       TURN
AK 320 IF O$<>"JSR" THEN 340
PJ 330 B=B-1:D=INT(B/256):GOSUB 3500
       :D=B-INT(B/256)*256:GOSUB 350
       0
IN 335 B=PEEK(OP+1)+PEEK(OP+2)*256:G
       OSUB 5005:RETURN
PG 340 GOSUB 5000:RETURN
LH 399 REM >IMMEDIATE MODE<
PN 400 PRINT "#$";:GOSUB 3000:GOSUB
       5000:RETURN
IG 499 REM >ZERO PAGE MODE<
NL 500 PRINT "$";:GOSUB 3000:GOSUB 5
       000:RETURN
OI 599 REM >ABSOLUTE,X<
KK 600 PRINT "$";:GOSUB 2500:PRINT "
       ,X";:GOSUB 5000:RETURN
OK 699 REM >ABSOLUTE,Y<
KM 700 PRINT "$";:GOSUB 2500:PRINT "
       ,Y";:GOSUB 5000:RETURN
CO 799 REM >(INDIRECT,X)<
PJ 800 PRINT "($";:GOSUB 3000:PRINT
       ",X)";:GOSUB 5000:RETURN
```

```
DA 899 REM >(INDIRECT),Y<
PL 900 PRINT "($";:GOSUB 3000:PRINT
       "),Y";:GOSUB 5000:RETURN
OK 999 REM >ZERO PAGE,X<
NB 1000 PRINT "$";:GOSUB 3000:PRINT
        ",X";:GOSUB 5000:RETURN
BD 1099 REM >ZERO PAGE,Y<
ND 1100 PRINT "$";:GOSUB 3000:PRINT
        ",Y";:GOSUB 5000:RETURN
MK 1199 REM >RELATIVE JUMP<
DG 1200 PRINT "TO ";:D=PEEK(B):B=B+1
        :D=D-(D>127)*256:D=B+D:B1=D
PN 1210 GOSUB 2000:PRINT "$";H$;:BM=
        BM(INT(C/64)):BC=INT(PEEK(P)
        /BM):BC=BC-2*INT(BC/2)
DM 1220 IF BC=(INT(C/32)-2*INT(C/64)
        ) THEN B=B1
CK 1230 GOSUB 5005:RETURN
MB 1299 REM >INDIRECT JUMP<
AJ 1300 PRINT "(";:GOSUB 2500:PRINT
        ")";:B=PEEK(OP+1)+PEEK(OP+2)
        *0
KA 1310 B=PEEK(B)+PEEK(B+1)*256:GOSU
        B 5005:RETURN
IJ 1499 REM > HEX TO DEC <
ON 1500 D=0:FOR I=1 TO LEN(H$):J=ASC
        (H$(I,I))-48:D=D*H+J-7*(J>9)
        :NEXT I:RETURN
IO 1999 REM > DEC TO HEX <
JD 2000 H$="":FOR I=1 TO 4:E=INT(D/H
        ):J=D-E*H:I$=H$:H$=CHR$(J+48
        +7*(J>9)):H$(2)=I$:D=E:NEXT I
KH 2005 RETURN
KF 2499 REM > 2BYTE OPERAND <
IN 2500 D=PEEK(B+1):POKE OP+2,D:GOSU
        B 2000:PRINT H$(3);:GOSUB 30
        00:B=B+1:RETURN
KJ 2999 REM > 1BYTE OPERAND <
KP 3000 D=PEEK(B):POKE OP+1,D:GOSUB
        2000:PRINT H$(3);:B=B+1:RETU
        RN
HH 3499 REM > PUSH <
MC 3500 J=PEEK(S):POKE ML+512+J,D
HC 3505 IF J=0 THEN PRINT :PRINT "WA
        RNING: STACK OVERFLOW":J=256
DJ 3510 POKE S,J-1:RETURN
CL 3999 REM > POP <
BH 4000 J=PEEK(S):D=PEEK(ML+513+J)
NN 4005 IF J=255 THEN PRINT :PRINT "
        WARNING: STACK UNDERFLOW":J=
        -1
DD 4010 POKE S,J+1:RETURN
JE 4999 REM > EXECUTE ONE INSTRUCTIO
        N <
FJ 5000 POKE 54286,0
LM 5001 D=USR(ML+24)
JF 5002 POKE 54286,64
GL 5005 PRINT :FOR K=0 TO 4:D=PEEK(A
        +K):GOSUB 2000
KO 5010 PRINT REG$(3*K+1,3*K+3);:PRI
        NT H$(3);:NEXT K:PRINT :RETU
        RN
NA 5999 REM > INITIAL STUFF <
MP 6000 ML=6*4096
PC 6020 A=ML+240:X=A+1:Y=X+1:S=Y+1:P
        =S+1:H=16:OP=ML+94
CB 6030 DIM R$(1024),H$(12),I$(12),O
        $(3),U$(1),REG$(15),BM(3):FO
        R I=0 TO 3:READ B:BM(I)=B:NE
        XT I
BH 6035 REG$=" A= X= Y= S= P="
EF 6040 FOR T=0 TO 255:READ H$:IF H$
        ="" THEN H$="{4 SPACES}"
BK 6045 R$(T*4+1)=H$:NEXT T
HC 6050 READ H$:IF H$<>"END" THEN PR
        INT "ERROR IN OPCODES":PRINT
         "CHECK FOR TYPO'S":END
NH 6060 I=0:FOR T=ML TO ML+166:READ
        B:POKE T,B:I=I+B:NEXT T
CE 6070 IF I<>19457 THEN PRINT "ERRO
        R IN ML DATA":PRINT "CHECK F
        OR TYPO'S":END
DD 6080 D=USR(ML)
MH 6090 PRINT "6502 ML TRACER"
KH 6100 RETURN
LN 6999 REM > PAUSE <
MD 7000 INPUT H$
CP 7010 IF H$="I" THEN D=B:L=4:GOSUB
         7100:B=D:GOTO 7000
LI 7020 IF H$="A" THEN D=PEEK(A):L=2
        :GOSUB 7100:POKE A,D:GOTO 70
        00
PO 7030 IF H$="X" THEN D=PEEK(X):L=2
        :GOSUB 7100:POKE X,D:GOTO 70
        00
AC 7040 IF H$="Y" THEN D=PEEK(Y):L=2
        :GOSUB 7100:POKE Y,D:GOTO 70
        00
PB 7050 IF H$="S" THEN D=PEEK(S):L=2
        :GOSUB 7100:POKE S,D:GOTO 70
        00
OJ 7060 IF H$="P" THEN D=PEEK(P):L=2
        :GOSUB 7100:POKE P,D:GOTO 70
        00
KO 7070 RETURN
OJ 7100 PRINT H$;"=";:INPUT H$:I=LEN
        (H$)-L+1:IF I<1 THEN I=1:IF
         NOT LEN(H$) THEN RETURN
BD 7120 H$=H$(I):GOSUB 1500:RETURN
MP 9000 DATA 128,64,1,2
IO 10000 DATA BRK ,ORAF,,,,ORAC,ASLC
         ,
HA 10001 DATA PHP ,ORAB,ASL ,,,ORAA,
         ASLA,
OE 10002 DATA BPLJ,ORAG,,,,ORAH,ASLH
         ,
IF 10003 DATA CLC ,ORAE,,,,ORAD,ASLD
         ,
PE 10004 DATA JSRA,ANDF,,,BITC,ANDC,
         ROLC,
JE 10005 DATA PLP ,ANDB,ROL ,,BITA,A
         NDA,ROLA,
NB 10006 DATA BMIJ,ANDG,,,,ANDH,ROLH
         ,
IA 10007 DATA SEC ,ANDE,,,,AMDD,ROLD
         ,
LP 10008 DATA RTI ,EORF,,,,EORC,LSRC
         ,
LL 10009 DATA PHA ,EORB,LSR ,,JMPA,E
         ORA,LSRA,
PJ 10010 DATA BVCJ,EORG,,,,EORH,LSRH
         ,
KD 10011 DATA CLI ,EORE,,,,EORD,LSRD
         ,
IK 10012 DATA RTS ,ADCF,,,,ADCC,RORC
         ,
IM 10013 DATA PLA ,ADCB,ROR ,,JMPK,A
         DCA,RORA,
ND 10014 DATA BVSJ,ADCG,,,,ADCH,RORH
         ,
HG 10015 DATA SEI ,ADCE,,,,ADCD,RORD
```

```
CE 10016 DATA ,STAF,,,STYC,STAC,STXC
         ,
MA 10017 DATA DEY ,,TXA ,,STYA,STAA,
         STXA,
EJ 10018 DATA BCCJ,STAG,,,STYH,STAH,
         STXI,
IP 10019 DATA TYA ,STAE,TXS ,,,STAD,
         ,
BI 10020 DATA LDYB,LDAF,LDXB,,LDYC,L
         DAC,LDXC,
JF 10021 DATA TAY ,LDAB,TAX ,,LDYA,L
         DAA,LDXA,
PI 10022 DATA BCSJ,LDAG,,,LDYH,LDAH,
         LDXI,
KN 10023 DATA CLV ,LDAE,TSX ,,LDYD,L
         DAD,LDXE,
PK 10024 DATA CPYB,CMPF,,,CPYC,CMPC,
         DECC,
JE 10025 DATA INY ,CMPB,DEX ,,CPYA,C
         MPA,DECA,
MJ 10026 DATA BNEJ,CMPG,,,,CMPH,DECH
         ,
HE 10027 DATA CLD ,CMPE,,,,CMPD,DECD
         ,
PK 10028 DATA CPXB,SBCF,,,CPXC,SBCC,
         INCC,
KA 10029 DATA INX ,SBCB,NOP ,,CPXA,S
         BCA,INCA,
MH 10030 DATA BEQJ,SBCG,,,,SBCI,INCI
         ,
HG 10031 DATA SED ,SBCE,,,,SBCD,INCD
         ,
OH 10032 DATA END
KF 20000 DATA 104,162,0,181,0,157,0,
         97
FE 20001 DATA 189,0,1,157,0,98,189,0
DO 20002 DATA 2,157,0,99,232,208,236
         ,96
DP 20003 DATA 120,104,162,0,181,0,16
         8,189
IH 20004 DATA 0,97,149,0,152,157,0,9
         7
MH 20005 DATA 189,0,1,168,189,0,98,1
         57
EP 20006 DATA 0,1,152,157,0,98,189,0
FL 20007 DATA 2,168,189,0,99,157,0,2
IF 20008 DATA 152,157,0,99,232,208,2
         13,186
MB 20009 DATA 138,174,243,96,154,141
         ,243,96
LJ 20010 DATA 172,242,96,174,241,96,
         173,244
FA 20011 DATA 96,72,173,240,96,40,23
         4,234
HE 20012 DATA 234,8,141,240,96,104,1
         41,244
IL 20013 DATA 96,142,241,96,140,242,
         96,186
LN 20014 DATA 138,174,243,96,154,141
         ,243,96
LK 20015 DATA 162,0,181,0,168,189,0,
         97
LM 20016 DATA 149,0,152,157,0,97,189
         ,0
FJ 20017 DATA 1,168,189,0,98,157,0,1
MC 20018 DATA 152,157,0,98,189,0,2,1
         68
MC 20019 DATA 189,0,99,157,0,2,152,1
         57
FO 20020 DATA 0,99,232,208,213,88,96 ©
```

C. Regena

The Singing Computer

If a computer can speak and can play music, can it sing? This month, I'll try to make the TI sing. First, to make the computer talk you need the TI Speech Synthesizer, a small peripheral device that attaches to the right side of the console. To use the speech synthesizer, you also need a command module that is made to provide speech.

To do your *own* programming with speech, you also need a command module. Right now the modules available are Speech Editor, TI Extended BASIC, and Terminal Emulator II. Terminal Emulator II is the easiest to work with because you can type any word in and the computer will pronounce it phonetically. Speech Editor and Extended BASIC use CALL SAY commands and have limited vocabularies.

I've had several letters from people wondering why certain phrases don't work. To make the computer say a phrase, such as Texas Instruments, use the number sign (SHIFT 3) before the phrase. For example, CALL SAY ("#Texas Instruments").

Unlimited Speech

A bit of history here—the original speech synthesizer was designed to use the words in the Speech Editor and Extended BASIC lists. Inserts were going to be made available that had different vocabularies (that's why some of the speech synthesizers have a lift-up lid). Then the Terminal Emulator II command module was invented, which provides unlimited speech, and inserts to the synthesizer were no longer needed.

Extended BASIC has also gone through at least one revision. I assume there are very few of the original version around because most users exchanged the original module for the second version as soon as they could. The first version did not support repeating keys and was notorious for "locking up" the computer. There were also some problems with using IMAGE statements.

The Terminal Emulator II command module has a dual purpose. In fact, it's called Terminal Emulator II because it is used to make your TI act as a terminal for another computer. For telecommunications you can use your TI-99/4A with an RS-232 Interface and a telephone modem, plus the Terminal Emulator II command module.

Pages 33–42 of the Terminal Emulator II instruction manual describe how to use speech. There are two main ways to use speech, "text-to-speech" and allophone speech. I use the text-to-speech method because all you have to do is spell the text phonetically. The allophone speech can be more exact because you can specify certain sounds. The manual contains a list of allophone numbers with their sounds plus a few sample programs of how to use this method.

Singing Requires Experimentation

Working with speech in a program takes a lot of experimentation. First, you need to try different spellings to get the computer to properly say what you want it to say. Then you can try different inflection symbols, ^, _, and >. These are used to change inflections and stress points, but they can also change the tone of the voice. You can also add different pause symbols for different sounds and contours. These symbols are the comma, period, semicolon, colon, exclamation point, question mark, and space. Finally, you can alter the pitch and slope—this is what I do to make the computer sing.

To create speech, you need the following statement:

```
OPEN #1:"SPEECH",OUTPUT
```

You may use any number after the number sign, just as in opening other types of files. Later, when you want the computer to speak, just use a command such as

PRINT #1:"MY NAME IS SINNDY."

The pitch is how high or low the voice sounds and can be a number from 0 through 63. Zero is a whisper, 1 is the highest pitched voice, and 63 is the lowest pitched voice. The slope is the rate at which the pitch changes in a spoken phrase. The slope may be a number from 0 through 255. For the best results, the manual recommends a slope 3.2 times the pitch. There are certain combinations of pitch and slope that will not be accepted. The default values of pitch and slope are 43 and 128. To change the pitch and slope, use the format *//xx yyy* where *xx* is the pitch period and *yyy* is the slope level. There must be a space between the numbers. An example in a program statement would be:

PRINT #1:"//30 96"

Changing The Pitch

The following is a sample program that illustrates how the pitch and slope change the sound of the voice. I am trying different pitches from 0 to 63 (and STEPping by 2 so it won't take forever). The slope S is calculated by taking the recommended factor of 3.2 times the pitch. Remember, you may try different slopes if you prefer. B$ combines the double slashes with the pitch, a space, and the slope, so line 170 can set the pitch and slope. Line 180 then speaks the phrase.

```
100 REM PITCH AND SLOPE
110 CALL CLEAR
120 OPEN #5:"SPEECH",OUTPUT
130 FOR P=0 TO 63 STEP 2
140 S=INT(P*3.2+.5)
150 B$="//"&STR$(P)&" "&STR$(S)
160 PRINT B$
170 PRINT #5:B$
180 PRINT #5:"TRY THIS TEST."
190 NEXT P
200 END
```

Since other statements can be executed while a sound is playing, you can play a tone, then say a word. By changing the pitch and slope numbers for the speech, you can make the voice go higher or lower, and program a singing computer.

Remember—I mentioned that working with speech involves a lot of experimentation. Singing takes even more time because there are many parameters that vary with each new tone. After you change the pitch and slope, you can try the inflection symbols and the punctuation marks to vary the voice even more. The TI with Terminal Emulator II can really create synthesized speech that sounds pretty good.

Teaching The ABCs

"Alphabet Song" illustrates simple singing on the computer. However, I did not spend a lot of time fiddling with the program and trying different things to make the speech sound better. You may want to try spelling out the letter as a word, and you may want to add the inflection symbols and punctuation marks. I used different pitches for the singing, but kept the slope numbers just 3.2 times the pitch. You could vary these numbers to get a more human sound and a better singing voice.

My little boy has played a lot with the Early Learning Fun command module. One section teaches the letters of the alphabet, and the child finds the letters on the keyboard. My son is quite proficient at this and knows the names of the letters, but I realized he'd learned them in a random order. Most children learn the alphabet from the ABC song, but I had never sung it to him. I decided I'd let the computer sing it to him.

Lowercase letters are used in the program because my son already knows the capital letters and really needs a little more practice with the lowercase letters. Schoolteachers often recommend learning the lowercase letters right along with the capital letters, and all beginning reading is in lowercase letters.

Lines 120–200 define the lowercase letters. If you have saved the lowercase letters program from my August 1983 column, you can load that program, delete the PRINTing lines, then continue typing this program. If you have any problems running this program, the most likely cause is in typing the data in lines 160–200. Your actual error message will cite line 130 or line 140, but those lines are dependent upon the DATA statements. Do not type a comma at the end of a line.

Extra Option

To hear the singing you will need the TI Speech Synthesizer and the Terminal Emulator II command module. When you turn on the computer with the module plugged in, press any key to start, then press 1 for TI BASIC and program as usual. To run the program without speech, you can select option 2 when the program starts. In this case, you don't need the module or the speech synthesizer.

If you choose no speech, the variable SP will equal 2. All the IF SP=2 THEN ... statements skip over commands that require the Terminal Emulator II module. The CALL SOUND statements play the tune. I used only one note; you may add accompaniment if you'd like. After the tone is played, the letter is sung. The CALL HCHAR or CALL VCHAR statements then place the letter on the screen.

Lines 1880–1910 wait after the song is over until the user presses ENTER, then the song is repeated.

If you prefer to save typing time and effort, you can obtain a copy of this program by sending a $3 copying fee, a blank cassette or diskette, and a stamped, self-addressed mailer to:

C. Regena
P.O. Box 1502
Cedar City, UT 84720

Please specify the name of the program and that you need the TI version.

Alphabet Song

```
100 REM ALPHABET SONG
110 CALL CLEAR
120 FOR C=97 TO 122
130 READ C$
140 CALL CHAR(C,C$)
150 NEXT C
160 DATA 3D438181818181433D,BCC281818181C2BC,3C4280808080423C,0000010101010101,3C4281FF8080423C
170 DATA 0609080808083E,0101010141221C,0000080808080808,00000008,08080808887,8890A0C0A0908884
180 DATA 0808080808080808,7884020202020202,BCC2818181818181,3C428181818181423C,808080808,010101010101
190 DATA BCC2818080808,3C42403C0202423C,00000808087F08,818181818181433D,4141222214140808,040488885050202
200 DATA 8244281028448282,10102020404,7F0204081020407F
210 T=600
220 PRINT TAB(8);"ALPHABET SONG"
230 PRINT : : : :"CHOOSE:"
240 PRINT : :"1  WITH SPEECH"
250 PRINT :"TERMINAL EMULATOR 2 REQUIRED"
260 PRINT : :"2  NO SPEECH": :
270 CALL KEY(0,K,S)
280 IF (K<49)+(K>50)THEN 270
290 SP=K-48
300 CALL CLEAR
310 IF SP=2 THEN 340
320 OPEN #1:"SPEECH",OUTPUT
330 PRINT #1:"//43 128"
340 CALL SOUND(T,262,2)
350 IF SP=2 THEN 370
360 PRINT #1:"A"
370 CALL HCHAR(3,3,97)
380 CALL SOUND(T,262,4)
390 IF SP=2 THEN 410
400 PRINT #1:"B"
410 CALL HCHAR(2,7,104)
420 CALL HCHAR(3,7,98)
430 CALL SOUND(T,392,2)
440 IF SP=2 THEN 470
450 PRINT #1:"//30 96"
460 PRINT #1:"C"
470 CALL HCHAR(3,11,99)
480 CALL SOUND(T,392,4)
490 IF SP=2 THEN 510
500 PRINT #1:"D"
510 CALL HCHAR(2,15,100)
520 CALL HCHAR(3,15,97)
530 CALL SOUND(T,440,2)
540 IF SP=2 THEN 570
550 PRINT #1:"//27 86"
560 PRINT #1:"E"
570 CALL HCHAR(3,19,101)
580 CALL SOUND(T,440,4)
590 IF SP=2 THEN 610
600 PRINT #1:"F"
610 CALL HCHAR(2,23,102)
620 CALL HCHAR(3,23,108)
630 CALL SOUND(T*2,392,2)
640 IF SP=2 THEN 670
650 PRINT #1:"//30 96"
660 PRINT #1:"G"
670 CALL HCHAR(3,27,97)
680 CALL HCHAR(4,27,103)
690 CALL SOUND(T,349,2)
700 IF SP=2 THEN 730
710 PRINT #1:"//34 109"
720 PRINT #1:"H"
730 CALL HCHAR(7,6,104)
740 CALL HCHAR(8,6,110)
750 CALL SOUND(T,349,4)
760 IF SP=2 THEN 780
770 PRINT #1:"I"
780 CALL HCHAR(7,10,105)
790 CALL HCHAR(8,10,108)
800 CALL SOUND(T,330,2)
810 IF SP=2 THEN 840
820 PRINT #1:"//36 115"
830 PRINT #1:"J"
840 CALL HCHAR(7,14,105)
850 CALL HCHAR(8,14,108)
860 CALL HCHAR(9,14,106)
870 CALL SOUND(T,330,4)
880 IF SP=2 THEN 910
890 PRINT #1:"K"
900 PRINT #1:"//39 125"
910 CALL HCHAR(7,18,104)
920 CALL HCHAR(8,18,107)
930 CALL SOUND(T/2,294,1)
940 IF SP=2 THEN 960
950 PRINT #1:"L"
960 CALL VCHAR(12,8,108,2)
970 CALL SOUND(T/2,294,3)
980 IF SP=2 THEN 1000
990 PRINT #1:"M"
1000 CALL HCHAR(13,12,110)
1010 CALL HCHAR(13,13,109)
1020 CALL SOUND(T/2,294,2)
1030 IF SP=2 THEN 1050
1040 PRINT #1:"N"
1050 CALL HCHAR(13,17,110)
1060 CALL SOUND(T/2,294,4)
1070 IF SP=2 THEN 1090
1080 PRINT #1:"O"
1090 CALL HCHAR(13,21,111)
1100 CALL SOUND(T*2,262,2)
1110 IF SP=2 THEN 1140
1120 PRINT #1:"//43 128"
1130 PRINT #1:"P"
1140 CALL HCHAR(13,25,98)
1150 CALL HCHAR(14,25,112)
1160 CALL SOUND(T,392,2)
1170 IF SP=2 THEN 1200
1180 PRINT #1:"//30 96"
1190 PRINT #1:"Q"
1200 CALL HCHAR(18,4,97)
1210 CALL HCHAR(19,4,113)
1220 CALL SOUND(T,392,4)
1230 IF SP=2 THEN 1250
1240 PRINT #1:"R"
1250 CALL HCHAR(18,8,114)
```

```
1260 CALL SOUND(T*2,349,2)
1270 IF SP=2 THEN 1300
1280 PRINT #1:"//34 109"
1290 PRINT #1:"S"
1300 CALL HCHAR(18,12,115)
1310 CALL SOUND(T,330,2)
1320 IF SP=2 THEN 1350
1330 PRINT #1:"//36 115"
1340 PRINT #1:"T"
1350 CALL HCHAR(17,16,116)
1360 CALL HCHAR(18,16,108)
1370 CALL SOUND(T,330,4)
1380 IF SP=2 THEN 1400
1390 PRINT #1:"U"
1400 CALL HCHAR(18,20,117)
1410 CALL SOUND(T*2,294,2)
1420 IF SP=2 THEN 1460
1430 PRINT #1:"//39 125"
1440 PRINT #1:"V"
1450 PRINT #1:"//30 96"
1460 CALL HCHAR(18,24,118)
1470 CALL SOUND(T,392,2)
1480 IF SP=2 THEN 1500
1490 PRINT #1:"DUB"
1500 CALL HCHAR(23,10,118)
1510 CALL HCHAR(23,11,119)
1520 CALL SOUND(T,392,4)
1530 IF SP=2 THEN 1550
1540 PRINT #1:"BL"
1550 CALL SOUND(T*2,349,2)
1560 IF SP=2 THEN 1590
1570 PRINT #1:"//34 109"
1580 PRINT #1:"U"
1590 CALL SOUND(T,330,2)
1600 IF SP=2 THEN 1630
1610 PRINT #1:"//36 115"
1620 PRINT #1:"X"
1630 CALL HCHAR(23,15,120)
1640 CALL SOUND(T,330,4)
1650 IF SP=2 THEN 1670
1660 PRINT #1:"Y"
1670 CALL HCHAR(23,19,118)
1680 CALL HCHAR(24,19,121)
1690 CALL SOUND(T*2,294,2)
1700 IF SP=2 THEN 1730
1710 PRINT #1:"//39 125"
1720 PRINT #1:"Z"
1730 CALL HCHAR(23,23,122)
1740 CALL SOUND(T,262,2)
1750 CALL SOUND(T,262,4)
1760 CALL SOUND(T,392,2)
1770 CALL SOUND(T,392,4)
1780 CALL SOUND(T,440,2)
1790 CALL SOUND(T,440,4)
1800 CALL SOUND(T*2,392,2)
1810 CALL SOUND(T,349,2)
1820 CALL SOUND(T,349,4)
1830 CALL SOUND(T,330,2)
1840 CALL SOUND(T,330,4)
1850 CALL SOUND(T,294,2)
1860 CALL SOUND(T,294,4)
1870 CALL SOUND(T*4,262,2)
1880 CALL KEY(0,K,S)
1890 IF K<>13 THEN 1880
1900 CALL CLEAR
1910 GOTO 330
1920 END
```

©

64 Searcher

John Krause, Assistant Technical Editor
and Michael Jacobi

"64 Searcher" is a time-saving utility that searches through your BASIC program and locates any character or string of characters that you choose. (This is a 64 version of VIC searcher that appeared in COMPUTE!, February 1983.)

When you're working on a long BASIC program, it pays to plan ahead. But it seems that no matter how hard you try, you can't keep track of everything in your program. Can I use H to store the high score, or is that variable already being used for something else? Where is this subroutine called from? You probably end up searching for a certain number or word hidden among scores of program lines.

"64 Searcher" allows you to spend less time searching and more time programming. You simply give it the string of characters to search for and it tells you the numbers of all lines in which the string appears. It can search 100 lines faster than it takes you to search one. It's fast because it's written in machine language. But you don't have to know machine language to use it.

Just LOAD it and RUN it, then LOAD your BASIC program. 64 Searcher doesn't use any BASIC memory, so you can work on your program normally. You can use 64 Searcher at any time by typing 0 followed by the string you want to find enclosed within either slashes or quotes, and hitting RETURN. This stores the string in your program as line 0. If your program already has a line 0, you will have to change that line number because the string must be the first line in the program.

Then type SYS49152 and hit RETURN. Instantly, you should see numbers appear on the screen. These are the line numbers that contain the string you specified. If no match is found, no numbers will be printed. If the string occurs more than once in a line, the line number is printed only once.

Because BASIC commands are stored differently from other characters in a program, there are two ways of specifying the search string. If the string is enclosed within slashes, BASIC commands are recognized as such. If the string is within quotes, it will be treated as a literal string of characters.

For example, to find the BASIC statement AND, line 0 should be:

0 /AND/

After entering SYS49152, 64 Searcher will find the AND in this line:

10 IF X AND Y THEN 50

but not in this line:

20 PRINT "X AND Y"

To find the AND in line 20 above, use quotes instead of slashes.

Remember to delete line 0 before saving or running your program.

64 Searcher

Refer to the "Automatic Proofreader" article before typing this program in.

```
10 FORI=49152TO49255:READJ:K=K+J:POKEI,J:
   NEXT                              :rem 66
20 IFK<>16302THENPRINT"ERROR IN DATA STAT
   EMENTS":STOP                     :rem 117
30 PRINT"{CLR}SYS49152 TO SEARCH" :rem 36
100 DATA169,1,133,251,169,8,133,252,160,0
    ,177,251,56,229,251,56           :rem 80
110 DATA233,5,141,104,192,233,2,141,105,1
    92,160,0,177,251,170,200        :rem 142
120 DATA177,251,240,67,133,252,134,251,16
    0,0,177,251,56,229,251,170       :rem 17
130 DATA202,134,2,198,2,165,2,205,104,192
    ,48,222,133,253,173,105         :rem 110
140 DATA192,133,254,164,253,177,251,164,2
    54,217,5,8,208,229,198,253       :rem 45
150 DATA198,254,208,239,160,2,177,251,170
    ,200,177,251,32,205,189,169      :rem 88
160 DATA32,32,210,255,76,26,192,96
                                    :rem 190
```

©

Jim Butterfield, Associate Editor

Decimal Mode
Part 2

Decimal mode is quite useful in arithmetic programming such as game scoring and simple accounting. It has other uses, too—for example, in converting binary numbers to decimal for output. It also has certain bugs, pitfalls, and conventions.

Bugs And Pitfalls

Don't depend on the Zero and Negative (Z and N) flags immediately following a decimal addition (ADC) and subtraction (SBC). If you really need them, perform a data transfer (for example, TAX) to insure the flags are set correctly. The Carry flag is correct and has its usual meaning after the addition or subtraction.

Remember that decimal mode uses only the ADC and SBC instructions. The increment and decrement instructions (INX, INY, INC, DEX, DEY, DEC) behave in binary; and comparisons (CMP, CPX, CPY) are based as usual on binary values.

Programmers using machines with interrupt sequences must be careful of decimal mode. The interrupt can clear decimal mode with CLD (Clear Decimal); when the interrupt code finishes with RTI, the status register will be restored and decimal mode will be reinstated if it was in effect before. On Commodore machines, the interrupt sequences do not include a CLD instruction; in this case, the interrupt should be locked out using a SEI (Set Interrupt Disable) before going into decimal mode.

The VIC-20 and Commodore 64 have a useful feature: Registers may be preset before a SYS call. Addresses $030C, $030D, $030E, and $030F (decimal 780 to 783) contain values that will be transferred to registers A, X, Y, and the status register at the time of a SYS. When the machine language program returns to BASIC, these same addresses will contain the contents of the respective register. In other words, we could POKE 780,65 followed by a SYS; and the machine language program would start running with a value of $41 (decimal 65) in the A register.

What does this mean to decimal mode? Here's the possible danger: If the wrong value is contained in address 783, it will be transferred to the status register at the time of a SYS. An uncontrolled value might set decimal mode, or even worse, set the interrupt disable flag. To make things worse, these flags will not be restored when we return to BASIC. They will be neatly stored in 783, but BASIC will resume with the flags in an unworkable state. There goes BASIC.

It's probably wise to leave address 783 alone. If it worries you, POKE 783,0 before giving a SYS command.

Conventions

We can handle fractions in decimal arithmetic. It's best to do this by using an "assumed decimal point." In other words, we will work dollar values as an integer number of pennies, and kilometers as integer values of meters. It's easier to stick in the decimal point at output time.

Negative numbers are a little tricky. We can use a scheme similar to that in binary numbers: That is, the "high bit" of a number represents the sign. This, however, splits positive and negative unevenly: A two-byte number will range from a low of −2000 (value 8000) up to +7999. If you use this method, don't forget that the N flag isn't dependable after an addition or subtraction and that you'll need to take an extra step to test the flag.

A better technique is called "tens complement" and it's been used in many household devices such as counters on tape recorders. We understand that a reading of 9994 really means −6. If we want to use this technique, we might choose to try to split positive and negative more evenly, so that a two-byte number would range from −5000 to +4999. In this case, we must remember not to use the N bit, but instead compare the high byte to 50 hex. If it is higher, the number is negative.

If "tens complement" is used, remember to invert a negative number at the time of printing. I find that the easiest way to do this is to subtract it from 0000 so that 9993 becomes 0007.

Multiplication

To multiply two decimal numbers we are almost forced to resort to repeated addition. As we go from one decimal digit to the next, we must "shift" either the multiplier or the product: This is a binary shift-four-places. It's awkward and we can quickly see why binary is preferred.

There's an elegant way to multiply a decimal number by a binary value, or by a fixed amount. We can use what I call a "decimal shift."

A binary shift multiplies a number by two. We can do the same thing with a decimal number by adding it to itself. Thus, to multiply by two we add the number to itself (in decimal mode). To multiply by four we multiply by two, twice. To multiply by five, we multiply by four and add the original number.

A Multiplication Example

We'll have the computer (PET, VIC, or 64) output a table of multiples of the number 5. (Two would be too easy.)

```
                ; set value to one
033C A2  01             LDX   #$01
033E 8E  B0  03         STX   LOW
0341 CA                 DEX
0342 8E  B1  03         STX   MED
0345 8E  B2  03         STX   HIGH
0348 8E  B6  03         STX   COUNT
                ; copy the number
034B A0  02        LOOP LDY   #$02
034D B9  B0  03    CP   LDA   LOW,Y
0350 99  B3  03         STA   COPY,Y
0353 88                 DEY
0354 10  F7             BPL   CP
                ; multiply by four
0356 A2  02             LDX   #$02
0358 18            FP   CLC
0359 A0  FD             LDY   #$FD
035B 78                 SEI
035C F8                 SED
035D B9  B3  02    TP   LDA   HIGH-255,Y
0360 79  B3  02         ADC   HIGH-255,Y
0363 99  B3  02         STA   HIGH-255,Y
0366 C8                 INY
0367 D0  F4             BNE   TP
0369 CA                 DEX
036A D0  EC             BNE   FP
                ; add original value
036C A0  FD             LDY   #$FD
036E 18                 CLC
036F B9  B3  02    AP   LDA   HIGH-255,Y
0372 79  B6  02         ADC   COPY-253,Y
0375 99  B3  02         STA   HIGH-255,Y
0378 C8                 INY
0379 D0  F4             BNE   AP
037B D8                 CLD
037C 58                 CLI
                ; print the number
037D A0  02             LDY   #$02
037F B9  B0  03    LP   LDA   LOW,Y
0382 4A                 LSR   A
0383 4A                 LSR   A
0384 4A                 LSR   A
0385 4A                 LSR   A
0386 09  30             ORA   #$30
0388 20  D2  FF         JSR   $FFD2
038B B9  B0  03         LDA   LOW,Y
039E 29  0F             AND   #$0F
0390 09  30             ORA   #$30
0392 20  D2  FF         JSR   $FFD2
0395 88                 DEY
0396 10  E7             BPL   LP
                ; print RETURN and loop
0398 A9  0D             LDA   #$0D
039A 20  D2  FF         JSR   $FFD2
039D EE  B6  03         INC   COUNT
03A0 AE  B6  03         LDX   COUNT
03A3 E0  07             CPX   #$08
03A5 D0  A4             BNE   LOOP
03A7 60                 RTS
```

Note the peculiar addressing in lines 035D to 0363 and again in 036F to 0375. We need to have a positive-incrementing index (in this case Y), since we must start our addition at the low-order value, LOW, and work upwards. We cannot use the obvious method of starting at zero and testing to see when we have done all three values, because we want the carry flag to be preserved; CPY (Compare Y) would destroy the previous value of the carry and our addition wouldn't work right.

If you'd rather enter the program from BASIC, here's the same program in DATA statements. It will work on all Commodore machines.

Multiples Of 5

```
100 DATA 162,1,142,176,3,202,142,177,3
110 DATA 142,178,3,142,182,3,160,2
120 DATA 185,176,3,153,179,3,136,16,247
130 DATA 162,2,24,160,253,120,248,185,179
    ,2
140 DATA 121,179,2,153,179,2,200,208,244,
    202
150 DATA 208,236,160,253,24,185,179,2,121
    ,182,2
160 DATA 153,179,2,200,208,244,216,88,160
    ,2
170 DATA 185,176,3,74,74,74,74,9,48,32,21
    0,255
180 DATA 185,176,3,41,15,9,48,32,210,255,
    136,16
190 DATA 231,169,13,32,210,255,238,182,3,
    174,182,3
200 DATA 224,8,208,164,96
300 FOR J=828 TO 935
310 READ X:T=T+X
320 POKE J,X
330 NEXT J
340 IF T<>13479 THEN STOP
350 SYS 828
```

©

Larry Isaacs

This month we will continue our look at printing characters to a bitmapped display. Last month we looked at a method which transferred a character dot pattern to the bitmapped display. This month we will look at a second method, which *draws* the characters.

Printing Bit By Bit

With the appropriate set of line segments, virtually any character shape can be drawn. The characters do not necessarily have to look like the standard ASCII character set. In addition, you are not restricted to a fixed character cell. Each character can be as complex and as large as you like. For this flexibility, you do lose a few advantages offered by the use of character dot patterns. (It becomes a little more difficult to print in reverse video and will likely take a little longer to print the character when characters are drawn rather than transferred.)

With the drawing method, we will need to make use of a line-drawing routine. For convenience, I will be using the machine language line-drawing routines presented in the May issue of COMPUTE!. However, for use in the example BASIC program which follows, almost any line-drawing routine will suffice. (The one found in COMPUTE!'s earlier "SuperBASIC 64" program could be used if you desire. Some minor modifications to the BASIC program will be required, though.)

To draw a given character in the bitmapped display, we will need some data to define how the character should be drawn. Unlike the transfer method, where the format for such data is already fixed, here we have total freedom to define our own format. The format must specify what line segments should be drawn to form the characters. This means that the data must define the starting and ending coordinates of each line segment. Another thing to note is that the data will need to define these coordinates relative to the previous coordinates. By specifying the next point based on the previous point, the character can be drawn anywhere in the bitmapped display.

To simplify the following discussion, I will use the term "vector" to refer to the line segments which make up a character. Also, I will use the term "vector string" for the data which defines how to draw a character.

One way we could define the format of the data in the vector string is to specify each vector with two pairs of relative coordinates. A single byte could be used for each relative coordinate, which could represent a value from 127 to −128. Thus, four bytes would be required for each vector in the vector string.

Moving Points

As I mentioned in the May column, I prefer to have the *draw* function continue from each previous endpoint. This eliminates the need to specify a new beginning point every time. The catch is that there must be some way of moving the last endpoint without drawing. Assuming we define a way of moving the endpoint for our vector string, then it will be possible to specify a vector using one pair of relative coordinates, rather than two. For this to be an advantage, a fair percentage of the vectors would need to draw from the end of the previous vector. When creating characters from vectors, I believe this will generally be true.

If the characters are not going to be that large, there is another phenomenon: Most of the vectors will be fairly short. Assuming they are typically short enough, we could save more bytes by using one byte to specify a vector. The byte could be split into an upper and lower four bits, with each half able to represent a relative coordinate of 7 to −8. This may not seem like very much, but if the vector isn't too long to be represented by two of these bytes, we haven't lost anything.

Vector Bytes

This isn't the only way to use a single byte to specify a vector. The byte could be split into two parts so that one part specifies a direction and the other a distance. The direction in this case would most likely be a multiple of 45 degrees. This actually works quite well for drawing characters. However, I will go with the format of putting relative coordinates into the byte. I will refer to such bytes as "vector bytes" in the discussion which follows.

Given that the vector has a limited range, we will need to define some way of invoking exceptions to handle the times when the range is exceeded. Also, we still need to define a way of moving instead of drawing, which we will also treat as an exception. One way to do this is to use one of the coordinate values to signal the exception. The other relative coordinate could be used to indicate which exception. Since this uses both halves of the vector byte, the exceptions will require additional bytes.

Now we are ready to get down to specifics. Let's try putting the relative coordinate for X in the upper half of the vector byte. Naturally that means putting the relative coordinate for Y in the lower half. As for a value to signal exceptions, it is most logical to choose a value at an extreme. Since our range is from 7 to −8, −8 would be the best choice. It also would be best to have this value in the upper half of the vector byte. This would cause the exception bytes to fall in the range of 128 to 143. Bytes outside this range will be regular "drawing" vector bytes.

There are four exceptions we will need to deal with initially. These exceptions are for signaling a move, an extended draw, an extended move, and the end of the vector byte string. With the upper half signaling an exception, this leaves the lower half to flag the exceptions.

Also, the numbering of the exceptions will be a little easier if we treat the four bits as unsigned rather than signed. This lets us have values from 0 to 15, instead of 7 to −8. For the exceptions, let's try values of 0, 1, and 2 to select move, extended draw, and extended move, respectively. To mark the end of a vector byte string, let's try 15, to choose an extreme again. The following table summarizes these choices:

Data Formats For Vector Byte String

Byte No.	Bit 7 6 5 4	3 2 1 0	
Vector Byte			
1	DX	DY	DX = 7 to −7 DY = 7 to −8
Move Exception			
1	−8	0	= 128
2	DX	DY	DX,DY = 7 to −8
Extended Draw Exception			
1	−8	1	= 129
2	DX		= 127 to −128
3	DY		
Extended Move Exception			
1	−8	2	= 130
2	DX		= 127 to −128
3	DY		= 127 to −128

Drawing Strings Of Characters

Now we are ready to implement the above in a BASIC program. The result is shown in Program 1. The task of drawing the character has been split into a number of routines. One routine fetches the next vector byte, and another unpacks the relative distances. There is also a separate routine to handle vector byte draws, vector byte moves, extended draws, and extended moves. Finally, there is a routine which draws a character, and which in turn uses that routine to draw a string of characters.

The vector byte data included in Program 1 defines vector byte strings for characters 65 through 90, or A through Z. The vector byte strings will draw the ASCII character corresponding to the character code. The space character is also defined. The vector byte strings are stored in the string array VB$ and are accessed by using the character code as the subscript into the array. Prior to running the program, it will be necessary to load the line-drawing routines presented in the May column.

Character Rotation

I have included a routine which lets you specify a character rotation in increments of 90 degrees. For drawn characters, the rotation involves simply negating or swapping the relative coordinates specified in a vector byte. Rotating the transferred character pattern is not too difficult provided the cell is square, as it is in our case. Rotating the character to other angles typically won't produce desirable-looking characters, and may be too complex to implement.

The following is a table showing the routines that are available, and what their start address is. define the base location of the second jump table. Here is a list of the routines:
Since this will be the second jump table (to complement the line-drawing jump table), I use J2 to

Loc.	Description
J2+0	SET PUT CHAR. DATA LOCATION
J2+3	PUT CHARS. IN BITMAP (TRANSFER METHOD)
J2+6	SET DRAW CHAR. DATA LOCATION
J2+9	DRAW CHARS. IN BITMAP
J2+12	SET ROTATION
J2+15	NOT USED YET
J2+18	NOT USED YET
J2+21	NOT USED YET

The jump vector location of these routines is shown as the variable J2 plus an offset. To obtain the actual address, J2 should be set to the base of the jump table which is 50176 or C400 hex. The following list gives the syntax for using each of the defined routines in the jump table:

```
SYS   J2,LOC
SYS   J2+3,CHAR or STRING
```

```
SYS   J2+6,LOC
SYS   J2+9,CHAR or STRING
SYS   J2+12,ROT
        ROT: 0=NO ROT.,  1=90 DEGREES
      2=180 DEG., 3=270 DEG.
```

Both the put character (J2+3) and draw character (J2+9) will accept either a single character or a string of characters as an argument. If the argument supplied is a numeric value, it will be interpreted as the ASCII value of a single character. If the argument is a string, the entire string will be printed.

The location required by the put character routine should be the base address of the character dot patterns to use. The location required by the draw character routine is the base address of a 256-byte table containing pointers to 128 vector byte strings. The pointers to the vector byte strings are each two bytes, stored as low byte followed by high byte. Use of a table is necessary because the length of each string may vary, making it impossible to calculate the locations of the vector byte strings directly.

Safe Entry

Program 2 will POKE the machine code for the character routines into the proper locations. Like the program which POKEs the line-drawing routines, that last number in each data line is the sum of the previous eight bytes on the line. Provided you don't make two errors which cancel each other, the program will report any lines that have mistakes in them. If there are no detected errors, a SUCCESSFUL LOAD is reported.

Program 3 provides a simple illustration of the use of the character routines. For vector byte data, add the DATA statements shown in Program 1, which will define ASCII characters A through Z, and space. The vector byte data will be placed at the top of BASIC's free RAM, after 1024 bytes of space is reserved from BASIC. You will be able to see the increase in speed over the BASIC routines.

Program 1: Draw Characters In A Bitmap

Refer to the "Automatic Proofreader" article before typing this program in.

```
10 REM DRAW CHARACTERS IN BIT-MAP:rem 212
20 JT=49152:REM DRAWING JUMP TABLE:rem 16
30 DIM VB$(256):REM DIM. STRING ARRAY
                                  :rem 99
40 X=0:Y=0:REM X,Y LOCATION       :rem 125
50 DX=0:DY=0:REM DELTA-X,DELTA-Y :rem 219
60 VB=0:REM VECTOR BYTE           :rem 144
70 VB$="":REM VECTOR BYTE STRING :rem 160
80 VP=0:VL=0:REM PTR INTO VB$, VB$ LEN
                                  :rem 149
90 GOTO 1000                      :rem 99
100 REM GET NEXT VECTOR BYTE      :rem 155
110 VP=VP+1:REM BUMP POINTER      :rem 234
120 IF VP>VL THEN VB=0:RETURN     :rem 246
130 VB=ASC(MID$(VB$,VP,1)):REM GET BYTE
                                  :rem 253
140 RETURN                        :rem 117
200 REM UNPACK VECTOR BYTE        :rem 63
210 DY=VBAND15:IF DYAND8 THEN DY=DYOR-8
                                  :rem 91
220 DY=VBAND15:IF DYAND8 THEN DY=DYOR-8
                                  :rem 92
230 DX=INT(VB/16):IF DXAND8 THEN DX=DXOR-
    8                             :rem 242
240 RETURN                        :rem 118
300 REM EXECUTE VECTOR BYTE DRAW :rem 191
310 X=X+DX:Y=Y+DY                 :rem 57
320 SYS JT+18,X,Y:REM DRAW THE BYTE
                                  :rem 48
330 RETURN                        :rem 118
400 REM EXECUTE VECTOR BYTE MOVE :rem 201
410 GOSUB 100:GOSUB 200:REM GET NEXT
                                  :rem 47
420 X=X+DX:Y=Y+DY                 :rem 59
430 SYS JT+12,X,Y:REM DO THE MOVE:rem 148
440 RETURN                        :rem 120
500 REM GET EXTENDED DX AND DY    :rem 182
510 GOSUB 100:DX=VB:REM EXTENDED DX
                                  :rem 93
520 IF DX AND 128 THEN DX=DX OR -128
                                  :rem 61
530 GOSUB 100:DY=VB:REM EXTENDED DY
                                  :rem 97
540 IF DY AND 128 THEN DY=DY OR -128
                                  :rem 66
550 RETURN                        :rem 122
600 REM EXECUTE EXTENDED DRAW     :rem 12
610 GOSUB 500:REM GET DX,DY       :rem 15
620 X=X+DX:Y=Y+DY                 :rem 61
630 SYS JT+18,X,Y:REM DO THE DRAW:rem 147
650 RETURN                        :rem 123
700 REM EXECUTE EXTENDED MOVE     :rem 22
710 GOSUB 500:REM GET DX,DY       :rem 16
720 X=X+DX:Y=Y+DY                 :rem 62
730 SYS JT+12,X,Y:REM DO THE DRAW:rem 142
750 RETURN                        :rem 124
800 REM DRAW STRING OF VECTOR BYTES
                                  :rem 112
810 VP=0:VL=LEN(VB$):IF VL=0 THEN RETURN
                                  :rem 152
820 IF VP>=VL THEN RETURN         :rem 251
830 GOSUB 100:REM GET NEXT VB     :rem 129
840 GOSUB 200:REM UNPACK          :rem 142
850 IF DX<>-8 THEN GOSUB 300:GOTO 820
                                  :rem 246
860 ON DY+1 GOSUB 400,600,700     :rem 205
870 GOTO 820                      :rem 114
900 REM PRINT P$                  :rem 126
910 FOR PP=1 TO LEN(P$)           :rem 214
920 VB$=VB$(ASC(MID$(P$,PP,1)))  :rem 181
930 GOSUB 800:REM DRAW THE CHAR.  :rem 45
940 NEXT:RETURN                   :rem 246
1000 REM MAIN ROUTINE             :rem 240
1010 GOSUB 10000                  :rem 51
1020 SYS JT:SYS JT+6,0:SYS JT+9,6,14
                                  :rem 34
1030 X=10:Y=100:SYS JT+12,X,Y    :rem 137
1040 FOR CH=64 TO 90              :rem 234
1050 VB$=VB$(CH):GOSUB 800        :rem 169
1060 NEXT                         :rem 6
1070 X=10:Y=80:SYS JT+12,X,Y     :rem 100
1080 P$="THIS IS AN EXAMPLE OF"  :rem 194
1090 GOSUB 900                    :rem 227
1100 X=10:Y=60:SYS JT+12,X,Y     :rem 92
1110 P$="PRINTING WITH VECTOR BYTES"
                                  :rem 185
1120 GOSUB 900                    :rem 221
9000 GET Z$:IF Z$="" THEN 9000   :rem 231
```

```
9010 SYS JT+3                              :rem 197
9020 END                                   :rem 162
10000 REM LOAD VB$()                         :rem 2
10010 C=0:READ CH:IF CH<0 THEN RETURN
                                           :rem 82
10020 READ VB:IF C>0 THEN C=C-1:GOTO 1008
      0                                    :rem 221
10030 IF ABS(VB)>7 THEN 10060             :rem 221
10040 READDY:VB=(VB*16+(DYAND15)):rem 137
10050 GOTO 10080                           :rem 40
10060 IF VB=143 THEN 10010                 :rem 20
10070 IF VB<>128 THEN C=2                  :rem 21
10080 VB$(CH)=VB$(CH)+CHR$(VB AND 255)
                                           :rem 234
10090 GOTO 10020                           :rem 38
11100 REM @                                :rem 23
11110 DATA 64,128,3,3,0,1,-1,0,0,-1,3,0
                                           :rem 61
11120 DATA 0,2,-1,1,-3,0,-1,-1,0,-4
                                           :rem 105
11160 DATA 128,-5,2,5,0,128,3,-2,143
                                           :rem 204
11130 DATA 1,-1,3,0,1,1,128,3,-1,143
                                           :rem 180
11140 REM A                                :rem 28
11150 DATA 65,0,4,2,2,1,0,2,-2,0,-4
                                           :rem 128
11170 REM B                                :rem 32
11180 DATA 66,0,6,4,0,1,-1,0,-1,-1,-1
                                           :rem 220
11190 DATA -4,0,128,0,-3,4,0,1,1,0,1
                                           :rem 177
11200 DATA -1,1,128,4,-3,143               :rem 64
11210 REM C                                :rem 28
11220 DATA 67,128,5,5,-1,1,-3,0,-1,-1
                                           :rem 233
11230 DATA 0,-4,1,-1,3,0,1,1               :rem 42
11240 DATA 128,3,-1,143                    :rem 90
11250 REM D                                :rem 33
11260 DATA 68,0,6,3,0,2,-2,0,-2,-2,-2
                                           :rem 225
11270 DATA -3,0,130,8,0,143                :rem 21
11280 REM E                                :rem 37
11290 DATA 69,0,6,5,0,128,-1,-3,-4,0
                                           :rem 199
11300 DATA 128,0,-3,5,0,128,3,0,143
                                           :rem 149
11310 REM F                                :rem 32
11320 DATA 70,0,6,5,0,128,-5,-3,4,0
                                           :rem 144
11330 DATA 128,4,-3,143                    :rem 93
11340 REM G                                :rem 36
11350 DATA 71,128,5,5,-1,1,-3,0,-1,-1
                                           :rem 232
11360 DATA 0,-4,1,-1,3,0,1,1,0,2,-2,0
                                           :rem 207
11370 DATA 128,5,-3,143                    :rem 98
11380 REM H                                :rem 41
11390 DATA 72,0,6,128,0,-3,5,0,128,0,3
                                           :rem 45
11400 DATA 0,-6,128,3,0,143                :rem 21
11410 REM I                                :rem 36
11420 DATA 73,128,0,6,2,0,128,-1,0,0,-6
                                           :rem 83
11430 DATA 128,-1,0,2,0,128,3,0,143
                                           :rem 148
11440 REM J                                :rem 40
11450 DATA 74,128,0,1,1,-1,2,0,1,1,0,5
                                           :rem 24
11460 DATA 128,-1,0,2,0,128,3,-6,143
                                           :rem 202
11470 REM K                                :rem 44
11480 DATA 75,0,6,128,0,-4,4,4,128,-3,-3
                                           :rem 145
11490 DATA 1,0,3,-3,128,3,0,143            :rem 215
11500 REM L                                :rem 39
11510 DATA 76,0,6,128,0,-6,5,0,128,3,0,14
      3                                    :rem 242
11520 REM M                                :rem 42
11530 DATA 77,0,6,3,-3,3,3,0,-6,128,3,0,1
      43                                   :rem 26
11540 REM N                                :rem 45
11550 DATA 78,0,6,5,-5,128,0,5,0,-6
                                           :rem 161
11560 DATA 128,3,0,143                     :rem 49
11570 REM O                                :rem 49
11580 DATA 79,128,1,0,-1,1,0,4,1,1,3,0,1,
      -1                                   :rem 8
11590 DATA 0,-4,-1,-1,-3,0,128,7,0,143
                                           :rem 29
11600 REM P                                :rem 44
11610 DATA 80,0,6,4,0,1,-1,0,-1,-1,-1
                                           :rem 214
11620 DATA -4,0,130,8,-3,143               :rem 69
11630 REM Q                                :rem 48
11640 DATA 81,128,1,0,-1,1,0,4,1,1,3,0,1,
      -1                                   :rem 254
11650 DATA 0,-4,-1,-1,-3,0,128,2,1:rem 82
11660 DATA 2,-2,128,3,1,143                :rem 28
11670 REM R                                :rem 53
11680 DATA 82,0,6,4,0,1,-1,0,-1,-1,-1,-4,
      0                                    :rem 200
11690 DATA 128,2,0,3,-3,128,3,0,143
                                           :rem 161
11700 REM S                                :rem 48
11710 DATA 83,128,0,1,1,-1,3,0,1,1,0,1,-1
      ,1                                   :rem 251
11720 DATA -3,0,-1,1,0,1,1,1,3,0,1,-1
                                           :rem 205
11730 DATA 128,3,-5,143                    :rem 98
11740 REM T                                :rem 53
11750 DATA 84,128,2,0,0,6,128,-2,0,4,0
                                           :rem 45
11760 DATA 128,3,-6,143                    :rem 102
11770 REM U                                :rem 57
11780 DATA 85,128,0,6,0,-5,1,-1,3,0,1,1
                                           :rem 83
11790 DATA 0,5,128,3,-6,143                :rem 38
11800 REM V                                :rem 52
11810 DATA 86,128,0,6,0,-4,2,-2,2,2,0,4
                                           :rem 82
11820 DATA 128,3,-6,143                    :rem 99
11830 REM W                                :rem 56
11840 DATA 87,128,0,6,0,-6,3,3,3,-3,0,6
                                           :rem 94
11850 DATA 128,3,-6,143                    :rem 102
11860 REM X                                :rem 60
11870 DATA 88,0,1,4,4,0,1,128,-4,0,0,-1
                                           :rem 83
11880 DATA 4,-4,0,-1,128,3,0,143           :rem 9
11890 REM Y                                :rem 64
11900 DATA 89,128,0,6,2,-2,2,2,128,-2,-2
                                           :rem 141
11910 DATA 0,-4,128,5,0,143                :rem 27
11920 REM Z                                :rem 59
11930 DATA 90,128,0,6,4,0,0,-1,-4,-4,0,-1
                                           :rem 168
11940 DATA 4,0,128,4,0,143                 :rem 240
11950 REM SPACE                            :rem 80
11960 DATA 32,130,8,0,143                  :rem 196
11970 DATA -1                              :rem 122
```

Programs 2 and 3 will appear in this column next month. ©

Are Computers A Home Appliance?

Fred D'Ignazio, Associate Editor

Necessary, Easy, And Inexpensive

In recent columns I have written about a growing consumer awareness that things are not right with the microcomputer industry. Some misleading advertisements have made people buy computers as a home appliance. Unfortunately, the computers have not met some people's expectations, and then ended up gathering dust in the closet.

To be a legitimate home appliance, a product should have three characteristics:

It should be inexpensive.

It should meet a real need.

It should be easy to use.

Let's look closely at each characteristic, and see how computers measure up.

A home appliance should be inexpensive. A low-end computer often appears to be inexpensive, but it turns out to be costly after a person adds the necessary "extras," including a disk drive, a printer, and some basic software.

A home appliance should meet a real need. For example, people use telephones to communicate; TVs for entertainment and news; ovens to cook food; and refrigerators to keep food fresh. But what do people need computers for?

A home appliance should be easy to use. For example, you can pick up a phone, dial seven numbers, and reach another person within seconds. You can push a button on a TV, and the world enters your living room. You can pull down a lever on the toaster oven and get a hot biscuit.

When you turn on the computer, it says, "READY." But it is not really ready. First you must load in additional software, turn on additional appliances (disk drives, a printer, a modem, etc.), answer questions, and type in additional information. All these cumbersome, time-consuming steps make the computer ready, but they do not make it easy to use.

WASH! Magazine

How do people learn how to use computers?

They might join a user group, ask a kid, or read a computer magazine.

A magazine like COMPUTE! can be a lifesaver for the consumer who has just bought an inexpensive computer. The magazine offers easy-to-read tutorials, practical tips, and lots of excellent, affordable software.

Kids can also be helpful. So can user groups. But all this is beside the point. The real question is: Should a home appliance be this difficult to use?

To put this question in perspective, ask yourself how many people would own a washing machine if, to operate it, they had to buy a monthly magazine called *WASH!* , and they had to get help from a washing-machine whiz kid and attend weekly meetings of the Whirlpool User Group?

And how fair is it to our children to assume that they will know how to use a machine that has us puzzled and bewildered?

It is easy for kids to get *intimate* with computers, because they share few of our fears, anxieties, and prejudices about these machines. But it is not nearly as easy for them to get computer *literate*—to be competent computer users and programmers. Nevertheless, we adults now have the misconception that all children take to computers as naturally as ducks to water. But what if our children *don't* take to computers? Does that make them less intelligent or less able than their friends? And where does that leave us?

A Growing Backlash

When millions of people buy a computer, take it home, then discover that it is not going to be inexpensive, that it meets no immediate need, and that it is not always easy to use, how do they feel? Whom do they blame?

Until recently, most people blamed themselves, their families, and their kids. But this is beginning to change. Too many people have been disappointed by computers, and they are talking to their neighbors. The secret is finally out. The fault is not with the consumer. It is with computers themselves—and the companies that make them.

New Consumer Savvy

The computer price wars of 1982–1983 had a disastrous effect on the computer industry and drove many companies out of the market, including Texas Instruments; Mattel, and Timex. In addition, many naive customers were lured by incredibly low prices into buying low-end computers. Unfortunately, the customers had no idea what to do with the computers once they got them home.

However, in spite of these setbacks, the ultimate effect of the price wars may be positive. Between 1982 and 1984, large numbers of people bought "throwaway" computers, became disgruntled consumers, and described their experiences to their neighbors. The result is that, today, people are a lot more knowledgeable about computers than they were just a year ago.

In fact, people's bad experience with computers and their "sour grapes" reaction have created a mild consumer backlash against computers. The average consumer, in mid-1984, is much more skeptical about computers than he was in 1982 or 1983. He realizes that a good price is not the only thing to look for when choosing a computer for the home. He understands that computers, to be useful, need good software, memory, printers, and disk drives. He realizes that even with all this equipment a computer is *not* a home appliance. On its own it won't guarantee him or his family anything.

The average consumer is returning to the healthier show-me attitude that prevailed before the era of *high-tech chic* that reigned from 1982 and 1984. "Show me real needs that computers meet," the consumer is saying. "Show me a computer with no hidden costs that is useful and simple to operate." ©

64 Error Suppression

Tom Nuss

There are times when you don't want error messages (and the resulting interruption) in a program. Here's how to avoid some kinds of system freezes.

While constructing a general graphing program that would handle varied equations, I realized that it would crash when it tried to divide by zero or take the square root of a negative number. Since a graphing program depends on drawing a fairly smooth curve, these two possibilities would definitely occur from time to time in a general program loop that plots. I also found I had to learn machine language to get things to happen before my hair turned gray.

After delving into the BASIC interpreter with my trusty Supermon-64, I discovered that if the accumulator contained a zero after a division, it would branch to the error routine at $A437, which would jump indirectly (via a vector address in $300 and $301) to $E38B and proceed to print messages and stop the BASIC program.

I soon confirmed that all error messages (at least the ones I tried) went through $A437, then jumped indirectly by the bytes loaded in $300 and $301 to $E38B. All I had to do was change the contents of $300 and $301 to an address pointing to where I would have my own routine that would skip over the error to allow the program to continue.

Back To BASIC

Simple enough, but how to get back in the BASIC program at the right spot once I left? No, not back into the interpreter again; anything but that!

I noticed that just down the page of the memory map at $A906 was the routine "Scan for next statement." Now that I look back on it, I should have started there. Needless to say, that entry point was the last of the major pieces to the puzzle of skipping the BASIC error handler.

To get down to the mechanics, I have made up a demonstration program to illustrate the method to bypass arithmetic errors. If you type in the program and RUN, you will get the results shown in the figure. AC SR XR YR are the Accumulator, Status, X, and Y registers respectively. Directly beneath them are several series of four numbers and either SYNTAX OK or a variation of F(#)= a number.

Ignoring the first line of eights and SYNTAX OK for the moment, you will notice that line 70 in the program defines the function SQR(4–C↑2)/C and that the second line of numbers in the figure is 255 49 14 1. When one has C=–3 in the above equation, mathematicians will shake their heads, and the computer should crash. Why didn't it? In fact it even gave an answer for SQR(–5)/–3 and blithely continued to calculate the rest of the F(C)s from –3 to 3. F(–2), F(–1), F(1), and F(2) give the correct answer, while F(–3), F(0), and F(3) don't.

Remember, our objective is to skip dilemmas like division by zero, so we must first find out if that is what the computer is trying to do. The way to do this is to look at line 10, which POKEs addresses 768 and 769 ($300 and $301) with 52 and 3 (or $34 and $03). These are the bytes indirectly used to tell the error routine where to go after it finds an error; normally these bytes contain the address $E38B, but line 10 changes this to address 820 decimal ($334). This is where our machine language routine is POKEd by line 20. Line 230 changes things back to normal after the program is finished. For those of you who wish to see the disassembled machine language routine, here it is:

```
STA $FB
PHP
PLA
STA $FC
STX $FD
STY $FE
PHA
PLP
LDA $FB
JMP $A906
```

The above routine is only used when there is an error. Locations 251–254 ($FB–$FE) are loaded with eights at the start (line 10) and loaded again each time through the loop that calculates F(C). However, if an error occurs in line 150, the error routine will load locations 251–254 with the contents of the registers at the time of the error and then continue with the next BASIC line. Line 160 prints out the contents of the registers and F(C). Thus the contents of 251–254 change only when an error occurs.

So, now the program doesn't crash; it just gives erroneous results, and that should also be avoided. Type in:

```
185 IF PEEK(252)>=48 THEN PRINT:GOTO 200
```

RUN the program again and there should be blanks where F(–3), F(0), and F(3) are involved. In other words, by PEEKing 252 and by comparing it to 48 we have skipped the errors; nothing has been printed, saved, recorded, or crashed. Only the proper numbers are still able to be used.

So much for mathematics. What if we define the function wrongly? LIST the program and change line 70 to: DEFFNF(C)=SQR(4 – C↑2/C and RUN. If all is not well you should see a line of four numbers, not eights, a SYNTAX ERROR (70) and line 70. Our error routine kicked in and in line 100 checked location 252 to see if it was less than 112 and told you about the error in syntax. This is really no advantage over the regular system, but if you are using the dynamic keyboard method to enter your DEFFNF(C) (see "Bootmaker for VIC, PET, and 64," COMPUTE!, May 1983), this routine would come in mighty handy.

Errors That Get Through

It should be pointed out that there is a potential problem with this routine. Change line 70 to DEFFNF(C)=SRR(4–C↑2)/C. Errors galore, but they weren't caught. Why not? I wish I knew. Please, not the BASIC interpreter again. All I can say is that in an instance like this you will, on most occasions, be able to tell there is an error and that the error is being caused by the DEFFN statement. Also, before including this specific Syntax Error routine in a program of your own, you should try putting a multiply sign (*) before the SQR in line 70 and then RUN. As you can see, the computer locks up. The only way to correct this situation is to turn the power off and reload the program. Weigh the advantages of including the Syntax Error routine described here against the very obvious disadvantage of system lockup.

To sum up:

1. POKE 768 and 769 ($300, $301) with the address of your machine language routine that will handle the BASIC errors. In the example presented here, 52 and 3 are POKEd, for location 820 ($334).

2. The error handling routine loads byte 252 ($FC) and provides the jump address to "Scan for next statement" at $A906 so you can reenter your program.

3. Check byte 252 (Status Register during an error) to see if it is greater than or equal to 48 for a mathematical error or 112 for a syntax error.

4. Take the appropriate action either to save an answer or to skip it.

5. POKE 768 and 769 with 139 and 227 respectively to restore the normal error vector address ($E38B). This is important since the computer won't be able to function in the immediate mode.

Error Suppression

```
10 POKE768,52:POKE769,3:FORC=0TO3:POKE251
   +C,8:NEXTC                          :rem 108
20 FORC=0TO16:READD:POKE820+C,D:NEXTC
                                        :rem 58
30 FORC=0TO17:PRINTCHR$(96);:NEXTC:PRINTC
   HR$(105)                             rem 51
40 PRINT" AC";TAB(5);"SR";TAB(10);"XR";TA
   B(15);"YR ";CHR$(125)               :rem 122
50 FORC1=0TO38:PRINTCHR$(96);:NEXTC1:PRIN
   T                                   :rem 178
60 PRINTCHR$(145);TAB(18);CHR$(177)
                                       :rem 215
70 DEFFNF(C)=SQR(4-C↑2)/C              :rem 206
80 SX=FNF(1)                           :rem 172
90 PRINTPEEK(251);TAB(4);PEEK(252);TAB(9)
   ;PEEK(253);TAB(14);PEEK(254);        :rem 27
100 IFPEEK(252)<112THENPRINT"{3 SPACES}SY
    NTAX OK":GOTO120                   :rem 134
110 PRINT"{3 SPACES}SYNTAX ERROR (70)":GO
    TO230                              :rem 148
120 FORC1=0TO38:PRINTCHR$(96);:NEXTC1:PRI
    NT                                 :rem 224
130 FORC=-3TO3                          :rem 49
140 C$=STR$(C)                         :rem 234
150 X=FNF(C)                           :rem 153
160 PRINTSTR$(PEEK(251));TAB(4);STR$(PEEK
    (252));TAB(9);                     :rem 198
170 PRINTSTR$(PEEK(253));TAB(14);STR$(PEE
    K(254))                             :rem 37
180 PRINTCHR$(145);TAB(20);"F("C$")=";
                                        :rem 16
190 PRINTX                             :rem 127
200 FORC1=0TO38:PRINTCHR$(96);:NEXTC1:PRI
    NT                                 :rem 223
210 FORC1=0TO3:POKE251+C1,8:NEXTC1:rem 34
220 NEXTC                               :rem 22
230 POKE768,139:POKE769,227              :rem 8
240 IFPEEK(252)>=112THENLIST70          :rem 21
300 DATA 133,251,8,104,133,252,134,253,13
    2,254,72,40,165,252,76,6,169:rem 65
```

©

Hi-Res VIC Drawing

Jeff Wise

There comes a time when programmers want more subtle graphics than can be achieved with characters and low resolution. Do you ever feel like creating swirling, intricate webs of delicate, slender lines? Here's how to achieve high resolution on the VIC.

The designers of the VIC-20 thoughtfully included in the VIC chip a special programmable character generator. Though mainly intended for creating custom alphabets and symbols, it can also be used to generate an entire high-resolution screen.

Each character that the VIC puts on the screen, whether user-defined or standard, is stored in eight bytes of memory. Each byte defines one of the eight rows that comprises a VIC character. Furthermore, each of the rows is split up into eight sections corresponding to the eight bits in that row's byte. If a bit is *on* (there is a 1 in its location), then its chunk of the row is lit up. If a bit is *off* (it contains a 0), then its chunk of the row is blank.

Character Matrices

"Microdraw" sets up a matrix of 12 × 15 custom characters, all of which are initially made blank (by POKEing 0 into the defining bytes). Since each character is defined by eight bytes of eight bits each, we have a total of 11,520 bits, or dots, on the screen which we can turn on or off at will. To light up a dot, simply POKE a 1 into its corresponding bit.

Such high resolution comes at a price. In order to use custom characters, we first must set the character memory apart from the BASIC program area. Since we are using so many characters, a lot of memory is consumed—1.5K, nearly half the memory available in an unexpanded VIC.

Now that we've covered the theory, it's time to enjoy your VIC's hi-res capability. Type in Microdraw, save it, then run it. Plug in your joystick, if you have one. Select the foreground and background colors for the drawing area by pressing the number key with the appropriate color on it. The program will then set up the drawing area and display the cursor. You control the cursor by moving the joystick in the direction you want the cursor to go.

Initially, the cursor is in the erase mode, which means that the cursor does not create a line as it travels and will erase any line it comes in contact with. In this mode the TV speaker emits a low beeping tone.

To change to the drawing mode, press the fire button on the joystick. The TV speaker will then beep in a higher-pitched tone, and the cursor will leave a line as it travels. To change back to the erase mode, simply press the fire button once again.

The SAVE Function

The function keys offer three additional options: The f1 key erases the drawing screen and leaves the cursor in position; f3 starts the program from the beginning and resets all variables; and f5 causes the program to jump to a screen-saving routine. The saving routine is self-explanatory, as is the retrieval routine. To replay the data you have stored, choose selection 2 ("load an old one").

If you do not have a joystick, a simple modification will allow you to use the keyboard instead. Delete lines 330 and 340, and change line 320 to read:

```
320 A=PEEK(197):J0=-(A=44):J1=-(A=36):J2=
    -(A=20):J3=-(A=12):IF A=32THENB=ABS(B
    -1)                              rem 155
```

Now the cursor's up, down, left, and right motions are controlled by the I, M, J, and K keys, respectively. To change modes, press the space bar instead of the fire button. It is not necessary to hit the control keys repeatedly; the cursor will move as long as the key is held down. If no key is pressed, the cursor will stop. In all other respects, the program works as before.

Microdraw

```
10 POKE36869,240:POKE52,24:POKE56,24:POKE
   36879,27:CLR                        :rem 84
40 PRINT"{CLR}{3 DOWN}1) DRAW A NEW PICTU
   RE.{DOWN}2) LOAD AN OLD ONE.":POKE198,
   0                                  :rem 105
60 GETA$:IFA$<>"1"ANDA$<>"2"THEN60
                                      :rem 133
70 IFA$="2"THEN450                    :rem 214
80 PRINT"{CLR}{3 DOWN}BORDER COLOR?":GOSU
   B100:G=VAL(A$)-1                   :rem 123
90 PRINT"{3 DOWN}BACKGROUND COLOR?":GOSUB
   100:H=VAL(A$):GOTO120              :rem 180
100 GETA$:IFA$<"1"ORA$>"8"THEN100   :rem 53
110 RETURN                            :rem 114
120 POKE36879,G+16*H-8:PRINT"{CLR}"
                                       :rem 77
130 FORX=0TO21:FORY=0TO22:POKE7680+X+22*Y
    ,160:POKE38400+X+22*Y,G:NEXT:NEXT
                                       :rem 44
140 IFW=1THEN160                      :rem 175
150 FORI=6144TO7679:POKEI,0:NEXT:FORI=742
    4TO7431:POKEI,255:NEXT            :rem 182
160 C=0:POKE36869,254:FORX=5TO16:FORY=3TO
    18:IFX+Y=34THENNEXT:GOTO180       :rem 53
170 C=C+1:POKE7680+X+22*Y,C:NEXT:NEXT
                                      :rem 235
180 Y=18:FORX=3TO18:POKE7680+X+22*Y,160:N
    EXT:C=90                          :rem 191
190 IFJ1THENF=F+1:IFF>7AND(C+1)/16<>INT((
    C+1)/16)THENF=0:C=C+1:GOTO210  :rem 82
200 IFF>7THENF=7                      :rem 197
210 IFJ3THENF=F-1:IFF<0AND(C-1)/16<>INT((
    C-1)/16)THENF=7:C=C-1:GOTO230  :rem 85
220 IFF<0THENF=0                      :rem 183
230 IFJ0THENE=E+1:IFE>7ANDC<177THENE=0:C=
    C+16:GOTO250                      :rem 222
240 IFE>7THENE=7                      :rem 199
250 IFJ2THENE=E-1:IFE<0ANDC>16THENE=7:C=C
    -16:GOTO270                       :rem 176
260 IFE<0THENE=0                      :rem 185
270 POKE6144+(8*C)+F,PEEK(6144+(8*C)+F)AN
    DNOT(2↑(7-E))                     :rem 203
280 POKE6144+(8*C)+F,PEEK(6144+(8*C)+F)OR
    2↑(7-E)                            :rem 88
290 POKE36878,15:POKE36874+2*B,130+INT(C/
    2.14):POKE36878,0                 :rem 221
300 POKE36874+2*ABS(B-1),0            :rem 120
310 IFB=0THENPOKE6144+(8*C)+F,PEEK(6144+(
    8*C)+F)-2↑(7-E)                    :rem 75
320 POKE37154,127:Z=128ANDPEEK(37152):J0=
    -(Z=0)                            :rem 123
330 POKE37154,255:Z=PEEK(37151)     :rem 217
340 J1=-((ZAND8)=0):J2=-((ZAND16)=0):J3=-
    ((ZAND4)=0):J=-((ZAND32)=0):IFJTHENB=
    ABS(B-1)                           :rem 91
350 GETA$:IFA$<>CHR$(133)ANDA$<>CHR$(134)
    ANDA$<>CHR$(135)THEN190           :rem 112
360 IFA$=CHR$(134)THEN10               :rem 68
370 IFA$=CHR$(135)THEN390             :rem 129
380 FORI=6144TO7423:POKEI,0:NEXT:FORI=743
    2TO7679:POKEI,0:NEXT:GOTO190    :rem 92
390 POKE36869,240:POKE36879,27      :rem 172
400 PRINT"{CLR}{DOWN}{RVS}T{OFF}APE OR
    {RVS}D{OFF}ISK?":POKE198,0       :rem 237
402 GETA$:IFA$<>"T"ANDA$<>"D"THEN402
                                       :rem 26
405 IFA$="T"THENPRINT"REWIND TAPE":rem 85
415 PRINT"HIT A KEY WHEN READY"     :rem 33
420 B$="":GETB$:IFB$<>" "THEN420 :rem 175
425 IFA$="T"THENOPEN1,1,1           :rem 176
426 IFA$="D"THENINPUT"FILENAME";N$:N$=N$+
    ",S,W":D=8:OPEN1,8,5,N$            :rem 5
430 PRINT#1,G:PRINT#1,H:FORA=6144TO7679:P
    RINT#1,PEEK(A):NEXT:CLOSE1      :rem 192
440 PRINT"{CLR}YOUR PICTURE IS SAVED.":GO
    TO10                              :rem 120
450 PRINT"{DOWN}{RVS}T{OFF}APE OR {RVS}D
    {OFF}ISK?":D=1:N$=""               :rem 17
455 GETA$:IFA$<>"T"ANDA$<>"D"THEN455
                                       :rem 42
460 IFA$="T"THENPRINT"{3 DOWN}INSERT CASS
    ETTE AND{3 SPACES}REWIND IT"      :rem 0
470 PRINT"{DOWN}WHEN YOU ARE READY HIT(SP
    ACE).{DOWN}":W=1:WAIT198,1       :rem 71
475 IFA$="T"THENOPEN1,1,0           :rem 180
480 IFA$="D"THENINPUT"FILENAME";N$:D=8:N$
    =N$+",S,R":OPEN1,8,5,N$            :rem 0
490 INPUT#1,G:INPUT#1,H:FORA=6144TO7679:I
    NPUT#1,C:POKEA,C:NEXT:CLOSE1 :rem 116
500 GOTO120                            :rem 97
10000 OPEN15,8,15:INPUT#15,A,B$,C,D:CLOSE
      15                              :rem 198
```

©

ML Applesort

Richard Salley

A machine language program is necessary when sorting large amounts of data. "ML Applesort" is a machine language utility that will quickly sort an array of any length.

In COMPUTE! (September 1982), David Lummis presented an excellent machine language sort routine for PET/CBM computers. "ML Applesort" is a modification of Lummis's program for the Apple.

The original program has been compressed to fit it into page 3 of memory. This is a safe area for ML programs so you do not have to worry about overwriting the routine with a long program or numerous and lengthy string arrays. The zero page locations used for temporary storage, counting, and address indexing were chosen because most of these locations are used primarily in connection with hi-res graphics, and it is unlikely such programs will be used concurrently with a sort utility.

ML Applesort makes use of the Apple's special & command. When the BASIC interpreter encounters the & in a program, it goes to location \$3F5 (1013 decimal) and then performs an unconditional jump to the address contained in \$3F6 and \$3F7. In this program, 0 is placed in \$3F6 (1014 decimal) and 3 is placed in \$3F7 (1015 decimal). This will cause a jump to location \$300 (768 decimal), which is where the machine language routine begins.

The first instruction at \$300 is a JSR (Jump to SubRoutine) \$F7D9. This is a monitor subroutine that fetches the address of a string and stores it in locations \$9B and \$9C. By placing the name of the string array we want sorted immediately after the &, this routine will tell us where in memory that array is stored. The correct format for calling the ML sort routine from BASIC is as follows:

```
100 &X$
```

where X\$ is the name of the array to be sorted. When the routine returns to BASIC, the named array will be sorted alphabetically in ascending order. How the program does the sorting can be understood by studying a disassembly. To enter the program, use the BASIC Loader (Program 1).

After placing the program into memory by running Program 1, save it to disk by typing:

```
BSAVE APPLESORT, A$300, L$FF
```

You can then BLOAD the sort routine and use it with any of your own BASIC programs.

Program 2 shows how easily the program can be used and how quickly it can sort an array with strings of varying lengths. I'm sure COMPUTE! readers with Apple machines will find numerous applications for this useful utility.

Program 1: ML Applesort

```
100  REM  ...ML APPLESORT
110  REM  ...POKE & JUMP ADDRESS
```

```
120  POKE 1013,76: POKE 1014,0: POKE 10
     15,3
130  REM  ...POKE ML
140  FOR ADDR = 768 TO 941: READ CODE:C
     KSUM = CKSUM + CODE: POKE ADDR,COD
     E: NEXT
150  IF CKSUM <  > 26104 THEN  PRINT "E
     RROR IN DATA STATEMENTS": STOP
768  DATA  32, 217, 247, 165, 155, 133,
      1, 165
776  DATA  156, 133, 2, 160, 5, 177, 1,
      133
784  DATA  208, 200, 177, 1, 133, 209,
     169, 1
792  DATA  133, 210, 169, 0, 133, 211,
     24, 165
800  DATA  1, 105, 7, 133, 235, 165, 2,
     105
808  DATA  0, 133, 236, 165, 235, 133,
     225, 165
816  DATA  236, 133, 226, 24, 165, 225,
      105, 3
824  DATA  133, 235, 165, 226, 105, 0,
     133, 236
832  DATA  160, 0, 177, 225, 208, 34, 2
     4, 165
840  DATA  210, 105, 1, 133, 210, 165,
     211, 105
848  DATA  0, 133, 211, 197, 208, 144,
     212, 165
856  DATA  210, 197, 209, 144, 206, 165
     , 212, 208
864  DATA  1, 96, 169, 0, 133, 212, 240
     , 174
872  DATA  133, 213, 177, 235, 240, 239
     , 133, 214
880  DATA  200, 177, 225, 133, 233, 177
     , 235,133
888  DATA  237, 200, 177, 225, 133, 234
     , 177, 235
896  DATA  133, 238, 160, 0, 177, 233,
     209, 237
904  DATA  144, 188, 240, 2, 176, 9, 2
     00, 196
912  DATA  213, 240, 179, 196, 214, 208
     , 237, 160
920  DATA  2, 177, 225, 72, 177, 235,
     145, 225
928  DATA  104, 145, 235, 136, 16, 243,
     169, 1
936  DATA  133, 212, 76, 70, 3, 0
```

Program 2: ML Applesort Demo

```
10  REM ...SORT DEMO
20  IF  PEEK (768) = 32 THEN 40
30  PRINT  CHR$ (4);"BLOAD APPLESORT"
40  HOME : VTAB 5: PRINT "POINTER SORT
    DEMO"
50  VTAB 7: INPUT "ENTER # OF STRINGS T
    O SORT ";N
60  DIM R$(N)
70 B$ = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
80  FLASH : PRINT : PRINT "BUILDING STR
    INGS"
90  NORMAL
100  FOR F = 1 TO N:X =  INT ( RND (1) *
    7) + 2: FOR G = 1 TO X
110 R$(F) = R$(F) +  MID$ (B$, INT ( RND
    (1) * 26) + 1,1)
120  NEXT : NEXT
130  FOR F = 1 TO N: PRINT R$(F),: NEXT
140  PRINT : INPUT "PRESS <RETURN> TO S
     ORT ";XX$
150  PRINT : PRINT "SORT BEGUN"
160  & R$
170  PRINT : PRINT "SORT FINISHED!!"
180  PRINT : INPUT "PRESS <RETURN> TO P
     RINT SORTED LIST ";XX$
190  FOR F = 1 TO N: PRINT R$(F),: NEXT
200  PRINT : PRINT "END OF DEMO": END
```

©

Questions Beginners Ask

Tom R. Halfhill, Staff Editor

Are you thinking about buying a computer for the first time, but you don't know much about computers? Or maybe you just purchased a computer and are still a bit baffled. Each month in this column, COMPUTE! *will answer questions frequently asked by beginners.*

Q Is it safe to plug a whole home computer system into a single wall socket? I'm talking about a computer, TV, cassette recorder, disk drive, printer, and a modem. Will it blow a fuse? Or will I have to run extension cords from nearby wall sockets for some of the peripherals?

A As long as no other power-hungry appliance is on the same socket, plugging a whole home computer system into one outlet is perfectly safe. Home computers and their peripherals actually don't use very much electricity at all. In fact, the typical home computer consumes less power than the light bulb you'll burn to see it by.

For instance, one of our editors has a computer system at home which consists of an Atari 800 with 48K of memory, a disk drive, a cassette recorder, an Atari 850 Interface Module, a color monitor, an 80-column dot-matrix printer, and a modem. Everything but the monitor and the modem is plugged into a six-socket power strip, which in turn is plugged into a single wall outlet. The power strip has a 15-amp circuit breaker which has never popped. That means the system uses less than 1650–1800 watts, or a little more electricity than a blow dryer.

By far the most power-hungry component of a home computer system is the TV set or monitor. A small black-and-white TV or monochrome monitor won't use much electricity, but a large color TV can use more power than the rest of the system put together. If you're worried about overloads, plug the TV into a different outlet.

One thing you should avoid is hooking up the computer system to a circuit shared by heavy-duty appliances like air conditioners, dishwashers, clothes washers, dryers, refrigerators, and water heaters. Have you ever noticed your room lights dim for a second when a heavy appliance kicks on? The sudden demand for power momentarily drains the circuit and lowers the voltage. Those kinds of fluctuations aren't healthy for computers, whose chips are very sensitive to power sags and surges. (That's why some people invest in surge protectors or voltage stabilizers.)

If you aren't sure whether a certain wall outlet is wired to the same circuit as another outlet serving a heavy appliance, test it by plugging in a lamp. Then switch on some of the major appliances in your home while someone watches the lamp for any telltale dimming. If an outlet is affected, you may have to run an extension cord from a more distant socket to reach your computer system. This is particularly true in houses and apartment buildings with older wiring.

Q I'm moving to another state and I'd like to transport my computer by plane. Do you think it would be safe in the baggage compartment?

A Recently some of COMPUTE!'s editors went on a trip to the Comdex trade show in Las Vegas and witnessed some unpleasant violence to a Compaq transportable computer. Although the Compaq is one of the better transportables, by the time the poor computer tumbled off the airport conveyor belt onto the revolving baggage-claim carousel, it looked almost destroyed. The top of the case was torn off, exposing the built-in monitor screen and delicate disk drives. Heavy hard-shell suitcases kept sliding off the conveyor belt and bashing into the computer, knocking more parts loose. Wires and cables were hanging out. It wasn't pretty.

Based on what we saw that day, and on other airline experiences, our advice is not to ship a computer as baggage unless it's *very* well packed and padded, preferably in its original box with the form-fitting Styrofoam inserts. Have you ever seen the TV commercial in which a suitcase is batted around by an ape? If your computer is packed well enough to withstand that kind of battering, you're probably safe. Otherwise, you might consider another method of shipping.

Incidentally, if you're traveling by air with a computer as carry-on baggage, insist on having it hand-checked when passing through security checkpoints. We know of a newspaper reporter who unknowingly allowed his TRS-80 Model 100 lap computer to suffer exposure from an airport x-ray machine. "It just went crazy," he said.©

NEWS&PRODUCTS

Percussion Emulator For Apple

Drum-Key, recently introduced by Peripheral Visions, Inc., is an electronic music interface board for use with the Apple II series of computers. It will allow you to interface stereos and electric instrument amplifiers to your computer.

Drum-Key lets you compose, play, and record percussion sounds and riffs, as well as play along with the 100 included rhythm patterns and 26 songs.

A complement of 28 sounds is included. Among these are snare, tom-toms, cymbals, cowbell, tambourine, and six sounds made by conventional drum synthesizers.

Suggested retail price is $139.

Peripheral Visions, Inc.
Great Valley Parkway
Malvern, PA 19355
(215) 627-3535

RS-232 Modem Adapter For Atari

Advanced Interface Devices, Inc., has announced the R-Verter, a serial bus modem adapter for Atari 400, 600XL, 800, and 800XL home computers.

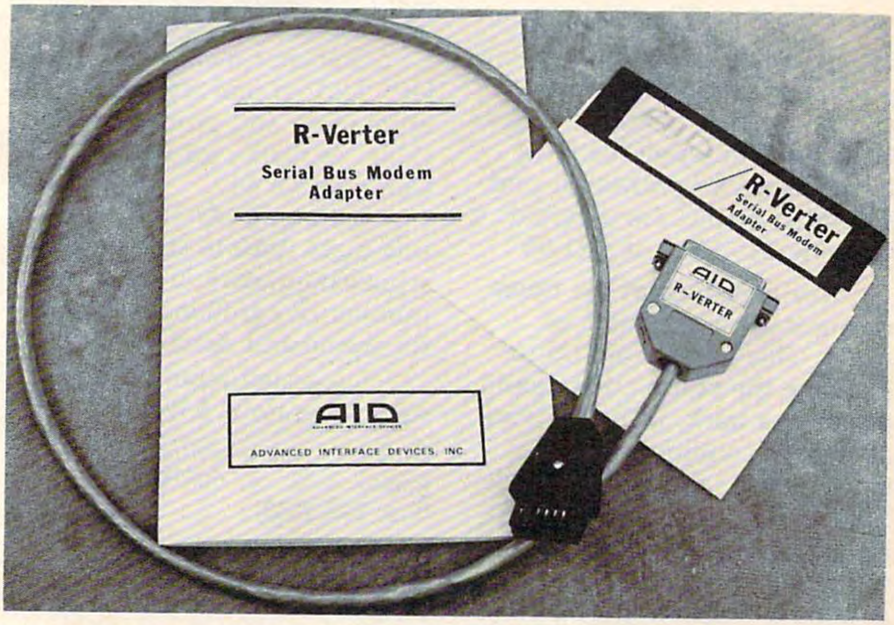
The R-Verter allows most modems and other RS-232C devices to be used directly with Atari computers without using the Atari 850 Interface Module or other interface. It requires no modifications of the computer or other peripherals. All circuitry is contained in an RS-232C type connector to minimize size.

The R-Verter comes with a software package which includes a smart terminal emulator and an RS-232C device handler.

Most common RS-232C handshaking configurations are available using internal jumpers.

Price for the R-Verter and print echo software is $49.95.

Advanced Interface Devices, Inc.
P.O. Box 2188
Melbourne, FL 32902
(305) 676-1275



The R-Verter allows most modems and other RS-232C devices to be used directly with Atari computers without using the Atari 850 Interface Module.

Game Development Program For The Commodore 64

Aspiring arcade-game designers can develop graphics for their games more quickly and easily by using the *Graphics Master*.

Written for the Commodore 64, this programming aid adds 52 new commands to BASIC and has numerous features that support game development.

Software Unlimited will soon release a compiler to make the completed game run faster.

Disk only; $29.95. (Please include $3.00 for postage and handling.)

Software Unlimited
P.O. Box 429
Klamath Falls, OR 97601

Educational And Entertainment Software For The TI-99/4A

American Software has announced four new software packages for the Texas Instruments 99/4A.

In *Fireball*, an arcade game for ages ten and older, you must climb a volcano without being hit by fireballs or falling into holes. The game requires either the Editor/Assembler cartridge or the Mini-Memory cartridge. Disk only; $16.95.

Letter Fun helps preschoolers learn the letters of the alphabet using colorful graphics and music. The child can choose from three different learning levels. Speech Synthesizer and Extended BASIC are required. Cassette $19.95; disk $21.95.

Try your luck at the horse racing track with *American Derby*. This game is set up to simulate the betting that would go on at a track, including variable track conditions, an insider's sheet, and realistic odds. You can bet on up to 36 different horses. Designed for ages ten to adult; up to six may play at a time. Requires Extended BASIC. Cassette $14.95; disk $16.95.

Speed Read was written for adults who want to improve their reading speed. This package of programs includes information on the reading process as well as pacing aids and reading passages to test your speed. It requires Extended BASIC. Cassette $29.95; disk $31.95 (disk version requires memory expansion).

American Software Design & Distribution Co.
P.O. Box 46
Cottage Grove, MN 55016
(612) 459-0557

Home Educational Software For Apple, Atari, And Commodore

Sunburst Communications, which has supplied educational materials to schools for 12 years, has released three new products from their microcomputer division.

The Incredible Laboratory (ages seven to adult) uses the problem-solving strategy of trial and error and note-taking to discover what combinations of mysterious chemicals make up crazy monsters. Apple and Atari versions are available.

Challenge Math (ages 6–11) lets children practice basic math, estimation, and problem-solving skills. Available for Apple and Commodore 64.

Getting Ready To Read And Add (ages three to six) gives preschoolers practice in letter and number recognition. Available for Apple and Atari.

Suggested retail price for each program is $39.95.

Sunburst Communications, Inc.
Pleasantville, NY 10570
(914) 769-5030

New Telecommunications Package For Apple

The Networker modem, recently introduced by ZOOM Telephonics, is a complete telecommunications package for the Apple II, II+, and IIe computers.

For $129, you get a single-slot, direct-connect, 300-baud modem, terminal software, and a free subscription to The Source.

An enhanced version of the terminal software, Netmaster, can be purchased separately for

Apple owners can get a complete telecommunications package, including modem and terminal software, by purchasing the Netmaster system.

$79. If purchased with the Networker, the price of the entire package is $179.

ZOOM Telephonics plans to offer a complete line of modems, including modems for the IBM-PC.

ZOOM Telephonics
207 South St.
Boston, MA 02111
(617) 423-1072

Telecommunications Aid

Source Telecomputing Corporation (STC) has announced *Apple Sourcelink*, the second in its series of communications software designed to supplement use of The Source by personal computer owners.

The software is compatible with the new Apple modem, as well as with the Hayes and Transend modem products, and is designed for the Apple II, IIe, and II+ with a minimum 48K of memory.

It combines features such as automatic dial-up and sign-on procedure for Telenet, Uninet, and Sourcenet data communications networks; "one-button" access to major services on The Source; simultaneous capture of data from The Source in the Apple memory or disks, including a capture editor; and data transfer from Apple disks to The Source, or vice versa, while on-line.

An additional feature allows Apple and IBM users to access automatically any number of predetermined services and data bases, once on-line.

The Source
1616 Anderson Road
McLean, VA 22102
(703) 734-7500

Inexpensive Light Pen For Commodore Computers

Creative Electronics has announced the introduction of a new light pen for the Commodore 64 and VIC-20.

The light pen, which offers close to one-pixel accuracy for high-resolution graphics, comes with two sample programs.

Both versions retail for $14.95.

Creative Electronics
P.O. Box 4253
1714 Sandalwood
Thousand Oaks, CA 91360
(805) 492-1506

Alphabet, Math Games For Children

Two educational software games designed to help children understand the alphabet, multiplication, and division have been introduced by Avalon Hill Game Company's Intelligence Quest Software division.

DIVEX, appropriate for ages 8–12, has three levels of multiplication and division to master, and requires a child to use mathematical skills to protect his or her "land" from incorrect answers.

It is available on diskette ($21) for Ataris with at least 32K memory. Cassette editions ($16) for the Commodore 64 and Atari will be available later.

In *ABC Caterpillar*, the player, controlling a bright green caterpillar, searches for letters of the alphabet as they pass by on the screen. The goal is to find and gobble up the letters in alphabetical order.

For children 3–8 years old, *ABC Caterpillar* is available for the VIC-20 at a cost of $16. A Commodore 64 edition is planned also.

Intelligence Quest Software
4517 Harford Road
Baltimore, MD 21214
(301) 254-9200

New Data Base Management Software For IBM Home Computers

Condor Jr. is a data base management system specially customized for beginning microcomputer users.

The program is available for the IBM-PC and PCjr, and retails for $195.

Beyond its extensive math and printing capabilities, *Condor Jr.* can be upgraded to *Condor 3* (a more sophisticated data base manager). Other features include multilevel sorts and a variety of report generators.

Condor
2051 S. State St.
Ann Arbor, MI 48104
(313) 769-2418

New Speech Synthesizers

Three new Voice Box speech synthesizers have been introduced by The Alien Group, two of which are designed for Apple II and Apple-compatible computers. The third is for any machine which has a standard (RS-232C) serial port.

Using a new speech chip, the programs produce speech directly from English text, adding inflection either automatically or according to numbers inserted by the user. All units have an unlimited vocabulary, and can speak with a male or female voice, fast or slow, or loud or soft, depending on what commands are added to the text. It is not necessary to mark syllable boundaries or to use phoneme spelling when adding intonation.

The Voice Box 3m model, designed for the Apple, retails for $129. Voice Box 3i, also for the Apple, costs $219. Prices include a Voice Box board, disk software, and external speaker.

The Voice Box 3s, which can connect to any computer via the standard RS-232C serial interface, includes an integral speaker and retails for $269.

The Alien Group
27 West 23rd St.
New York, NY 10010
(212) 741-1770

New Product releases are selected from submissions for reasons of timeliness, available space, and general interest to our readers. We regret that we are unable to select all new product submissions for publication. Readers should be aware that we present here some edited version of material submitted by vendors and are unable to vouch for its accuracy at time of publication.

COMPUTE! *welcomes notices of upcoming events and requests that the sponsors send a short description, their name and phone number, and an address to which interested readers may write for further information. Please send notices at least three months before the date of the event, to: Calendar, P.O. Box 5406, Greensboro, NC 27403.* ©

MLX Machine Language Entry Program For Commodore 64

Charles Brannon, Program Editor

MLX is a labor-saving utility that allows almost fail-safe entry of machine language programs published in COMPUTE!. *You need to know nothing about machine language to use MLX—it was designed for everyone.*

MLX is a new way to enter long machine language (ML) programs with a minimum of fuss. MLX lets you enter the numbers from a special list that looks similar to BASIC DATA statements. It checks your typing on a line-by-line basis. It won't let you enter illegal characters when you should be typing numbers. It won't let you enter numbers greater than 255 (forbidden in ML). It won't let you enter the wrong numbers on the wrong line. In addition, MLX creates a ready-to-use tape or disk file. You can then use the LOAD command to read the program into the computer:

LOAD "filename",1,1 (for tape)
LOAD "filename",8,1 (for disk)

To start the program, you enter a SYS command that transfers control from BASIC to machine language. The starting SYS number appears in the article.

Using MLX

Type in and save MLX for your 64 (you'll want to use it in the future). When you're ready to type in an ML program, run MLX. MLX asks you for two numbers: the starting address and the ending address. These numbers are given in the article accompanying the ML program.

You'll see a prompt corresponding to the starting address. The prompt is the current line you are entering from the listing. It increases by six each time you enter a line. That's because each line has seven numbers—six actual data numbers plus a *checksum number*. The checksum verifies that you typed the previous six numbers correctly. If you enter any of the six numbers wrong, or enter the checksum wrong, the computer rings a buzzer and prompts you to reenter the line. If you enter it correctly, a bell tone sounds and you continue to the next line.

MLX accepts only numbers as input. If you make a typing error, press the INST/DEL key; the entire number is deleted. You can press it as many times as necessary back to the start of the line. If you enter three-digit numbers as listed, the computer automatically prints the comma and goes on to accept the next number. If you enter less than three digits, you can press either the SPACE bar, or RETURN key to advance to the next number. The checksum automatically appears in inverse video for emphasis.

To simplify your typing, MLX redefines part of the keyboard as a numeric keypad (lines 581–584):

	U	I	O			7	8	9
H	J	K	L	become	0	4	5	6
	M	,	.			1	2	3

MLX Commands

When you finish typing an ML listing (assuming you type it all in one session), you can then save the completed program on tape or disk. Follow the screen instructions. If you get any errors while saving, you probably have a bad disk, or the disk is full, or you've made a typo when entering the MLX program itself.

You don't have to enter the whole ML program in one sitting. MLX lets you enter as much as you want, save it, and then reload the file from tape or disk later. MLX recognizes these commands:

SHIFT-S: Save
SHIFT-L: Load
SHIFT-N: New Address
SHIFT-D: Display

When you enter a command, MLX jumps out of the line you've been typing, so we recommend you do it at a new prompt. Use the Save command to save what you've been working on. It will save on tape or disk as if you've finished, but the tape or disk won't work, of course, until you finish the typing. Remember what address you stop at. The next time you run MLX, answer all the prompts as you did before, then insert the disk or tape. When you get to the entry prompt, press SHIFT-L to reload the partly completed file into memory. Then use the New Address command to resume typing.

To use the New Address command, press SHIFT-N and enter the address where you previously stopped. The prompt will change, and you can then continue typing. Always enter a New Address that matches up with one of the line numbers in the special listing, or else the checksum won't work. The Display command lets you display a section of your typing. After you press SHIFT-D, enter two addresses within the line number range of the listing. You can abort the listing by pressing any key.

What if you forgot where you stopped typing? Use the Display command to scan memory from the beginning to the end of the program. When you reach the end of your typing, the lines will contain a random pattern of numbers. When you see the end of your typing, press any key to stop the listing. Use the New Address command to continue typing from the proper location.

MLX: Machine Language Entry

```
10 REM LINES CHANGED FROM MLX VERSION 2.0
   0 ARE 750,765,770 AND 860        :rem 50
100 PRINT"{CLR}[6]";CHR$(142);CHR$(8);:
    POKE53281,1:POKE53280,1         :rem 67
101 POKE 788,52:REM DISABLE RUN/STOP
                                   :rem 119
110 PRINT"{RVS}{39 SPACES}";       :rem 176
120 PRINT"{RVS}{14 SPACES}{RIGHT}{OFF}
    [*]£{RVS}{RIGHT} {RIGHT}{2 SPACES}
```

```
    [*]{OFF}[*]£{RVS}£{RVS}
    {14 SPACES}";                                     :rem 250
130 PRINT"{RVS}{14 SPACES}{RIGHT} [G]
    {RIGHT} {2 RIGHT} {OFF}£{RVS}£[*]
    {OFF}[*]{RVS}{14 SPACES}";                        :rem 35
140 PRINT"{RVS}{41 SPACES}"                           :rem 120
200 PRINT"{2 DOWN}{PUR}{BLK} MACHINE LANG
    UAGE EDITOR VERSION 2.01{5 DOWN}"
                                                      :rem 237
210 PRINT"[5]{2 UP}STARTING ADDRESS?
    {8 SPACES}{9 LEFT}";                              :rem 143
215 INPUTS:F=1-F:C$=CHR$(31+119*F)
                                                      :rem 166
220 IFS<256OR(S>40960ANDS<49152)ORS>53247
    THENGOSUB3000:GOTO210                             :rem 235
225 PRINT:PRINT:PRINT                                 :rem 180
230 PRINT"[5]{2 UP}ENDING ADDRESS?
    {8 SPACES}{9 LEFT}";:INPUTE:F=1-F:C$=
    CHR$(31+119*F)                                    :rem 20
240 IFE<256OR(E>40960ANDE<49152)ORE>53247
    THENGOSUB3000:GOTO230                             :rem 183
250 IFE<STHENPRINTC$;"{RVS}ENDING < START
    {2 SPACES}":GOSUB1000:GOTO 230
                                                      :rem 176
260 PRINT:PRINT:PRINT                                 :rem 179
300 PRINT"{CLR}";CHR$(14):AD=S:POKEV+21,0
                                                      :rem 225
310 A=1:PRINTRIGHT$("0000"+MID$(STR$(AD),
    2),5);":";                                        :rem 33
315 FORJ=ATO6                                         :rem 33
320 GOSUB570:IFN=-1THENJ=J+N:GOTO320
                                                      :rem 228
390 IFN=-211THEN 710                                  :rem 62
400 IFN=-204THEN 790                                  :rem 64
410 IFN=-206THENPRINT:INPUT"{DOWN}ENTER N
    EW ADDRESS";ZZ                                    :rem 44
415 IFN=-206THENIFZZ<SORZZ>ETHENPRINT"
    {RVS}OUT OF RANGE":GOSUB1000:GOTO410
                                                      :rem 225
417 IFN=-206THENAD=ZZ:PRINT:GOTO310
                                                      :rem 238
420 IF N<>-196 THEN 480                               :rem 133
430 PRINT:INPUT"DISPLAY:FROM";F:PRINT,"TO
    ";:INPUTT                                         :rem 234
440 IFF<SORF>EORT<SORT>ETHENPRINT"AT LEAS
    T";S;"{LEFT}, NOT MORE THAN";E:GOTO43
    0                                                 :rem 159
450 FORI=FTOTSTEP6:PRINT:PRINTRIGHT$("000
    0"+MID$(STR$(I),2),5);":";                        :rem 30
451 FORK=0TO5:N=PEEK(I+K):PRINTRIGHT$("00
    "+MID$(STR$(N),2),3);",";                         :rem 66
460 GETA$:IFA$>""THENPRINT:PRINT:GOTO310
                                                      :rem 25
470 NEXTK:PRINTCHR$(20);:NEXTI:PRINT:PRIN
    T:GOTO310                                         :rem 50
480 IFN<0 THEN PRINT:GOTO310                          :rem 168
490 A(J)=N:NEXTJ                                      :rem 199
500 CKSUM=AD-INT(AD/256)*256:FORI=1TO6:CK
    SUM=(CKSUM+A(I))AND255:NEXT                       :rem 200
510 PRINTCHR$(18);:GOSUB570:PRINTCHR$(146
    );                                                :rem 94
511 IFN=-1THENA=6:GOTO315                             :rem 254
515 PRINTCHR$(20):IFN=CKSUMTHEN530
                                                      :rem 122
520 PRINT:PRINT"LINE ENTERED WRONG : RE-E
    NTER":PRINT:GOSUB1000:GOTO310:rem 176
530 GOSUB2000                                         :rem 218
540 FORI=1TO6:POKEAD+I-1,A(I):NEXT:POKE54
    272,0:POKE54273,0                                 :rem 227
550 AD=AD+6:IF AD<E THEN 310                          :rem 212
560 GOTO 710                                          :rem 108
570 N=0:Z=0                                           :rem 88
580 PRINT"[£]";                                       :rem 81
581 GETA$:IFA$=""THEN581                              :rem 95
582 AV=-(A$="M")-2*(A$=",")-3*(A$=".")-4*
    (A$="J")-5*(A$="K")-6*(A$="L"):rem 41
583 AV=AV-7*(A$="U")-8*(A$="I")-9*(A$="O"
    ):IFA$="H"THENA$="0"                              :rem 134
584 IFAV>0THENA$=CHR$(48+AV)                          :rem 134
585 PRINTCHR$(20);:A=ASC(A$):IFA=13ORA=44
    ORA=32THEN670                                     :rem 229
590 IFA>128THENN=-A:RETURN                            :rem 137
600 IFA<>20 THEN 630                                  :rem 10
610 GOSUB690:IFI=1ANDT=44THENN=-1:PRINT"
    {OFF}{LEFT} {LEFT}";:GOTO690                      :rem 62
620 GOTO570                                           :rem 109
630 IFA<48ORA>57THEN580                               :rem 105
640 PRINTA$;:N=N*10+A-48                              :rem 106
650 IFN>255 THEN A=20:GOSUB1000:GOTO600
                                                      :rem 229
660 Z=Z+1:IFZ<3THEN580                                :rem 71
670 IFZ=0THENGOSUB1000:GOTO570                        :rem 114
680 PRINT",";:RETURN                                  :rem 240
690 S%=PEEK(209)+256*PEEK(210)+PEEK(211)
                                                      :rem 149
691 FORI=1TO3:T=PEEK(S%-I)                            :rem 67
695 IFT<>44ANDT<>58THENPOKES%-I,32:NEXT
                                                      :rem 205
700 PRINTLEFT$("{3 LEFT}",I-1);:RETURN
                                                      :rem 7
710 PRINT"{CLR}{RVS}*** SAVE ***{3 DOWN}"
                                                      :rem 236
715 PRINT"{2 DOWN}(PRESS {RVS}RETURN{OFF}
    ALONE TO CANCEL SAVE){DOWN}":rem 106
720 F$="":INPUT"{DOWN} FILENAME";F$:IFF$=
    ""THENPRINT:PRINT:GOTO310                         :rem 71
730 PRINT:PRINT"{2 DOWN}{RVS}T{OFF}APE OR
    {RVS}D{OFF}ISK: (T/D)"                            :rem 228
740 GETA$:IFA$<>"T"ANDA$<>"D"THEN740
                                                      :rem 36
750 DV=1-7*(A$="D"):IFDV=8THENF$="0:"+F$:
    OPEN15,8,15,"S"+F$:CLOSE15                        :rem 212
760 T$=F$:ZK=PEEK(53)+256*PEEK(54)-LEN(T$
    ):POKE782,ZK/256                                  :rem 3
762 POKE781,ZK-PEEK(782)*256:POKE780,LEN(
    T$):SYS65469                                      :rem 109
763 POKE780,1:POKE781,DV:POKE782,1:SYS654
    66                                                :rem 69
765 K=S:POKE254,K/256:POKE253,K-PEEK(254)
    *256:POKE780,253                                  :rem 17
766 K=E+1:POKE782,K/256:POKE781,K-PEEK(78
    2)*256:SYS65496                                   :rem 235
770 IF(PEEK(783)AND1)OR(191ANDST)THEN780
                                                      :rem 111
775 PRINT"{DOWN}DONE.{DOWN}":GOTO310
                                                      :rem 113
780 PRINT"{DOWN}ERROR ON SAVE.{2 SPACES}T
    RY AGAIN.":IFDV=1THEN720                          :rem 171
781 OPEN15,8,15:INPUT#15,E1$,E2$:PRINTE1$
    ;E2$:CLOSE15:GOTO720                              :rem 103
790 PRINT"{CLR}{RVS}*** LOAD ***{2 DOWN}"
                                                      :rem 212
795 PRINT"{2 DOWN}(PRESS {RVS}RETURN{OFF}
    ALONE TO CANCEL LOAD)"                            :rem 82
800 F$="":INPUT"{2 DOWN} FILENAME";F$:IFF
    $=""THENPRINT:GOTO310                             :rem 144
810 PRINT:PRINT"{2 DOWN}{RVS}T{OFF}APE OR
    {RVS}D{OFF}ISK: (T/D)"                            :rem 227
820 GETA$:IFA$<>"T"ANDA$<>"D"THEN820
                                                      :rem 34
830 DV=1-7*(A$="D"):IFDV=8THENF$="0:"+F$
                                                      :rem 157
840 T$=F$:ZK=PEEK(53)+256*PEEK(54)-LEN(T$
    ):POKE782,ZK/256                                  :rem 2
```

```
841  POKE781,ZK-PEEK(782)*256:POKE780,LEN(
     T$):SYS65469                  :rem 107
845  POKE780,1:POKE781,DV:POKE782,1:SYS654
     66                             :rem 70
850  POKE780,0:SYS65493             :rem 11
860  IF(PEEK(783)AND1)OR(191ANDST)THEN870
                                   :rem 111
865  PRINT"{DOWN}DONE.":GOTO310     :rem 96
870  PRINT"{DOWN}ERROR ON LOAD.{2 SPACES}T
     RY AGAIN.{DOWN}":IFDV=1THEN800
                                   :rem 172
880  OPEN15,8,15:INPUT#15,E1$,E2$:PRINTE1$
     ;E2$:CLOSE15:GOTO800          :rem 102
1000 REM BUZZER                    :rem 135
1001 POKE54296,15:POKE54277,45:POKE54278,
     165                           :rem 207
1002 POKE54276,33:POKE 54273,6:POKE54272,
     5                              :rem 42
1003 FORT=1TO200:NEXT:POKE54276,32:POKE54
     273,0:POKE54272,0:RETURN      :rem 202
2000 REM BELL SOUND                 :rem 78
2001 POKE54296,15:POKE54277,0:POKE54278,2
     47                            :rem 152
2002 POKE 54276,17:POKE54273,40:POKE54272
     ,0                             :rem 86
2003 FORT=1TO100:NEXT:POKE54276,16:RETURN
                                    :rem 57
3000 PRINTC$;"{RVS}NOT ZERO PAGE OR ROM":
     GOTO1000                       :rem 89
```

©

CAPUTE!

Modifications Or Corrections To Previous Articles

Atari XL Compatibility Update

Upon testing with our new 800XL, we have been pleased to discover that the vast majority of our previously published Atari programs will run without modification. Of the few programs that will not run as is, almost all, including the popular "Scriptor" word processor (April 1983), operate properly when used with the Atari *Translator*. This program, available on a disk (DX5063) from Atari, enables most programs written for the older Ataris to be run on the new XL models. So far we have discovered only two programs, "Demons Of Osiris" (January 1984) and "Ski" (February 1983), that the *Translator* cannot cure. These programs can be run on the new computers only if you have a BASIC cartridge from the older Atari series to plug in.

For information on obtaining the *Translator* disk, call Atari's Customer Relations Department at 800-538-8543 (inside California, 800-672-1404).

MLX For Commodore 64

There is an error in the article accompanying the "MLX" machine language editor program in the March and May issues. The article states, "If you enter less than three digits, you can press either the comma, SPACE bar, or RETURN key to advance to the next number." However, when the numeric keypad feature was added to MLX, the comma key was redefined as the numeral 2. As a result, the comma key can no longer be used to advance to the next number; however, the SPACE bar and RETURN key still work as stated.

A number of readers have expressed concern at the number of revisions to MLX since it was first published. These changes generally represent enhancements, *not* corrections. Any version of MLX may be used to type in any program for the 64 presented in MLX format, regardless of whether the program is from COMPUTE!, COMPUTE!'s GAZETTE, or a COMPUTE! book. The only version of 64 MLX known to contain a bug is the one from the March issue, and the correction was given in the May "CAPUTE!" column.

Automatic Proofreader For The 64

The final paragraph of the article which accompanies the "Automatic Proofreader" program each month indicates that on the 64 the Proofreader can be protected during tape LOADs and SAVEs by typing POKE 178,165. Richard Murphy points out that the proper value to protect the Proofreader is POKE 178,251. This POKE is not necessary for disk operations.

64 Hi-Res Screen Printing

Many readers have asked for a way to print a copy of the elaborate designs they create with the "3-D Plotting" program from the May issue (p. 58). Reader Henry Mervis observes that, for Commodore 64 owners, the solution is in the same issue, in the "Hi-Res Graphics Editor" program (p. 82). To create a hard copy of the results of the 3-D Plotting programs (or of almost any other hi-res screen display), load the machine language program you created for the Hi-Res Editor (Program 2, p. 80), using the LOAD command format described on page 82. Remember to enter the line to move the BASIC memory area:

```
POKE 642,128:POKE 44,128:POKE 32768,0:NEW
```

Next, load either "Rectan" (Program 1, p. 60) or "Spheri" (Program 2, p. 62). For Rectan, change line 600 to read 600 SYS 49152; for Spheri, change line 610 to 610 SYS 49152. Then RUN the program in the normal manner. When your design is complete, a rectangle will appear on the screen. Turn on your printer and press the P key and your design should begin to print.

The screen dump routine will work only on a Commodore 1525 printer or with an interface that emulates the 1525. The routine will not work with Commodore 1526 printers. ©

SAVE MORE THAN EVER ON 3M Scotch® DISKETTES

LIFETIME WARRANTY!

$1.85 ea. 5¼" SSDD (744) Qty. 20 — $2.35 ea. 5¼" DSDD (745) Qty. 20

5¼" SSQD—96TPI (746)	$2.60 ea.
5¼" DSQD—96TPI (747)	$3.25 ea.
8" SSSD (740)	$2.05 ea.
8" SSDD (741)	$2.50 ea.
8" DSDD (743)	$3.10 ea.

Shipping: 5¼" DISKETTES—Add $3.00 per 100 or fraction thereof. 8" DISKETTES—Add $4.00 per 100 or fraction thereof. OTHER ITEMS: Shipping charges as shown in addition to diskette shipping charges. **Payment:** VISA or MasterCard. COD orders only, add $3.00. **Taxes:** Illinois customers, please add 8%.

Hours: 9 AM-5 PM Central Time
For fast service call
Nationwide: 1-800-621-6827
In Illinois: 312-944-2788

DISK WORLD!
Suite 4806 • 30 East Huron Street • Chicago, Illinois 60611

Authorized Distributor Information Processing Products 3M

BREAK-THRU

ONLY $34.95
SHIPPING & HANDLING ADD $2.50
Dealer inquiries welcome

A compact and inexpensive Eprom eraser for the hacker. It erases two chips per exposure, so if you are one of those smart people who only makes little mistakes and only needs to erase two Eproms at a time, this eraser is for you.

WALLING CO.

4401 S. Juniper • Tempe, AZ 85282
(602) 838-1277
Arizona residents add 6% tax

GREAT DISKETTES Super low prices SYNCOM

The low priced, high quality diskette with a LIFETIME WARRANTY. Packed in polybags of 10 with Tyvek envelopes, labels and reinforced hubs.
One of the best buys we've seen.

$1.39 QTY. 20 ea. 5¼" SSDD — 5¼" DSDD ea. $1.85 QTY. 20

OTHER GREAT VALUES:

DISKETTE 70—Holds 70 5¼" diskettes in dust free safety **$14.95** ea. + $3.00 Shpng.
DISK CADDIES—Flip up style holds 10 5¼" diskettes **$1.65** ea. + .20 Shpng.

$15.00 Bonus Offer!

For the lowest priced, highest quality diskettes, storage cases, printer ribbons and paper products, send for our catalog. FREE with your order; $1.00 otherwise. BONUS! Every catalog includes $15.00 worth of bonus coupons.

Shipping: 5¼" DISKETTES—Add $3.00 per 100 or fraction thereof. OTHER ITEMS: Shipping charges as shown in addition to diskette shipping charges. **Payment:** VISA or MC. COD orders, add $3.00. **Taxes:** Illinois customers, please add 8%.

Nationwide: 1-800-621-6827
In Illinois: 1-312-944-2788
Minimum Order: $35.00
WE WILL BEAT ANY NATIONALLY ADVERTISED PRICE!
DISK WORLD!
Suite 4806 • 30 East Huron Street • Chicago, Illinois 60611

SYNCOM

FLIGHT SIMULATOR GAMES

COCKPIT 64

NEW For the Commodore 64

- 100% Machine Language
- Windshield View
- 7 Airports

Tape **$29.95** Disk **$31.95**

	Tape	Disk
Runway (C64•16KVIC-20•Adam)	$24.95	$26.95
Sky Pilot (8K VIC-20•Atari)	$17.95	19.95
Flight Sim. II (C64•Apple•IBM)	N/A	$47.95
IFR (Commodore 64 only)	$29.95	$29.95
Air Traffic Controller (Apple)	N/A	$19.95

COD ORDER PHONE
WE SHIP WITHIN 48 HOURS
(312) 577-5154

SUSIE SOFTWARE
874A E. N.W. Highway
Mt. Prospect, IL 60056

TEXAS INSTRUMENTS 99/4A

ASSEMBLY LANGUAGE PRIMER

Teaches TI assembly language in step by step fashion for Basic programmers. Explains concepts in detail with many examples. This is what you have been waiting for if you haven't been able to understand the TI Editor/Assembler manual. $20

EDITOR/ASSEMBLER

The Dow E/A turns your TI into an assembly language machine. For use with TI's Mini Memory Module. Fast and convenient. Allows use of entire RAM. Manual includes sample program with detailed explanations. See review in Aug 83 Home Computer Magazine. Cassette. $25.
(Dow E/A and PRIMER $40)

FLIGHT SIMULATOR

Learn to fly with the Dow-4 Gazelle, a realistic IFR simulation of a typical 4-place private plane. It is not a game. A manual with 30 pages of text plus 7 figures helps the novice learn to fly. Experienced pilots will enjoy flying the ILS approach. Response time under 1 sec average. Display shows full panel (10 dials and 11 lights) and indicates position of runway for landing. Realistic sound effects. See reviews in Jan 83 Home Computer Magazine and Jun 83 AOPA Pilot. Requires joystick. Cassette. $30.

For additional information, write or call 412-521-9385. To order, send check or MO U.S. funds:
JOHN T. DOW
6560 Rosemoor Street
Pittsburgh, Pa. 15217
Postage to U.S. and Canada included. (If foreign, add U.S. $2.)
Pa. residents add 6%

Aspen Ribbons® brand replacements for Texas Instruments® 850/855 Ribbons

Buy directly from a manufacturer and save! These ribbons are manufactured by Aspen Ribbons, Inc., as replacement ribbons for use on printers manufactured by other companies.* Standard ink color is black. Red, green, blue, brown, and purple colors are available for $2.00 extra per ribbon.

CALL FOR FREE CATALOG

PRICES (BLACK)

T.I.® 850/855 $4.75 to $9.75 ea.
Price depends on quantity ordered.

*Aspen Ribbons, Inc., is not affiliated with or licensed by any other company.

Aspen Ribbons, Inc. • 1700 N. 55th St.
Boulder, CO 80301-2796 • (303)444-4054
Telex: 45-0055 • End User: 800-525-0646
Wholesale: 800-525-9966

DISKETTES

$1.35 100% Certified

Buy 100 Diskettes GET 10 FREE

DISKMINDER™ Capacity 75 $17.95

1-800-322-DATA

HOME COMPUTER DESK PLANS

PROTECT YOUR INVESTMENT!

Designed by home computer user. All the room you need for computer monitor, printer, peripherals, etc. Shelves for software, everything at your fingertips. Fits COMMODORE, ATARI, APPLE I & II, IBM-PC, TRS 80. Bottom shelf slotted for printer paper plus storage. 28" deep x 51¾" high x 71¾" length. **Quality Plans, Instructions.**

PLANS - $10.00

CARPENTER'S CREATIVE DESIGNS
P.O.Box 122 / Desert Center, CA 92239

SAVE ON MEMOREX DISKETTES

$1.80 ea. 5¼" SSDD Qty. 20 — $2.35 ea. 5¼" DSDD Qty. 20

5¼" SSDD-96TPI **$2.49** ea. 5¼" DSDD-96TPI **$3.25** ea.

Boxed in 10's with Tyvec sleeves, reinforced hubs and labels.

DISKETTES FOR MACINTOSH & HP-150 AT TERRIFIC SAVINGS!

3½" Memorex **$3.99** ea.

See our big ad in this issue for other great values!

Shipping: 5¼" or 3½" DISKETTES—Add $3.00 per 100 or fewer diskettes. **Payment:** VISA and Mastercard accepted. COD orders only, add $3.00 handling charge. **Taxes:** Illinois residents, please add 8% sales tax.

WE WILL BEAT ANY NATIONALLY ADVERTISED PRICE ON THE SAME PRODUCTS AND QUANTITIES!

Nationwide: 1-800-621-6827
Illinois: 1-312-944-2788
Hours: 9AM - 5PM Central Time
Minimum Order: $35.00

DISK WORLD!, Inc.
Suite 4806 • 30 East Huron Street • Chicago, Illinois 60611

MEMOREX

Lyco Computer Marketing & Consultants

TO ORDER CALL US TOLL FREE **800-233-8760**
In PA 1-717-327-1824

DISK DRIVES

RANA
1000 ... $ CALL

PERCOM
ATARI AT88-PD $329.00
AT-88 DOUBLER $129.00
ADD-ON DRIVES .... $ CALL

HARD	FOR	5 MEG
DISK	APPLE	10 MEG
DRIVES	IBM-PC	15 MEG
AVAILABLE		20 MEG

TRAK DISK DRIVES
AT-D1 .... $379.00
AT-D2 .... $399.00
AT-D2 TURBO PAK $22.95
PRINTER CABLE $22.95

CONCORD DISK DRIVES
APPLE 163K DRIVE $199.00
APPLE 326K DRIVE $229.00
APPLE CONTROLLER CARD $69.00

COMMODORE VIC 174K ... $289.00
COMMODORE VIC 348K ... $359.00

ATARI 176K MASTER ... $289.00
ATARI 348K MASTER ... $369.00
ATARI ADD-ON DRIVES ... $CALL

DISKETTES

ELEPHANT
SINGLE SIDE SD (10) $17.75
SD (100) $16.75/10
SINGLE SIDE DD (10) $21.75
DD (100) $20.75/10
DOUBLE SIDE DD (10) ... $26.75
DD (100) $24.75/10

MAXELL
MD1 (10) $27.75
MD2 (10) $37.75

CERTRON CASSETTES
CC-10 (12) $15.99
CC-20 (12) $17.99

COMPUTER CARE

BIB
DISK DRIVE CLEANER ... $12.75
COMPUTER CARE KIT ... $19.75

NORTRONICS
DISK DRIVE CLEANER WITH SOFTWARE FOR IBM-PC, ATARI, VIC, APPLE, & TI $29.75
DISK CLEANER REFILLS ... $14.75
CASS DRIVE CLEANER $9.95
MEDIA BULK ERASER ... $46.75

commodore
VIC 64 .... $ CALL
VIC 1541 .... $239.00

SSI
KNIGHTS OF DESERT ... $26.75
EAGLES $26.75
TIGERS IN SNOW $26.75
COMBAT LEADER $26.75
BATTLE FOR N $26.75

EPYX 64
JUMPMAN C/D $27.75
JUMPMAN JR R $27.75
PITSTOP R $27.75
TEMPLE ASPHAI $27.75
GATE T ASPHAI $27.75
CRUSH. C&C C/D $21.75

INFOCOM 64
ENCHANTER $34.75
PLANETFALL $34.75

DATASOFT
POOYAN C/D $21.75
O'RILEYS MINE $21.75

PARKER 20
FROGGER $33.75

PROFESSIONAL SOFTWARE
WORD PRO 3 .... $59.75
SPELL RIGHT $39.75
BOTH TOGETHER $79.75

EASTERN HOUSE
RABBIT 20 ROM $34.75
RABBIT 64 ROM $34.75

INNOVATIVE CONCEPTS
DISK STORAGE (10) $4.95
DISK STORAGE (15) $9.95
DISK STORAGE (25) $19.95
DISK STORAGE (50) $26.75
ROM STORAGE (10) $24.75

MODEMS
ANCHOR MARK I $45.75
ANCHOR MARK II $79.75
HAYES SMART $239.00
HAYES MICRO II $309.00
MICROBIT 1000C $129.00
CAT $144.00
D-CAT $155.00
J-CAT $115.00
APPLE CAT II $279.00
212 APPLE CAT $589.00

MONITORS
SAKATA COLOR $249.00
TAXAN GREEN $119.00
TAXAN AMBER $129.00
TAXAN RGB COMPOSITE $289.00
ZENITH AMBER $105.00
ZENITH GREEN $95.00
GORILLA GREEN $88.00
GORILLA AMBER $95.00
NEC JB1260 .... $99.00
NEC JB1205 .... $145.00
NEC JB1215 COLOR ... $269.00
AMDEK GREEN $145.00
AMDEK AMBER $149.00
AMDEK COLOR 1 ...... $289.00

apple

BRODERBUND
BANK STREET D $44.75
CHOPLIFTER D $24.95
DAVID'D MAGIC $24.95
DROL D $24.95
LODE RUN D $24.95
SEAFOX D $22.95
SPARE CHANGE D $24.95
AE D $24.75

KOALA
KOALA TABLET $84.75
PROGRAMMERS GUIDE ... $12.75
GEOMETRIC DESIGNS ... $22.75
LOGO DESIGN $27.75
ILLUSTRATOR $99.75
SONG WRITER D $27.75
MASTER TYPE $27.75

CONTINENTAL
HOME ACCOUNT D $44.75
TAX ADVANTAGE $35.75
BOOK OF APPLE SOFTWARE $16.95
GENERAL LEDGER $179.95
ACCOUNTS PAY $179.95
ACCOUNTS REC $179.95
PAYROLL $179.95
PROPERTY MGMT $329.95

INFOCOM
DEADLINE $34.75
ZORK 1, 2, or 3 .... $34.75
ENCHANTER $34.75
PLANETFALL $34.75

SPINNAKER
KINDERCOMP D $21.75
STORY MACH D $23.75
FACE MAKER D $23.75
SNOOPER TR D $29.75
SNOOPER T2 D $29.75
DELTA DRAW D $32.75
FRACTION F D $23.75
ALPHABET ZOO D $21.75
MOST AMAZING D $26.75
RHYMES & RID D $21.75

APPLE DUMPLING GX $99.75
APPLE DUMPLING 16K BUFFER $179.75
TEXT PRINTER INTERFACE $79.75

SSI
KNIGHTS OF DESERT ... $26.75
EAGLES $26.75
TIGERS IN SNOW $26.75
GERMANY 1985 .... $36.75
BATTLE FOR NORMANDY $26.75
SHATTERED ALLIANCE ... $39.75

SIERRA ON-LINE
ULTIMA II $39.75
DARK CRYSTAL $27.75

ATARISOFT
PACMAN $25.75
DONKEY KONG $25.75
DIG DUG $25.75
CENTIPEDE $25.75
ROBOTRON $25.75

PROGRAM DESIGN
ANALOGIES $18.50
PREP FOR SAT'S $79.75
PRESCHOOL IQ BUILDER $18.75
READING COMP $18.75
VOCABULARY BUILDER ... $18.75

PEACHTREE SOFTWARE
LIST MANAGER $199.00
BUSINESS GRAPHICS $225.00
GRAPHICS LANGUAGE ... $299.00

REQUIRES COBAL RUNTIME
ACCTS PAYABLE I $495.00
ACCTS PAYABLE II $1695.00
PAYROLL I $495.00
PAYROLL II $1695.00
GENERAL LEDGER I $495.00
GENERAL LEDGER II ... $1695.00

CP/M VERSION
PEACHPACK 4 .... $295.00
ACCTS PAYABLE III $549.00
ACCTS PAYABLE IV $899.00
ACCTS REC III $549.00
ACCTS REC IV $899.00
GENERAL LEDGER III ... $549.00
GENERAL LEDGER IV ... $899.00
INVENTORY MGMT I $549.00
INVENTORY MGMT II ... $899.00
PAYROLL III $549.00
PAYROLL IV $899.00
SALES INVOICING I $549.00
SALES INVOICING $899.00

ATARI

INFOCOM
DEADLINE $34.75
ZORK 1, 2, or 3 .... $26.75
ENCHANTER $34.75

EDUFUN
GULP ARROW GRAPHICS $24.75
FACE FLASH $24.75

CONTINENTAL
HOME ACCOUNT D $44.75
TAX ADVANTAGE $35.75
BOOK OF ATARI SOFTWARE $16.95

SCARBOROUGH
SONG WRITER D $27.75
MASTER TYPE $27.75

EASTERN HOUSE
MONKEY WRENCH II $52.75

DON'T ASK SOFTWARE
SAM $39.75
ABUSE $22.75
WORD RACE $24.75

PROGRAM DESIGN
ANALOGIES $14.50
PREP FOR SAT'S $79.75
PRESCHOOL IQ BUILDER $16.75
READING COMP $16.75
VOCABULARY BUILDER ... $16.75
LET'S SPELL $14.75

MasterCard — TO ORDER — VISA

CALL TOLL FREE
800-233-8760
In PA 1-717-327-1824

or send order to
Lyco Computer
P.O. Box 5088
Jersey Shore, PA 17740

Lyco Computer Marketing & Consultants

TO ORDER CALL US **TOLL FREE 800-233-8760**
In PA 1 717-327-1824

SAVE ON THESE IN-STOCK PRINTERS

MANNESMANN TALLY
SPIRIT 80 .... $299.00
MTL-160L $559.00
MTL-180L $775.00

NEC
NEC8023 .... $369.00
NEC8025 .... $699.00

PRINTER INTERFACING
AVAILABLE FOR COMMODORE VIC, APPLE, ATARI, IBM-PC, TRS-80, TI, AND OTHERS

STAR MICRONTICS
GEMINI 10X $259.00
GEMINI 15X $379.00
DELTA 10 .... $449.00
DELTA 15 .... $525.00
RADIX 10 .... $575.00
RADIX 15 .... $675.00
POWERTYPE .... $CALL
SWEET P (Model 100) ... $549.00
STX80 .... $149.00

SMITH CORONA TP2 ... $449.00

CITOH
GORILLA GX-100 .... $169.00
GORILLA SERIAL $199.00
PROWRITER 8510 ...... $335.00
PROWRITER II $575.00
8600 .... $899.00
STARWRITER $999.00
PRINTMASTER $1299.00
SHEET FEEDER $425.00
620 .... $929.00
630 .... $1699.00
8510 SP $499.00
8510 SCP $559.00
A10 LETTER QUAL $499.00

EPSON
RX-80 .... $CALL
RX-80 FT .... $FOR
FX-80 .... $CURRENT
FX-100 .... $PRICES

OKIDATA
80 .... $SAVE
82A .... $ON THESE
83A .... $IN-STOCK
84 .... $PRINTERS
92 .... $CALL &
93 .... $SAVES

PRINTING PAPER
3000 SHEETS FANFOLD $42.75
1000 SHEETS FANFOLD $19.75
1000 SHEETS LETTER ... $21.95
200 SHEETS LETTER $8.99
150 RAG STATIONERY ... $10.99
MAILING LABELS (1 in) ... $9.75
14x11 1000 FANFOLD ... $24.75

WICO COMMODORE & ATARI
JOYSTICK $16.75
RED BALL $18.75
BOSS JOYSTICK $12.75
TRACK BALL $32.75
12 FT EXT CORD $6.95
TI ADAPTER $9.95
APPLE ANALOG $34.75
IBM-PC ANALOG $34.75

DUST COVERS
800 .... $3.99
400 .... $3.99
1200 .... $3.99
410 .... $3.99
810 .... $3.99
600 XL $5.99
800XL $5.99
1050 .... $5.99
1010 .... $5.99
PROWRITER $5.99
GEMINI 10X $5.99
PERCOM $5.99
EPSON $5.99
RANA $5.99
VIC 64/20 .... $5.99
VIC 1541 .... $5.99

RESTON 64
MINER 2049 ROM . $29.75

TIMEWORKS
INVENTORY $39.75
ACCOUNTS REC ... $39.75
ACCOUNTS PAY ... $39.75
GENERAL LEDGER . $39.75
PAYROLL $39.75
INVENTORY MAN .. $39.75
CASH FLOW $39.75
SALES ANALYSIS .. $39.75
DATA MANAGER ... $19.75
MONEY MGR C/D ... $19.95
CHECKBOOK C/D .. $19.95
FORGET-ME-NOT .. $19.95
STAR BATTLE $19.95
ROBOT REVOLT $19.95
SPELLBOUND C/D . $19.95

SCARBOROUGH 64
SONG WRITER D ... $27.75
PHI BETA FILER $32.75
MASTER TYPE $27.75

HES 64
TURTLE TUTOR $22.75
TURTLE TRAINER .. $22.75
TURTLE GRAPHICS $37.75
SOUND BOX $9.95
64 FORTH $39.95
HESMON $22.75
GRIDRUNNER $19.75
ATTACK OF M C $22.75
HESWRITER $28.75
OMNI WRITER $45.75
TYPE N WRITER $24.75
PAINT BRUSH $22.75
BENJI $25.75
HOME MANAGER .. $28.75
TIME MONEY MGR . $44.75
OMNI CALC $33.75
SWORD POINT $19.95
HES MODEM $49.95
M. MULTIPLAN $65.75

CONTINENTAL 64
HOME ACCOUNT D $44.75
TAX ADVANTAGE $35.75
F.C.M. $35.75
64 USERS OF ENCYC $12.75
20 ENCYCLOPEDIA $12.75

KOALA 64
KOALA TABLET $69.75
PROGRAMMERS GUIDE ... $12.75
GEOMETRIC DESIGNS ... $22.75
LOGO DESIGN $27.75
SPIDER EATER $22.75

SYNAPSE 64
BLUE MAX C/D $24.75
ZEPPELIN C/D $24.75
PHAROAH'S C/D $24.75
SHAMUS C/D $24.75

PARKER 64
FROGGER $32.75
QBERT $32.75
TUTANKHAM $32.75

SPINNAKER 64
KINDERCOMP D/R $21.75
STORY MACH ROM $24.75
FACE MAKER D/R $24.75
SNOOPER TR D $26.75
SNOOPER T2 D $26.75
DELTA DRAW ROM $26.75
FRACTION F ROM $24.75
KIDS ON KEYS $24.75
ALPHABET ZOO $24.75
MOST AMAZING R $26.75

commodore

CARDCO
G PRINTER INTERFACE $64.75
B PRINTER INTERFACE $39.75
KEY PAD 64 .... $29.75
LIGHT PEN $29.75
3 SLOT EXPANSION $24.75
5 SLOT EXPANSION $54.75
6 SLOT EXPANSION $79.75
PRINTER UTILITY $19.75
CASSETTE INTERFACE ... $29.75
WRITE NOW 20 .... $29.75
WRITE NOW 64 .... $39.75
MAIL NOW $29.75
DM-1 PRINTER $109.75
LQ-1 PRINTER $499.75

BRODERBUND 64
BANK STREET D $44.75
CHOPLIFTER R $29.95
DAVID'D MAGIC $24.95
DROL D/R $24.95
LODE RUN D/R $24.95
SEAFOX R $29.95
SPARE CHANGE D $24.95

ADVENTURE INTERNATIONAL
ADVENTURE 1-12 ea . $17.75
SAGA $27.75
PREPPIE $24.75
DISKEY $34.75
SEA DRAGON $24.75

KOALA TABLET $69.75
PROGRAMMERSG ... $12.75
GEOMETRIC DESIGN . $22.75
LOGO DESIGN $27.75
ILLUSTRATOR $99.75

INHOME
400 KEYBOARD .. $IN-STOCK

BIT 3
80 COLUMN SCREEN BOARD $259.75

PARKER
FROGGER $32.75
QBERT $32.75

SPINNAKER
KINDERCOMP D/R .. $20.75
STORY MACH D/R ... $26.75

PERCOM
AT88S1 .. $249.00
RDF40SI $269.00

SSI
KNIGHTS OF DESERT $26.75
EAGLES $26.75
TIGERS IN SNOW $26.75
COMBAT LEADER $26.75

DATASOFT
SPELL WIZZARD $34.75
TEXT WIZZARD $34.75
WORD WIZZARD $46.75
ZAXXON C/D $26.75

GEMINI 10 .... $259.00

BRODERBUND
BANK STREET D ... $44.75
CHOPLIFTER D $32.95
DAVID'D MAGIC $24.95
DROL D $24.95
LODE RUN D $24.95
SEAFOX D $22.95
SPARE CHANGE D . $24.95
AE D $24.75

EPYX
JUMPMAN C/D $27.75
JUMPMAN JR R $27.75
PITSTOP R $27.75
TEMPLE ASPHAI ... $27.75
GATE T ASPHAI $27.75
CRUSH. C&C C/D ... $21.75

LJK
LETTER PERFECT .. $69.75
DATA PERFECT $69.75
SPELL PERFECT ... $69.75

ATARI Computers for people.
PACMAN $29.75
DONKEY KONG $29.75
DIG DUG $29.75
CENTIPEDE ... $29.75
STAR RAIDERS $32.75
ROBOTRON $32.75
PAINT $32.75
JOUST $35.75
LOGO $69.75
ATARI WRITER $79.75
BOOK KEEPER $104.75
VISICALC $159.75
HOME FILING MGR $35.75
PILOT HOME $54.75
KEYPAD $95.75

800XL .... $ CALL
1050 DISK DRIVE $ CALL
1050 RECORDER $74.00
1027 PRINTER $ CALL
1025 PRINTER IN-STOCK

MasterCard

TO ORDER

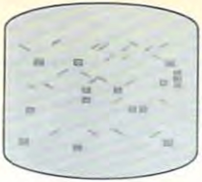
VISA

CALL TOLL FREE
800-233-8760
Customer Service 1-717-327-1825

or send order to
Lyco Computer
P O Box 5088
Jersey Shore PA 1774C

POLICY
In-stock items shipped within 24 hours of order. Personal checks require four weeks clearance before shipping. No deposit on C.O.D. orders. Free shipping on prepaid cash orders within the continental U.S. PA residents add sales tax. All products subject to availability and price change. Advertised prices show 4% discount offered for cash, add 4% for Master Card or Visa. DEALER INQUIRIES INVITED

Advertisers Index



Save UP TO 40% on COMPUTE!

Every issue of COMPUTE! contains up to 30 new programs and games. And a year's subscription brings them to you for less than 15 cents each! Plus you'll enjoy the most useful home computer advice, ideas and information anywhere! Subscribe now at up to 40% off the newsstand price. At less than 15 cents per program, this COMPUTE! offer is too good to pass up!

☐ 1 year $24—Save 32%! ☐ 2 years $45—Save 36%!
☐ 3 years $65—Save 40%!

Name ______________________

Address ______________________

City __________ **State** __________ **Zip** __________

☐ Payment enclosed ☐ Bill me

Charge my ☐ Visa ☐ MasterCard ☐ American Express

______________________ ___/___

Account No. **Exp. date**

COMPUTE! brings you programs and games for the following machines: Atari, PET/CBM, VIC-20, TI 99/4A, Apple, Commodore 64, Radio Shack Color Computer, IBM PC and IBM PCjr.

348BO1

Save UP TO 40% on COMPUTE!

Every issue of COMPUTE! contains up to 30 new programs and games. And a year's subscription brings them to you for less than 15 cents each! Plus you'll enjoy the most useful home computer advice, ideas and information anywhere! Subscribe now at up to 40% off the newsstand price. At less than 15 cents per program, this COMPUTE! offer is too good to pass up!

☐ 1 year $24—Save 32%! ☐ 2 years $45—Save 36%!
☐ 3 years $65—Save 40%!

Name ______________________

Address ______________________

City __________ **State** __________ **Zip** __________

☐ Payment enclosed ☐ Bill me

Charge my ☐ Visa ☐ MasterCard ☐ American Express

______________________ ___/___

Account No. **Exp. date**

COMPUTE! brings you programs and games for the following machines: Atari, PET/CBM, VIC-20, TI 99/4A, Apple, Commodore 64, Radio Shack Color Computer, IBM PC and IBM PCjr.

348BO1

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 2312 GREENSBORO, NC

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE!
P.O. Box 914
Farmingdale, NY 11737

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO. 2312 GREENSBORO, NC

POSTAGE WILL BE PAID BY ADDRESSEE

COMPUTE!
P.O. Box 914
Farmingdale, NY 11737

The Editor's Feedback:

Computer: ☐ PET ☐ Apple ☐ Atari ☐ VIC-20 ☐ Commodore 64
☐ TI-99/4A ☐ Timex/Sinclair ☐ Radio Shack Color Computer
☐ Other ________________ ☐ Don't yet have one ...

Are you a COMPUTE! Subscriber? ☐ Yes ☐ No I would like to see:

More	Just Right	Fewer	
☐	☐	☐	Specific applications programs.
☐	☐	☐	BASIC programs.
☐	☐	☐	Machine language programs.
☐	☐	☐	Tutorials.
☐	☐	☐	Educational articles.
☐	☐	☐	Detailed explanations of programs.

More	Just Right	Fewer	
☐	☐	☐	Games.
☐	☐	☐	Reviews of game software.
☐	☐	☐	Reviews of business software.
☐	☐	☐	Reviews of educational software.
☐	☐	☐	Reviews of hardware.

What do you like best about COMPUTE!?

What do you like least?

89101112C

Place
Postage
Here

COMPUTE! Magazine
P.O. Box 5406
Greensboro, NC 27435-0406

COMPUTE!'s
FREE Reader Information Service

Use these cards to request FREE information about the products advertised in this issue. Clearly print or type your full name and address. Only one card should be used per person. Circle the numbers that correspond to the key number appearing in the advertisers index.

Send in the card and the advertisers will receive your inquiry. Although every effort is made to insure that only advertisers wishing to provide product information have reader service numbers, COMPUTE! cannot be responsible if advertisers do not provide literature to readers.

Please use these cards *only* for subscribing or for requesting product information. Editorial and customer service inquiries should be addressed to: COMPUTE!, P.O. Box 5406, Greensboro, NC 27403. Check the expiration date on the card to insure proper handling.

Use these cards and this address only for COMPUTE!'s Reader Information Service. Do not send with payment in any form.

COMPUTE!

101	102	103	104	105	106	107	108	109	110	111
112	113	114	115	116	117	118	119	120	121	122
123	124	125	126	127	128	129	130	131	132	133
134	135	136	137	138	139	140	141	142	143	144
145	146	147	148	149	150	151	152	153	154	155
156	157	158	159	160	161	162	163	164	165	166
167	168	169	170	171	172	173	174	175	176	177
178	179	180	181	182	183	184	185	186	187	188
189	190	191	192	193	194	195	196	197	198	199
200	201	202	203	204	205	206	207	208	209	210
211	212	213	214	215	216	217	218	219	220	221
222	223	224	225	226	227	228	229	230	231	232
233	234	235	236	237	238	239	240	241	242	243
244	245	246	247	248	249	250	251	252	253	254
255	256	257	258	259	260	261	262	263	264	265
266	267	268	269	270	271	272	273	274	275	276
277	278	279	280	281	282	283	284	285	286	287
288	289	290	291	292	293	294	295	296	297	298
299	300	301	302	303	304	305	306	307	308	309
310	311	312	313	314	315	316	317	318	319	320
321	322	323	324	325	326	327	328	329	330	331
332	333	334	335	336	337	338	339	340	341	342
343	344	345	346	347	348	349	350			

Circle 101 for a one year new U.S. subscription to COMPUTE!: you will be billed for $24.

Please print or type your full name and address. Limit one card per person.

Name

Address

City

State/Province Zip

Country

Please include zip code. Expiration 10/31/84. CO884

Place
Postage
Here

COMPUTE! Reader Service
P.O. Box 2141
Radnor, PA 19089

CARDCO "NOW" SOFTWARE

. . . available now for your Commodore-64™ and more!

A fine line of software developed by CARDCO for your Commodore-64 computer with all the features you should expect in much more costly software. CARDCO's "NOW" Series provides many unique and exclusive features and are packaged for easy reference, simple storage, instant recognition.

"WRITE NOW" . . . WORD PROCESSOR SOFTWARE . . . An excellent time saver, CARDCO offers the "Write Now" C/ Ø2 word processor program with built-in 80 column display. You see exactly what will print. All special codes can be transmitted to printers maintaining justification. Easy full-screen editing; works with any printer.

"MAIL NOW" . . . MAILING LIST SOFTWARE . . . CARDCO's D/ Ø1 "Mail Now" quickly (in memory) sorts by zip, category, name and state; fully compatible with "Write Now". Other fine features include: user-oriented; menu-driven operation; each disk supports 600 entries. Format can print single, double or triple labels across.

"SPELL NOW" . . . Cardware D/ Ø4 . . . a fine program designed as a spell checker for use with "Write Now" on the Commodore-64. A 34,000 word dictionary with two additional user constructed dictionaries. Menu-driven operation for ease of use. And "Spell Now" allows you to see each misspelled word in the context of your document for correction.

"FILE NOW" . . . D/ Ø5 . . . is a totally integrated, menu-driven database software package which interfaces with both the "Write Now!" for the 64 and the "Spell Now." 40K of working storage space is available with "File Now". "File Now" appears on the screen as index cards for easier manipulation of your data base; you see 5 index cards at a time. Cards are user defineable, i.e., user determines what goes where on the "index cards" and can sort by any given field. Every card has a general topic field which allows for quick sorting through cards.

"GRAPH NOW" INCLUDING . . . "PAINT NOW" . . . D/ Ø6 . . . This disk-based graphic/logo generator is totally menu-driven. Allows for the development of pies, charts, bar graphs and other vivid graphic illustrations. Also has the ability to design, and print logos and high resolution pictures. "Commodore-ready"; interfaces with CARDCO'S "Write Now" Word Processor, "Mail Now", "Spell Now" and "File Now".

Write for illustrated literature and prices or see CARDCO Computer Accessories and Software wherever Computers are sold.

cardco, inc.

300 S. Topeka Wichita, Kansas 67202 (316) 267-6525

"The world's largest manufacturer of Commodore accessories."

Commodore™ is a registered trademark of Commodore Business Systems, Inc.

Commodore 64 Magic Desk I

Only From Commodore—The Excitement and Simplicity of Magic Desk!

Only Commodore brings you the magic of MAGIC DESK . . . the next generation of "user friendly" software! Imagine using your computer to type, file and edit personal letters and papers *without learning any special commands!* All MAGIC DESK commands are PICTURES. Just move the animated hand to the picture of the feature you want to use (like the TYPEWRITER) and you're ready to go.

The MAGIC DESK Typewriter works just like a real ELECTRIC TYPEWRITER . . . and it's COMPUTERIZED. All the filing is *electronic*. Excellent sound effects and screen animation make typing fun, whether you're typing letters, reports or memos . . . and the built-in filing feature makes MAGIC DESK useful for keeping names and addresses, home inventory lists, insurance information and more.

Your COMMODORE 64, COMMODORE DISK DRIVE and MAGIC DESK are an unbeatable combination. Filing operations are automatically linked to your Commodore disk drive—but you don't have to know any commands—just "file" the pages you type in the file cabinet and your text is automatically saved on diskette. There are 3 file drawers with 10 file folders in each drawer and 10 pages in each folder.

To PRINT a page you've typed, just "point" at the picture of the printer and your pages are automatically printed on your COMMODORE PRINTER or PRINTER/PLOTTER. If you want to erase what you've typed, the WASTE-BASKET under the desk lets you "throw away" pages. There's even a DIGITAL CLOCK which helps you keep track of time while you're typing.

Not only is MAGIC DESK easy to use . . . it's hard to make a mistake! Just press the COMMODORE key and one of several "help menus" appears to tell you exactly what to do next. Special messages show you how the various picture commands work and help you when you make a mistake. Help messages also show you how to use the printer, filing cabinet, digital clock and wastebasket.

Another reason why Commodore is number one in the world of microcomputers—Come join us.

commodore COMPUTERS

First In Quality Software