Now Monthly!
NO. 15 JANUARY $3.00
THE MAGAZINE FOR ATARI® COMPUTER OWNERS™

A.N.A.L.O.G COMPUTING

INSIDE THE NEW 600XL

PIRACY SURVEY

KNIGHTS & CHALICES

MUSIC SYNTHESIZER

BRICKLAYER'S NIGHTMARE

ALTERNATIVE KEYBOARD HANDLER

THE SECRETS OF PERFECT MEMORY: ONE AND ONE HALF EARTH DOLLARS

AT LAST: THE WHOLE TRUTH ABOUT FLOPPIES.

Amazing book reveals all!

How to keep from brainwashing your disk so it never loses it's memory.

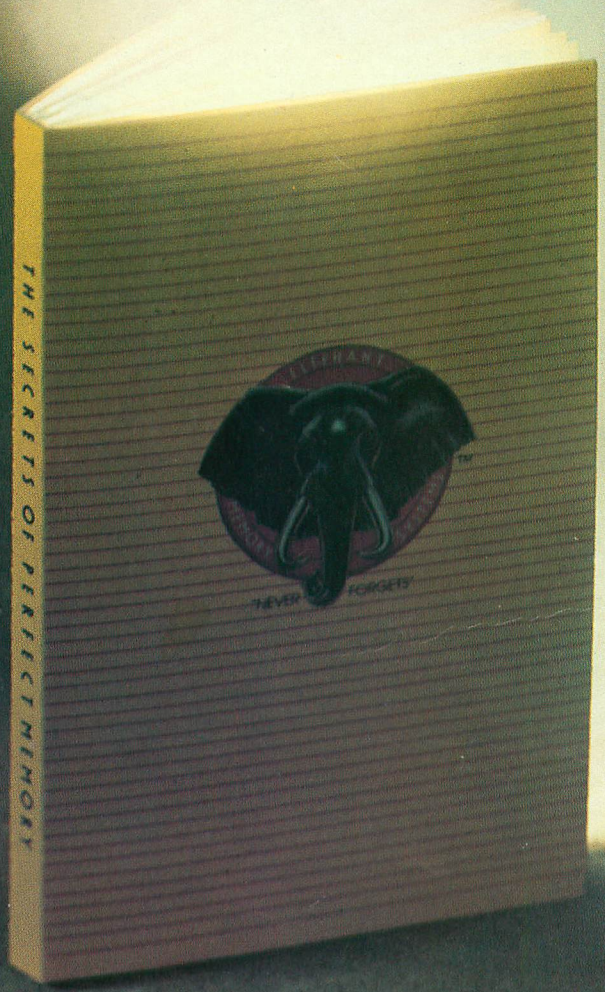
How fingerprints can actually damage disks. Unretouched Kirlian photographs of UFO's (Unidentified Floppy Objects)! The incredible importance of making copies: the Department of Redundancy Department—and what goes on when it goes on! Powerful secret methods that scientists claim can actually prevent computer amnesia! All this, and much more . . .

In short, it's an 80-page plain-English, graphically stunning, pocket-sized definitive guide to the care and feeding of flexible disks.

For The Book, ask your nearest computer store that sells Elephant™ disks, and bring along one and one half earth dollars.

For the name of the store, ask us.

Elephant Memory Systems®
Marketed exclusively by Dennison Computer Supplies, Inc., 55 Providence Highway, Norwood, MA 02062. Call toll free 1-800-343-8413. In Massachusetts, call collect (617) 769-8150. Telex 951-624.

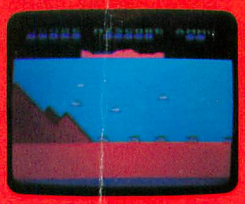


OUT OF THIS WORLD ENTERTAINMENT FOR A DOWN TO EARTH PRICE.

Datasoft® has added a new dimension to computer game enjoyment. Each of Gentry's™ 12 software products are available at prices that are sure to take the gamble out of the game. From $14.95 to $19.95, each selection incorporates the same quality and playability of far more expensive software. See your nearest dealer and find out how Gentry really does give you more than you bargained for.

ROSEN'S BRIGADE™

Face an array of war planes and choppers that come at you in wave after wave. If you can face up to the rigors of battlefield action, you'll face a hero's welcome at home. Penetrate behind enemy lines and rescue your crew. Not only must you fight off the steady onslaught of aircraft, but be crafty enough to pinpoint enemy tanks and battleships below. Your mission. To beat the clock and save your troops.



Available for Atari® Computer Systems

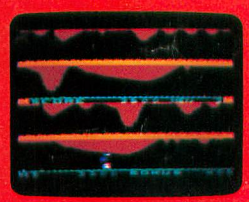
ALIEN MUNCHIES™

This is the space age cookout that cooks with excitement. It takes quick decisions and fancy footwork to catch these little morsels in the barbecue grill. They are downright dangerous on the ground. But let them drink your supply of propane and the tables are suddenly turned. Try Alien Munchies. It's the latest protein diet craze, but be careful, they're habit forming.

Available for Apple® II, II+, IIe

MANIAC MINER™

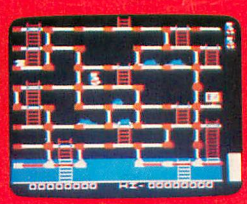
Jumping, turning and hovering in mid-air are your only defenses as you dodge dangerous bats and spiders in search of valuable jewels and treasures. And if you're lucky, defying gravity will be your only chance at defying death. The Magic Pole suddenly appears and takes you deeper and deeper into the mine. The game play is magical, the sounds, maniacal. Maniac Miner will make you crazy for more.



Available for Atari® Computer Systems

HANDY DANDY™

A game of many levels with four completely different screens. As the fearless fix-it man, you find yourself busy at work in a multi-level boiler room. Wild rats, leaking hot steam, falling debris, pounding pile drivers and disappearing floors will earn you an honest day's work. Race to the top of each room before the clock runs out. But watch out! As handy as you are with tools, you could find yourself in a real fix. You can't swim and the water level is rising.

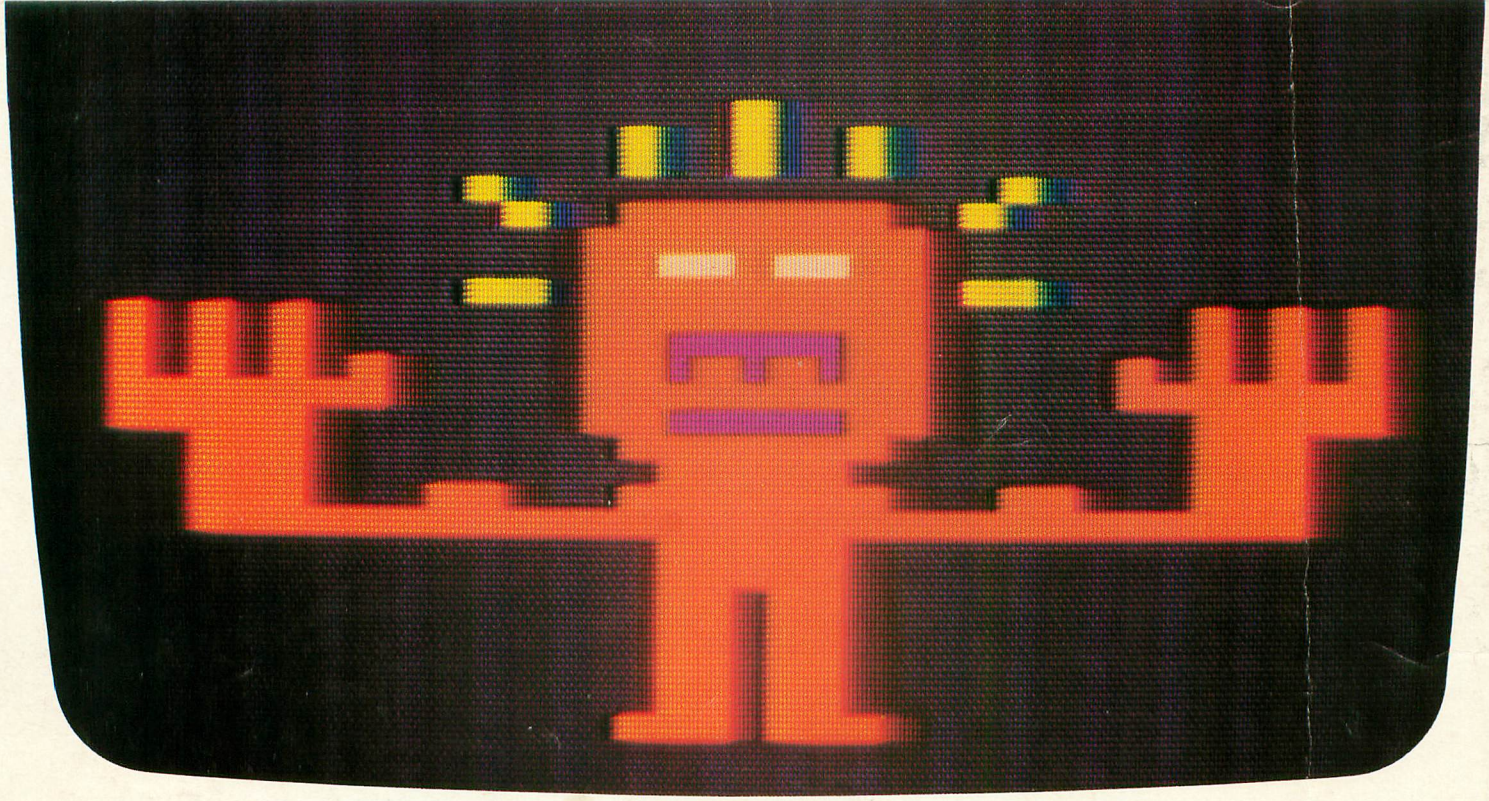


Available for Apple® II, II+, IIe

You'll Find The Best in Home Computer Software Entertainment, Home Management, Graphics, Education from

Datasoft®

9421 Winnetka Avenue, Chatsworth, CA 91311 (213) 701-5161



WOULD YOU SHELL OUT $1000 TO MATCH WITS WITH THIS?

Meet your match. Meet Infocom games: perhaps the best reason in software for owning a personal computer.

In fact, people have been known to purchase computers and disk drives solely for the purpose of playing our games. And they haven't been disappointed. Because Infocom's prose stimulates your imagination to a degree nothing else in software approaches. Instead of putting funny little creatures on your screen, we put you inside our stories. And we confront you with startlingly realistic environments alive with situations, personalities, and logical puzzles the like of which you won't find elsewhere. The secret? We've found the way to plug our prose right into your imagination, and catapult you into a whole new dimension.

If you think such an extraordinary experience is worth having, you're not alone. Everything we've ever written—ZORK® I, II, and III, DEADLINE,™ STARCROSS,™ SUSPENDED,™ The WITNESS,™ PLANETFALL,™ ENCHANTER,™ and INFIDEL™—has become an instant best-seller. For the simple reason that Infocom offers you something as rare and valuable as anything in software—real entertainment.

At last, you can fritter away your evenings playing a computer game without feeling like you're frittering away your computer investment.

Step up to Infocom. All words. No pictures. The secret reaches of your mind are beckoning. A whole new dimension is in there waiting for you.

(For more information on Infocom games contact: Infocom, Inc., P.O. Box 855, Garden City, NY 11530.)

INFOCOM™

The next dimension.

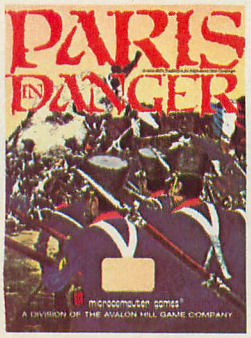
For your: Apple II, Atari, Commodore 64, CP/M 8," DEC Rainbow, DEC RT-11, IBM, MS-DOS 2.0, NEC APC, NEC PC-8000, Osborne, TI Professional, TI 99/4A, TRS-80 Model I, TRS-80 Model III.


From the creators of
Frogger,™ Scramble,™ Super Cobra,™
Tutankham™ and Time Pilot™. . .
KONAMI'S
POOYAN
by DATASOFT®
Datasoft® brings you POOYAN,® the popular arcade game from Konami, for home computers. Enjoy all the fun, fast action, colorful, high-resolution graphics and arcade-style play without all the quarters.
Quick reflexes and fast thinking will protect your helpless piglets in the forest valley from a pack of vicious, hungry wolves who travel up and down the valley clinging to balloons and throwing objects at you. Shoot your arrows to burst their balloons or throw chunks of meat to get them to let go, but don't give up, because the wolves won't!
POOYAN® is a super challenge that will keep any experienced gamer glued to the screen. And Datasoft® will be making it available for Apple,® Atari® 400/800/1200, Radio Shack Color, Commodore 64 and VIC 20 computer systems.
Datasoft Inc.®
HOME COMPUTER SOFTWARE
9421 Winnetka Avenue
Chatsworth, California 91311
(213) 701-5161
©1983 Datasoft® Inc.
Datasoft® is a registered trademark of Datasoft Inc.
POOYAN® is a registered trademark of
Konami® Industries Company Ltd.

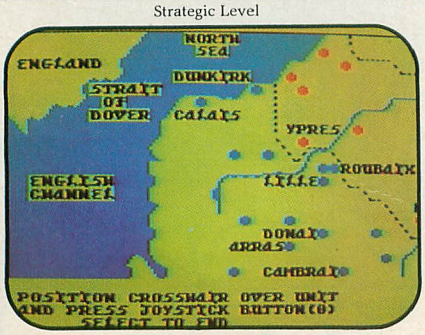
Wargames

Not the movie . . . . the real things!

The Avalon Hill Game Company, America's premiere strategy game maker, has combined their years of experience designing military strategy board games with the latest in artificial intelligence for home computers. The resulting computer games are designed to assist you, the player, with combat results, lines of fire and double hidden movement in two player games and provide a worthy opponent in solitaire games.



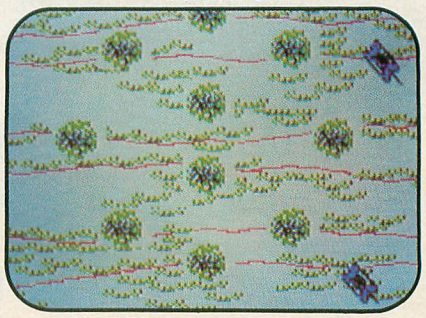
Strategic Level

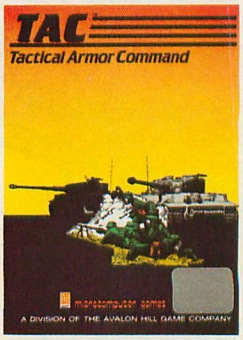


Paris in Danger: A simulation of Napoleon's 1914 campaign in France. One of Napoleon's finest, against the invading Allied Armies. (Austrian, Prussian and Russian). You can choose to take the role as Napoleon, Commander Schwarzenberg, or play both sides to re-create the actual campaign. PARIS IN DANGER is unique, in that it allows the players to compete on both the strategic and tactical levels, on a full-color scrolling map of France and surrounding countries.

For all Atari Home Computers, 48K Disk: $35.00

Tactical Level





T.A.C.: Tactical Armor Command during World War II. You control individual tanks, anti-tank guns, and infantry squads. For one or two players featuring outstanding Hi-Resolution graphics, enhanced sound, and stimulating challenge. Five different scenarios are available from Meeting Engagement, Rear Guard, and Static Defense, to Breakout and Stalemate. The players control up to eight vehicles, guns and squads simultaneously, utilizing the equipment of either the German, British, Russian or American forces.

Atari & Apple Disks (48K): $40.00

LEGIONNAIRE (by Chris Crawford):

Consumer Electronics Showcase Award for Innovative Programming
Wargame of the Year, VIDEO GAMES PLAYER Magazine
Nominee for Wargame of the Year, Game Manufacturers' Association

"On a scale of 1 to 100, this is a 95" SOFTLINE Magazine, March '83.
"Legionnaire is a wonderful game that combines the graphics and movement of arcade games with the depth of strategy games" BYTE, March '83.
"An entertaining, attractive game in which thinking is more important than fast reflexes" COMPUTE!, July, '83.

For all Atari Home Computers, 16K Cassette: $35.00
32K Diskette for Atari Home Computers: $40.00
Apple II Computer Diskette (48K): $40.00

®Trademarks of Warner Communications, Apple Computers Inc. and Tandy Corporation.

Atari 4/800 (40K), TRS-80 (48K) cassettes: $30.00

Apple II (48K)
Atari Home Computers (48K)
TRS-80 I/III (48K)
diskettes: $35.00

CLOSE ASSAULT: Advanced wargame of tactical infantry combat. Russian, German, and American forces are represented in this WWII simulation which blends the allure of computerization with tabletop gaming. **CLOSE ASSAULT** permits original scenario development or pre-programmed ones. Features include double hidden movement, solitaire or two player option, morale factors, and most unique, a game system that actually lets you control squad level units in life-like situations.

AVAILABLE AT LEADING COMPUTER GAME STORES EVERYWHERE
or call Toll-Free: **1 (800) 638-9292**
for the name of the dealer near you.
Ask for Operator A.

AH QUALITY

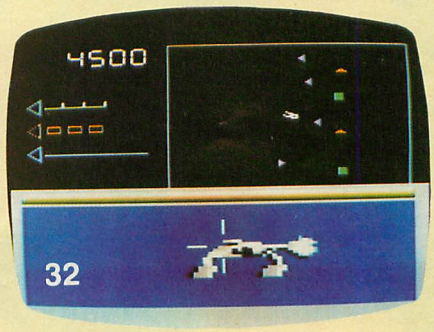
microcomputer games®
A DIVISION OF
The Avalon Hill Game Company
4517 Harford Road • Baltimore, MD 21214 • (301) 254-5300

NO. 15 JANUARY

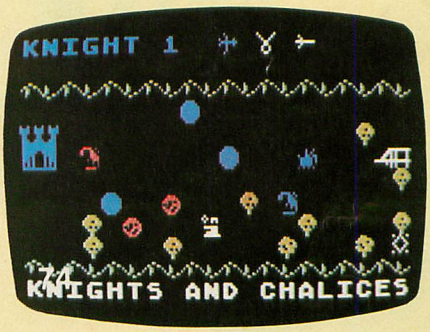
THE MAGAZINE FOR ATARI® COMPUTER OWNERS™

ANALOG COMPUTING

FEATURE ARTICLES

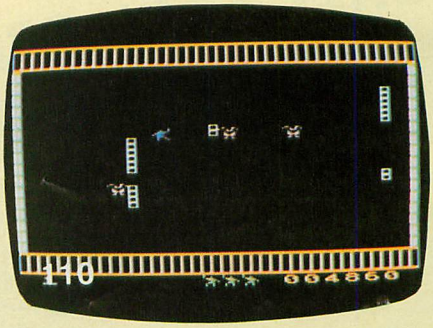


PRODUCT REVIEWS



PROGRAM LISTINGS

COLUMNS



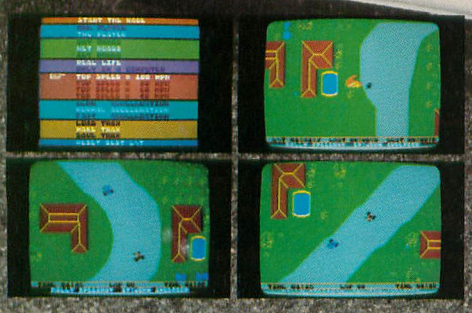
John Anderson's

RALLY SPEEDWAY

SLIP INTO SOMETHING TURBO-CHARGED

like the road-scorching speedster in RALLY SPEEDWAY — the game that turns your ATARI into a full-blown, four-wheeled demon! Take on our demanding course, or use the unique Construct-A-Track option to design your own. Push yourself to the limit for a better lap time, or get a friend in on the action for a one-on-one duel to the finish line — it's your choice!

CALL TOLL FREE 1-800-327-7172



MAKE TRACKS TO YOUR DEALER TODAY!

ATARI 400/800/1200 CARTRIDGE

053-0171 . . . . . $49.95

If your dealer doesn't have Rally Speedway in stock, phone 1-800-327-7172 with your order, or write us at Adventure International, P.O. Box 3435, Longwood, FL 32750 for your FREE catalog.

DEALER ORDERS WELCOME

155

Adventure® INTERNATIONAL

A DIVISION OF SCOTT ADAMS, INC.

ANALOG STAFF

Editors/Publishers
MICHAEL DESCHENES
LEE H. PAPPAS

Managing Editor
JON A. BELL

Technical Editor
BRIAN MORIARTY

Contributing Editors
JOEL GLUCK
BRADEN GRIFFIN, M.D.
TONY MESSINA

Art Director
BOB DESI

Contributing Artist
LINDA RICE

Technical Division
CHARLES BACHAND
TOM HUDSON
KYLE PEACOCK

Advertising Manager
MICHAEL DESCHENES

Distribution
PATRICK J. KELLEY

Typography
HENDRICKX &
LARRIVEE Co., Inc.

Production/Distribution
LORELL PRESS, INC.

Contributors
EDWARD BEVER
CLAUS BUCHOLZ
KEN COLLIER
BOB FINE
AUSTIN FRANKLIN
ALLEN HARBERG
STEPHEN HOWARD
TOM KELLEY
ROBIN E. NOWELL
GORDON ROBSON
JERRY WHITE
BRUCE WILLARD

U.S. POSTAL SERVICE
STATEMENT OF OWNERSHIP, MANAGEMENT AND CIRCULATION
(Required by 39 U.S.C. 3685)

1. TITLE OF PUBLICATION	A. PUBLICATION NO. 6 7 9 3 7 0
ANALOG COMPUTING	2. DATE OF FILING 9/12/83
3. FREQUENCY OF ISSUE	A. NO. OF ISSUES PUBLISHED ANNUALLY 12 (AS OF 1/1/84 12)
BI-MONTHLY (AS OF 1/1/84 MONTHLY)	B. ANNUAL SUBSCRIPTION PRICE $28

4. COMPLETE MAILING ADDRESS OF KNOWN OFFICE OF PUBLICATION (Street, City, County, State and ZIP Code) (Not printers)
565 Main Street Cherry Valley MA 01611

5. COMPLETE MAILING ADDRESS OF THE HEADQUARTERS OR GENERAL BUSINESS OFFICES OF THE PUBLISHERS (Not printers)
SAME

6. FULL NAMES AND COMPLETE MAILING ADDRESS OF PUBLISHER, EDITOR, AND MANAGING EDITOR (This item MUST NOT be blank)

PUBLISHER (Name and Complete Mailing Address)
Michael J Des Chenes 565 Main St Cherry Valley MA 01611

EDITOR (Name and Complete Mailing Address)
Lee Pappas 565 Main St Cherry Valley MA 01611

MANAGING EDITOR (Name and Complete Mailing Address)
Jon Bell 565 Main St Cherry Valley MA 01611

7. OWNER (If owned by a corporation, its name and address must be stated and also immediately thereunder the names and addresses of stockholders owning or holding 1 percent or more of total amount of stock. If not owned by a corporation, the names and addresses of the individual owners must be given. If owned by a partnership or other unincorporated firm, its name and address, as well as that of each individual must be given. If the publication is published by a nonprofit organization, its name and address must be stated.) (Item must be completed)

FULL NAME	COMPLETE MAILING ADDRESS
Michael J Des Chenes	565 Main St Cherry Valley MA 01611
Lee Pappas	" " "

8. KNOWN BONDHOLDERS, MORTGAGEES, AND OTHER SECURITY HOLDERS OWNING OR HOLDING 1 PERCENT OR MORE OF TOTAL AMOUNT OF BONDS, MORTGAGES OR OTHER SECURITIES (If there are none, so state)

FULL NAME	COMPLETE MAILING ADDRESS
NONE	

9. FOR COMPLETION BY NONPROFIT ORGANIZATIONS AUTHORIZED TO MAIL AT SPECIAL RATES (Section 411.3, DMM only)
The purpose, function, and nonprofit status of this organization and the exempt status for Federal income tax purposes: (Check one)

☐ (1) HAS NOT CHANGED DURING PRECEDING 12 MONTHS ☐ (2) HAS CHANGED DURING PRECEDING 12 MONTHS (If changed, publisher must submit explanation of change with this statement.)

10. EXTENT AND NATURE OF CIRCULATION	AVERAGE NO. COPIES EACH ISSUE DURING PRECEDING 12 MONTHS	ACTUAL NO. COPIES OF SINGLE ISSUE PUBLISHED NEAREST TO FILING DATE
A. TOTAL NO. COPIES (Net Press Run)	58000	70000
B. PAID CIRCULATION 1. SALES THROUGH DEALERS AND CARRIERS, STREET VENDORS AND COUNTER SALES	42000	53000
2. MAIL SUBSCRIPTION	9000	13000
C. TOTAL PAID CIRCULATION (Sum of 10B1 and 10B2)	51000	66000
D. FREE DISTRIBUTION BY MAIL, CARRIER OR OTHER MEANS SAMPLES, COMPLIMENTARY, AND OTHER FREE COPIES	430	455
E. TOTAL DISTRIBUTION (Sum of C and D)	51430	66455
F. COPIES NOT DISTRIBUTED 1. OFFICE USE, LEFT OVER, UNACCOUNTED, SPOILED AFTER PRINTING	5000	2000
2. RETURN FROM NEWS AGENTS	1570	1625
G. TOTAL (Sum of E, F1 and 2 - should equal net press run shown in A)	58000	70000

11. I certify that the statements made by me above are correct and complete

SIGNATURE AND TITLE OF EDITOR, PUBLISHER, BUSINESS MANAGER, OR OWNER: Michael J Des Chenes

PS Form (Page 1)

ABOUT THE COVER

This issue's cover shows the flagship of the new Atari computer line: the 600 XL. To produce this image, a slide of the 600 XL was composited onto a tinted circuit board pattern. Will the 600 XL start a new era of Atari computer products? How well does it stack up against the venerable 400 and 800? See our review on page 32 and judge for yourself.

ANALOG COMPUTING (ISSN 0744-9917) is published monthly for $28 per year by **ANALOG** Magazine Corp., 565 Main Street, Cherry Valley, MA 01611, Tel. (617) 892-3488. Second-class postage paid at Worcester, MA and additional mailing offices. POSTMASTER: Send address changes to **ANALOG COMPUTING**, P.O. Box 615, Holmes, PA 19043.

EDITORIAL

by Brian Moriarty

You're holding the first monthly issue of **ANALOG Computing**. It's our way of saying thanks to the 80,000+ Atari users who have helped make our magazine one of the finest machine-specific journals in the industry. We're going to do everything we can to deserve your continued support. Want proof? This overview of Issue 15 will give you an idea of what to expect in the future:

The 1983 Christmas season will be over by the time you read this, and final figures will be coming in on which home computers grabbed the biggest chunk of the market. The 600XL computer was Atari's do-or-die entry in the Santa Claus Sweepstakes. Can the new XL Series (or what's left of it) revive Atari's sagging stance in the home computer market? Read our 600XL review on page 32 and decide for yourself.

This month's Utilities Department features an elegant solution to an exasperating problem. Steve Howard's **Alternative Keyboard Handler** gets rid of DLI glitches, unwanted clicks and other nasties that have plagued Atari programmers for years. Cassette users will be kissing the feet of Bob Fine, whose **BASIC Recovery Program** can save hours of retyping when your program recorder is on the blink.

Tom Hudson fans will be happy to discover **BASIC Training**, a new companion for his already popular **Boot Camp** column. **BASIC Training** will help beginners get the most out of Atari BASIC while providing a useful refresher course for us old-timers who think we know everything. **Sally Forth** takes a quick squint at the FORTH systems available for the Atari, and the esteemed Dr. Griffin continues his monthly survey of educational software.

Ever wonder why the 5200's **Pacman** looks virtually identical to the 400/800 version? Claus Bucholz got out a screwdriver and came up with an in-depth technical comparison of the two hardware systems. His superb article reveals much of what you need to know about **Transporting Atari Computer Programs To The 5200**.

No issue of **ANALOG Computing** would be complete without a couple of public-domain games. You'll find an entertaining and instructive example of character set animation in Bruce Willard's **Knights And Chalices**. It's an action-adventure with the professional touches that prove how much can be accomplished with Atari BASIC. Assembly hackers will want to study the source for **Bricklayer's Nightmare**, Gordon Robson's fast-moving arcade fantasy that takes up only about 2K of object code. Just think, you could fit four games like **Nightmare** into a single **Super Breakout** cartridge! Don't overlook Joel Gluck's **Our Game** column, either. His little "Bounce" demo is good for hours of fun and experimentation.

We hope you enjoy our new monthly format. Don't hesitate to send in ideas, articles and programs — we need your input now more than ever before. In the meantime, keep an eye out for Issue 16, when we'll present some of the most advanced Atari graphics utilities ever published. □

ISSUE 14 CORRECTIONS

A sincere apology to John Euker of Houston, Texas, co-author (with Bernard Ertl) of the game program **Lumberjack**. His name was accidentally omitted from the article's byline, although it did appear in the table of contents.

A few of our readers reported difficulties with **Lumberjack** on the higher levels of play. Replacing line 1360 of the published BASIC listing with the following will eliminate this problem:

```
1360 FISH=C1:RAN=INT(C12*RND(C0)+C1):O
N RAN GOTO 1400,1400,1400,1400,1490,14
90,1490,1490,1580,1580,1580,1580
```

Ron Bishop of Tulsa Regional Atari Computer Enthusiasts (TRACE) discovered an obscure problem with **mUse**. If the BASIC code you're analyzing has a PRINT-comma statement in it, and you try to RUN the program after executing **mUse**, the system will lock up. The fix turns out to be very simple. Replace line 260 and add line 265 to the published BASIC listing as follows:

```
260 RESTORE LINE:FOR I=1 TO 24:READ BY
TE:IF LINE=1210 AND BYTE=77 THEN BYTE=
73
265 PUT #2,BYTE:NEXT I:READ CHECK
```

The fix is even easier to implement if you're working with the assembly version of **mUse**. Just adjust the BASIC warmstart entry address in line 250 to read:

```
0250 WARM =  $A049
```

and the PRINT-comma glitch will disappear. □

Great Software • Great Service • ~~Great~~ incredible Prices

from Synapse

Quasimodo

32K Tape or Disk **$23.88**

Soldiers pursue Quasimodo believing that he stole the jewels—but he didn't. As Quasi's alter-ego, you must avoid the manhunt and grab the jewels to prove your innocence.
All Synapse programs including Rainbow Walker, Dimension X and Blue Max available at similar savings.

from Infocom

Planetfall

32K Disk **$34.88**

Join the Patrol! Maybe your luck will change. Maybe you'll narrowly escape disaster or travel to an unknown corner of the universe, where you'll save a doomed planet or die trying. There's danger and the famous Infocom wit and humor in this one.
All Infocom releases including Infidel, Enchanter and The Witness in stock and available at similar savings.

from Datasoft

Zaxxon

16K Tape or Disk **$25.88**

The long-awaited version of the 3-D arcade hit of 1983. A must-have program for the space gamer.
All Datasoft programs, including Pooyan and Moon Shuttle in stock and available at similar savings.

from Thorn- EMI

Home Financial Management

16K Tape **$19.88**

Set up a budget and then see where you stand financially either in figures or through bar charts. This program should pay for itself!
All Thorn-EMI programs including Computer War, Jumbo Jet Pilot and Soccer in stock and available at similar savings.

from Koala

Koala Pad Touch Tablet w/Micro Illustrator

req. 48K Disk **$69.88**

One of the only truly "new" products of 1983. The Koala Pad attaches to the joystick port enabling you to draw, play music or play games (depending on the Koala Software being used) by using a stylus, or your finger, on the pad. Comes with the Micro Illustrator, an incredible disk-based drawing program. You can save the masterpieces you create to disk.

from SSI

Eagles

48K Disk **$27.88**

Fly yesterday's planes today. Fulfill your dream of being a WWI flying ace. Fly thru balloon-busting missions, one-on-one dogfights, and melees involving up to 20 planes. Play with a friend or against the computer. The skies await you!
All SSI programs including Combat Leader and Battle for Normandy in stock and available at similar savings.

from Big Five

Miner 2049er

ROM Cart. **$34.88**

Rated the top arcade game cartridge of 1983 by the staff of this magazine *and* by the staff at Software Plus. Ask us why we think it's so great—but be prepared for a long answer!

from Sirius

Bandits

48K Disk **$22.88**

Bouncing nerve gas balloons, energy shields, space bandits attempting to steal your supplies — they're all on the screen in one of the best space arcade games yet.
All Sirius Software including Capture the Flag and Way Out in stock and available at similar savings.

from Broderbund

Arcade Machine

48K Disk **$39.88**

This program contains several sample arcade games, which are fun by themselves, but, more importantly, it is a utility that lets you develop and play your own arcade games without requiring experience as a machine language programmer.
All Broderbund Software including Lode Runner, Choplifter and Bank Street Writer in stock and available at similar savings.

from Parker Brothers

Frogger

ROM Cart. **$27.88**

The official Atari cartridge version of the smash coin-op hit. Excellent graphics and game play.
All Parker Brothers programs including Q*bert and Popeye also available at similar savings.

from Sierra On- Line

Ultima II

48K Disk **$39.88**

After a long wait, fantasy role-playing game fans have a top-notch Atari program of their own. Disk comes with cloth map of the fantasy realm of Ultima and extensive rules.
All Sierra On-Line programs including B.C.'s Quest for Tires, Oil's Well and Homeword available at similar savings.

from Lightning

Mastertype

32K Disk **$25.88**

A breakthrough in interactive educational software. Learn to type as you battle wave after wave of invading words riding in hi-res spaceships! 17 pre-programmed lessons, plus others that you can create, guarantee continuous challenge and help you develop mastery of the Atari Keyboard.

Prices good thru Jan. 31, 1983

Call or write for FREE CATALOG (specify Atari)

Mail order prices do not apply at our retail location.

HOW TO ORDER:
Use mail or phone. For immediate shipment pay by Money Order, Certified Check, Cashiers Check, VISA, or Master Card (add 4% for credit card). Personal checks are accepted but require 1-3 weeks to clear. 10% deposit required for C.O.D.

SHIPPING & HANDLING:
$2.50 for orders under $50; free shipping over $50. Mass. residents add 5% sales tax. Prices are mail order only and are subject to change without notice.

SOFTWARE PLUS

(617) 369-1992

for orders and information
phone hours: 10-6 EST Mon.—Sat.

45 Walden St., Box AA, Concord, Mass. 01742

READER COMMENT

I recently picked up my first copy of **ANALOG Computing** Magazine. I was delighted and impressed, and have since subscribed. I wish you continued success with your fine publication.

I read with interest Sheila Neece Spencer's article **Sound Effector** (Issue #12), as the Atari's sound potential was one reason in my choosing it. I enjoy the program, but would respectfully submit a few minor enhancements. In order to achieve a gradual decay, the volume loop must step down with a negative number (e.g., vol=15 to 0 step -.5). By changing lines 230 thru 280 as shown, **Sound Effector** can accommodate this effect.

```
230 IF PL3 AND DL3=0 AND V
L3=0 THEN 330
240 IF DL3 AND PL3=0 AND V
L3=0 THEN 380
250 IF VL3 AND PL3=0 AND D
L3=0 THEN 430
260 IF PL3 AND DL3 AND VL3
=0 THEN 500
270 IF PL3 AND VL3 AND DL3
=0 THEN 610
280 IF DL3 AND VL3 AND PL3
=0 THEN 720
```

Also, lines 540, 650, 760, 880, 940, 1000, 1060 and 1120 should all read:

```
GOSUB 1200:GOTO 470
```

And now for a suggestion: Why not make **The ANALOG Compendium** available on disk or tape? Looking forward to your next issue.

David J. Kessel
Asst. General Manager
KAZM
Sedona, AZ

Thanks for the ***Sound Effector*** *enhancement. A multi-disk edition of the* ***Compendium*** *is available; no cassette edition is planned. Look for details in the* ***Compendium****.*

—B.M.

On Omnimon...

It was with both relief and concern that I read Brian Moriarty's review of Electronic Arts' games and their ridiculous protection against our product, **Omnimon!**. It is obviously an asset to have Brian's endorsement as to the advantages of having **Omnimon!** installed in one's system. However, the reader is left with the impression that it is not possible to run these games in the same system with the monitor. Instructions are provided in the **Omnimon!** documentation for the addition of a simple toggle switch, which makes the monitor invisible and thus compatible with all software.

But lest software publishers become too upset, let me point out that there are ways of making protection schemes less vulnerable to people using **Omnimon!** to break them. We have always given out this information freely to anyone taking the trouble to ask.

1) **Omnimon!** uses the bottom of the stack as its terminal input buffer. Any code down there will get wiped out when one enters **Omnimon!**

2) **Omnimon!** must use the top of memory for its graphics mode 0 screen data. Any code up there will likewise get wiped out upon entry to the monitor.

3) During execution of critical parts of the protection mechanism, the program could monitor the SELECT button to see if someone is fixing to interrupt the program with **Omnimon!**. The program would then have plenty of time to scramble itself before it was actually interrupted.

After public reaction to the outrageous protection scheme **Zaxxon** initially came out with, Datasoft has become much more discriminating and their products are now, as far as I know, compatible with everybody else. I hope Electronic Arts will now take the same course of action. By making the above information available by sending them an **Omnimon!** unit free of charge, I have done my best to cooperate.

Sincerely,
David Young
CDY Consulting
Richardson, TX

Instructions for installing a disable switch were not provided in the original documentation for **OMNIMON!**, *which I reviewed in Issue 12. Mr. Young assures me that the latest version of the product does include this information. His candidness and responsible attitude towards the Atari community set an example which other software and hardware manufacturers would do well to emulate.*

—B.M.

Software Piracy: The last word?

As soon as I saw the letter in **ANALOG** Issue 13, I knew I had to write. I'd seen many letters like this one (variations on a theme, you might say), but this was the one that broke the programmer's back, if you'll pardon the mangled metaphor. What was in the letter that got me so steamed? Some quotes:

"*. . . most software is grossly overpriced, especially when you consider*

the price of media and documentation."

"If software companies would reduce. . . a $35 program to. . . the low twenties I think there would be less piracy."

"It's harder to justify spending. . . money on a copying system if the price of legally buying the program is low."

If I hear any more of this swill in defense of theft, I'm gonna puke. And make no mistake, pirating software is theft. But first, an introduction. I'm a games designer and programmer. I make my living by writing games. I don't do it in my spare time. I'm not a programmer for IBM by day and a hacker by night. I sit at my desk, writing, programming, designing, sometimes 20 hours a day, in an effort to write the best and most fun games that I know how. And what's the result? "Gee, man, I just can't see paying $34.95 for your game, so I stole it. I mean, if it was $24.95 then I would've bought it, but that extra ten bucks. . ." Bull. This person would have stolen it, regardless of the price. Yes, I know that $34 or $40 is quite a bit of money, but the "it's too expensive" line is just an excuse. In fact, all of the arguments quoted above are just excuses. And so that they can't be bandied about quite as easily anymore, I want to look at a few excuses, and the facts behind them.

Excuse #1: The all-time champ. "It's too expensive. If it were just $25 (or $20, or $15, or. . .)." People look at the cost of disk (about $2), the cost of packaging (add another $2), and voila! decide that it costs the company about four bucks to make a game. They then look at the $34.95 selling price, and in a lightning-quick burst of mathematical insight, conclude that the manufacturer is getting around $31 profit. Well, I hate to break the news, gang, but that ain't how the real world works. First of all, the media and packaging costs are not the only expenses a software manufacturer has. They're not even the largest part of the cost. Here are some other items that your friendly neighborhood pirate hasn't been taking into account:

Advertising. You think all those 4-color super-graphic ads that you see every month in 15 different computer magazines are free? Let's not be absurd.

Artwork. Let's see, the artist does all that artwork for nothing, right?

Inventory, shipping, overhead. All of these are extremely unglamorous (and easily ignored) costs. The manufacturer has to have a place to put all these umpty-ump copies of **Galactic Death** and **Mega-Ants**. He also has to pay to have them shipped to distributors and stores, he has to pay insurance, salaries, building rental, and on and on. And all of these costs have to be made up by the sale of the games. Otherwise we've got one more belly-up video game company.

(continued on next page)

Also, the manufacturer is *not* getting $34.95 for the game. Often, he's getting $25 or less. Your local software store then marks up the price (he's got rent to pay, too), and sells it to you. Make no mistake, people are making money on video games. But they're not making the outrageous amounts that pirates would like you to believe.

Excuse #2: "It doesn't hurt anybody." Well, I thought this one had long since been laid to rest, but it still manages to rear its ugly head every now and then. I can name you one person that piracy hurts very directly. Me. If you pirate my game instead of buying it, that's money out of my pocket. And if enough money disappears that way, I won't be in this business much longer. Because no matter how much I enjoy writing games, and I enjoy it very much, if I can't pay my rent and living expenses with game designing, then I'll have to pay them some other way. And there won't be too much incentive for me to release any designs that I *do* come up with. If your attitude is "Gimme," then why shouldn't mine be "Get lost?"

Excuse #3: "The game isn't worth $34.95. The (game play/graphics/design) are lousy." Fine. Feel free not to buy it. After all, nobody has a gun at your head, saying "Buy this game." And if the excuse is "Well, it isn't worth $34, but I still like it," you have to make a decision. Either you like the game enough to pay 34 bucks for it, or you don't, in which case you don't get to play it at all. You can't have some in-between area, like "I like it 25 bucks worth." There are lots of games out there that are worth far MORE than their purchase price. Buy one of them. But as I said before, if you don't like a game enough to buy it, *don't buy it.*

Excuse #4: "There's nothing wrong with making a copy of a game." This excuse has the dubious virtue of at least being honest about the situation. The person employing it sees nothing wrong with stealing someone else's work. There's only one problem with this excuse. It's dead wrong. There is *everything* wrong with stealing someone else's work. It is a morally, ethically and socially indefensible position. We are not talking about a life-threatening situation here. Nobody is going to die if we don't steal the game, and no one is going to be saved if we do. It is a little bit like asking Leonardo Da Vinci to paint a picture for us, and then telling him "Well, Leo, it isn't worth $34.95, but tell you what, I'm going to take it anyway." What about his time? What about mine?

In the final analysis, this is the position. You, as a pirate, have no right to any particular game. I don't care what its price is, if you are not willing to pay it, you have no ground to stand on, none. Not legal. Not moral. Not ethical. You have no right to steal someone's efforts, and then rationalize that theft with vague (and, as we have seen, spurious) arguments about "excessive price" and "moral right." I don't know which of the arguments above you use, and frankly I don't care. Whichever one it is, I don't want to hear it.

Piracy is not a monetary issue, nor a legal one. It is a moral issue. It is wrong to steal someone else's program, and until the pirates have the maturity to realize this and cease their piracy, manufacturers will have to spend ever-increasing amounts of money to protect their programs (doesn't sound like the price is coming down, does it?), magazines will continue to print letters like this, and the pirates will continue to whine about how abused they are.

Alexander Leavens
Fremont, CA

Send letters to:
Reader Comment
P.O. Box 23
Worcester, MA 01603

Start off on the right foot with an ATR8000 for your Atari. In the long run, it makes more sense.

You're going to expand your Atari®... So why not do it right.

Your first step, purchase of an Atari 400 or 800, was smart. Your next logical step is to buy a SWP ATR8000™ interface unit.

An ATR8000 gives you more capability for less money. You can start small, but you can build big . . . from an outstanding home computer to a powerful CP/M™ machine. And prices for the ATR8000 start at only $349.95.

Yes . . . I'd like to know more about SWP's ATR8000 and CO-POWER-88 for my Atari 400/800 computer. Please rush me free literature.

Send to:
SWP Microcomputer Products, Inc.
2500 E. Randol Mill Rd. - 125
Arlington, Texas 76011
200

name

street address

city state zip

MAIL TODAY!

DEALER INQUIRIES INVITED.

SWP Inc MICROCOMPUTER PRODUCTS
2500 E. Randol Mill Rd
Arlington, Texas 76011
817/469-1181

We used to be Software Publishers, Inc.

What you get first, in the amazingly adaptable ATR8000, is plug-together interfacing to all ordinary system devices:

- To floppy disk drives — The ATR8000 lets you hook up from one to four low-cost add-on drives, 5" or 8", one- or two-sided, in any mix . . . and go right to work with either a single- or double-density DOS. (DOS is optional.)
- To a printer — The ATR8000 parallel "Centronics-interface" is supported by a ROM driver program that you can access from Atari Basic.
- To a modem, to a serial data terminal or to any other RS-232 communicating device — The comprehensive users manual includes an Atari Basic serial printer driver.
- To Atari peripherals such as the 810 Disk Drive.

And of course the ATR8000 interfaces to the 400 or 800 computer itself. But system versatility is just the beginning; the ATR8000 also opens the door to another world of microcomputing.

Atari®-CP/M™ — A whole new world of personal computing

Quite simply, CP/M is the dominant operating system of the 8-bit microworld. Make the ATR8000 your first system expansion unit, and you can put the CP/M world in your own little jug.

Since the ATR8000 is actually a self-contained Z-80 computer, a low-cost upgrade converts your Atari-ATR8000 system into a full-capability CP/M machine. Of course you can also start right off with a CP/M-ATR8000. In either case, you get CP/M program-development capability, plus access to an enormous base of proven application programs that just load and go. The CP/M upgrade, which includes 64 Kbytes of RAM, costs just $249.95.

And it doesn't end there!

You can further upgrade your CP/M-ATR8000 to 16-bit microprocessing power . . . to CP/M-86 or MSDOS operation. Moreover, with SWP's CO-POWER-88™ upgrade, you not only retain your 8-bit processing capability, but you also can use CO-POWER-88 RAM — up to 256 Kbytes — as fast, simulated disk storage for CP/M operation.

Like the ATR8000, CO-POWER-88 comes with comprehensive documentation. Prices begin at $749.95 with 128 Kbytes of RAM and CP/M-86 operating system.

PROFESSIONAL BUSINESS GRAPHICS FOR THE ATARI — AT LAST!

by Tom Kelley

B/GRAPH
by Michael Reichman and Robert Wilson
INHOME SOFTWARE
2485 Dunwin Drive, Unit 8
Mississauga, Ontario
Canada L5L 1T1
48K BASIC and machine language
Two disks and manual, $99.00

"These numbers are just fine, and you've done a good job, Tom, but a few graphs could really sharpen the picture."

Those comments were true, but I sure could have done without them. I grumbled at the Atari 800 sitting near my desk as I walked into my office. I thought to myself: If it had a rear end, I'd kick it.

My Atari 800 was purchased with enthusiasm when only a few micros were available in the winter of 1980. The only real competitor close in price at the time was the Apple II, and the design of the 800 was clearly 'second generation' by comparison. Little did I know that business-oriented software writers would flock to the Apple II and ignore the Atari 800.

Even so, I still use **Visicalc**, **Text Wizard** and **Filemanager+** and, except for limited spreadsheet size, I can manage project planning, sales analysis, sales and promotion tracking, and field communications just fine. I've just never been able to make good graphs.

Sure, I've tried Atari's **Graph It**, but quite frankly, it's pathetic; and even with a screen dump routine for my Epson printer, it can only do fair graphs at best. No custom labels, and forget about altering data, or going from a line graph to a bar chart.

What a paradox! A great graphics machine, competitive with $4000 business micros, and no decent graphing software! If I sound disillusioned, I am. But let me get off my soapbox.

Enter **B/Graph**, a professional graphics-charting and statistical analysis program. Is it the answer? Read on and find out.

Hardware requirements.

The first menu of **B/Graph** is unique among Atari software. It asks you to define your printer, and then loads the appropriate "printer driver" into your machine. It's nice to see a software company acknowledge that computer purchasers don't always buy the same printer. In fact, you don't need any special boards, chips or other fancy accessories to run **B/Graph**. You will need a 48K Atari computer, a disk drive, Atari BASIC, and an 850 Interface (if you want hard copy).

Here's a partial list of printer manufacturers that **B/Graph** supports:

Centronics
Epson with Graftrax
C.Itoh
N.E.C.
Seikosha AT/100
Okidata 92
Gemini
Prowriter

Once you've selected the appropriate printer, then you move to the main **B/Graph** menu. **B/Graph** is actually 9 different programs. Together, they allow you to format the data in the most meaningful visual format. If you have straightforward data, you will be using Graphing, Pie Charts, and the Graph Imaging/-Labeling programs the most. These are the graphics generating programs.

Is it easy to use?

I was using the graphing module to make graphs within 10 minutes of receiving the package in the mail, and I'm no computer wiz or hacker. Just a user.

The graphing modules (both graphing and pie chart) are outstanding. The sample graphs in **Figure 1** were generated on my system and printed with my Epson MX printer. **B/Graph** uses a unique combination of single key commands plus menus to manipulate the graphs. This format allows for detail, yet it's fast. The functions are in English, and logical. For example, should you forget a command while

the graph is on screen, pushing OPTION returns you to the prompting menu, but don't worry, pushing "R" instantly re-displays your graph. **B/Graph** uses the Atari's highest resolution mode to draw with. By using artifacting, color is sharp and clear, and the graphs have excellent detail on color TVs as well as monitors.

Once you've plunked in some basic data called "factors," you can begin drawing graphs. And *what graphs!* Your choices include: line graphs, point graphs, bars (in 3D), segmented bar (with bars stacked), and market (either tic or connected) for stock charting. I will assume the reader has some knowledge of graphs, and if you don't, the **B/Graph** manual contains an excellent tutorial on common types of graphs.

Graphing features.

Here's the real power of **B/Graph:** Any graph can be changed almost as fast as you can type. If your data isn't meaningful as a point graph, press s (for switch) and choose a bar, and **B/Graph** will draw you a new bar chart with the same data. You can change your mind as often as you like, or draw your graph several different ways and then decide which is the most effective graph. To change actual data is just as easy as using the standard Atari editing features. You are allowed up to 3 factors for any particular graph. More than this tends to get messy anyway. Plus, you can have up to 100 data points for each factor. That large a data file would only be needed for a complex analysis, but it deomonstrates the author's dedication to making the program useful.

Bar graphs are brilliantly displayed in 3-D until you reach a total of 21 columns or bars, at which time they automatically become two-dimensional. 48 columns are the limit for bar charts. Segmented bar charts are simply bar charts with 2 or 3 factors, stacked on top of each other. The segmented bars will be 3-D up to 21 columns, and then switch to two-dimensional.

B/Graph sets all the scaling automatically, including negative numbers. However, you can rescale to a scale of your own choosing any time. To enhance your graph you can FILL in under a line graph (or between factors), ATTACH points (point graph only), add or remove a GRID, add BORDERS, ALTER LABELS, CHANGE INTENSITY, CHANGE HUE, add or remove vertical GRIDS, or even OVERLAY a previously stored graph. If you need lots of labels you can use the Imaging/Labeling program to add countless variations to your basic graph.

B/Graph contains one more important graphing/image processing capability in its IMAGING program. This module allows you to set up a simple slide show format, and instruct the computer to place your graphs on the screen in sequence. Each can be called up to the screen on your command. Imagine those board room presentations!

While all this graphics power is exciting, **B/Graph's** power would be limited without its FILE MANIPULATION module. This function lets you format your data and perform algebraic functions on it, including simple statistics. One utility allows you to load and convert **Visicalc** files to **B/Graph** format. There is even a function to "browse" your data when you are done, which prevents you from jumping out of the manipulation module and back to the graphing module only to find that your data was not properly prepared. This can prevent lots of frustrating mistakes.

Statistics.

Graphing and fancy images are only half the story with **B/Graph**. The other half is *statistics*. While many of the functions are mathematically complex, their usefulness to a researcher or statistician is outstanding. You can perform regression analysis (often used in forecasting), CHI square tests, T tests, F tests, and most of the standard functions found in a college statistics text book. It is important to note that you don't have to have much knowledge of statistics to make good use from these **B/Graph** modules, because the manual provides excellent descriptions of their practical uses.

File structure.

B/Graph uses an edited version of Atari DOS 2.0; the authors call it "Mini-DOS." In case you built forget to format a data disk before starting, you can do it from within **B/Graph**. You can also perform other DOS functions like locking, unlocking or deleting a file. For some users, the best news could be that the standard DOS file format means you can access **B/Graph** files with your own programs.

There is one more remarkable feature of **B/Graph** that I must salute. Finally, somebody built in a logical method of getting a disk directory when you really need it. That is, of course, when you are about to load or save a file or image. If you don't know which file to load or what to name a file when saving, simply leave the title entry box blank and just hit return! You will immediately be presented with a screen with all the titles available on the disk in your drive. **B/Graph** even tells you how much room is left on the disk. At this point you can continue your disk load or save, or even hit escape to return to a main menu. How simple and logical!

Inhome Software is providing support for **B/Graph** and has produced an excellent manual. It is written for the uninitiated user by an author that speaks English (as opposed to computerese), Ian Chadwick. It is an easy manual to understand. Inhome has even published a phone number for registered users who need help.

With a call to Inhome, I discovered that they plan to release more software, and when extentions/im-

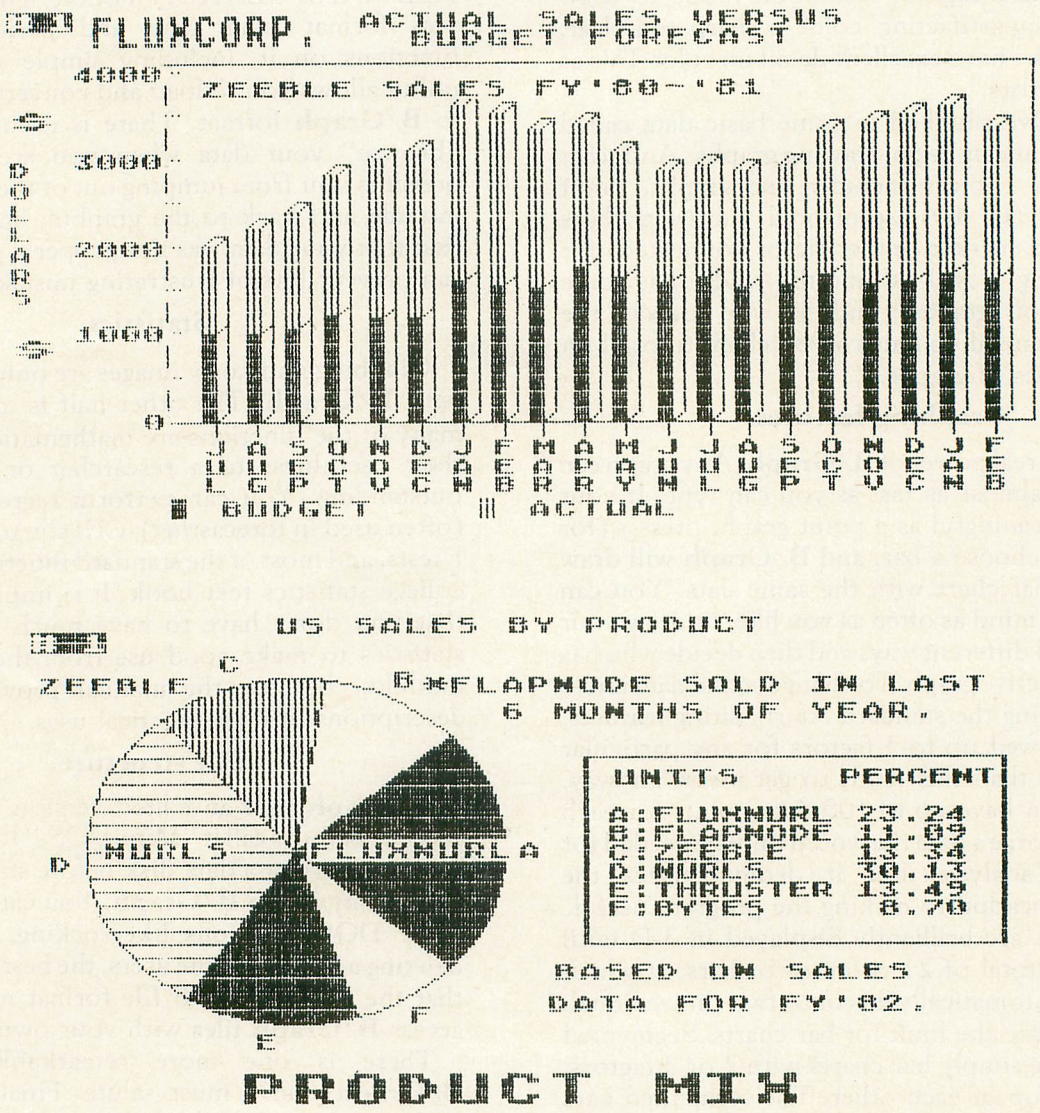


Figure 1.
Sample B/Graph printouts.

provements become available for **B/Graph**, registered owners will be notified and offered an upgrade for a nominal fee ($20). One of the first upgrades will include support for some of the new low-priced plotters about to hit the market.

Will there be a **B/** series? As of this writing, Inhome isn't talking, but I would not be surprised if they were encouraged to write a **B/CALC**, **B/WORD**, **B/TERM** or **B/FILE**. I can think of a dozen improvements I'd like to see with things like **Visicalc**, and a terminal or data base management program. I hope their software development direction will come in part from what their customers encourage them to try.

Inhome Software's B/Graph is comparable to **PFS:GRAPH** and **Visiplot**, neither of which is available for the Atari. I've used **Visiplot** on my company's Apple II, but it is slow and awkward to use compared to **B/Graph**. While **B/Graph** is not quite as fancy or detailed as **Chart/Draw** on our multi-million dollar mainframe Hewlett Packard 3000, it is more user-friendly by far!

B/Graph is clearly the best package of Atari business software to appear since **Visicalc**. It's a little late in coming, but it represents an excellent value and sets a new standard of performance for products of this type. Now, excuse me while I start shopping for a color plotter. □

ATARI SOFTWARE FOR THE WHOLE FAMILY

Here are four software packages designed for the different people in your family.

A BASIC COMPILER FOR THE PRO-GRAMMER

ABC (A BASIC Compiler) automatically translates Atari BASIC programs into high-performance integer P-code that runs up to 12 times faster!

Perfect for developing system software and commercial games, ABC accepts most BASIC programs (unless floating point dependent) with little or no modification. Compiled P-code is a self-standing DOS object module that is unLISTable and runs without the BASIC cartridge.

ABC allows expressions in DIM, GOTO, GOSUB, and RESTORE statements, doesn't require you to re-order lines, and fully supports string and sub-string operations.

Give your BASIC programs the look and "feel" of professional products with ABC. 40K Disk ***$69.95***. Manual alone ***$9.95*** (credited toward compiler purchase).

MAKEBOOT lets you create self-booting disk or cassette versions of your ABC compiled software. Reduces overall program load time and saves memory and disk space by eliminating DOS. 40K Disk ***$14.95***.

TOOLS FOR THE SERIOUS USER

Power Tools I combines four sophisticated text processing tools on one easy-to-use utility disk.

DIFF shows you differences between two ATASCII text files: for example, changes you made in a program or document from one version to the next.

Manually searching for a particular text file can take hours. But now, with *SEARCH,* you specify a search string and a list of files. *SEARCH* examines the files and points out which ones have that string. You'll never lose your Fudge Brownie recipe again!

CHANGE is a powerful search and replace utility that operates on multiple files with one command. For example, you could change character names throughout your novel with a single command, even if each chapter is a separate file.

The special pattern-matching and multiple disk capabilities of *SEARCH* and *CHANGE* are an added plus.

TRANSLIT lets you swap one character set for another (for example, upper case for lower case) throughout a file with one command.

Power Tools I is ideal for professional business and software development text applications. 40K Disk ***$34.95***.

AN EDUCATIONAL TOY FOR PRE-SCHOOLERS

Monarch is proud to present ***SofToy***, an educational program smart enough to act simple.

Bells ring, balls bounce, owls hoot as SofToy and its colorful interactive display gently introduce children (two years and older) to spatial relations, letters, numbers, even elementary programming! SofToy lets kids become familiar with computers, without arbitrary demands, competition, or intimidation. SofToy grows with children, too. At more difficult levels, the match game is a real challenge for the whole family. 24K Disk ***$29.95***.

Look for these fine products at your nearest ATARI dealer, or order direct. Include $2.00 for shipping and handling in U.S./Canada. Checks or money orders accepted. MasterCard/Visa add 3%. COD higher. MA residents add 5% tax. Dealer and distributor inquiries invited. ATARI is a trademark of Atari, Inc. ABC, Makeboot, PowerTools, and SofToy are trademarks of Monarch Data Systems, Inc.

MONARCH DATA SYSTEMS

MONARCH DATA SYSTEMS, INC., P.O. Box 207, Cochituate, MA 01778 (617) 877-3457

BASIC CASSETTE RECOVERY PROGRAM

16K Cassette

by Bob Fine

So, you've just finished typing in your thousand line ULTIMATE PROGRAM, and decided to take a break. You save the program to cassette, and go. Later, in an attempt to impress your friends with the program, you are greeted with ERROR-148 or ERROR-133. Not only do you feel a bit foolish trying to explain where the program is, but angry that you have to type the whole thing over again.

BASIC Recovery is a program that resides on page six, and will retrieve as much of your BASIC CSAVEd program as possible. Very often, the error occurs in the part of the program flagged as the final frame — which contains no data. This is the frame that tells the ATARI when to stop reading the tape. If there is an error in the final data frame, the entire program is erased and all of the BASIC pointers are reset. This program will help you recover most of your "lost" program as long as the first data frame is complete.

There is an excellent discussion about the format of BASIC tokenized tape files in *De Re Atari*, so I will only briefly go into it.

The most important information in a CSAVEd BASIC file is in the first data frame (a frame is the noisy part of the tape between the short silences). Each frame consists of two timing bytes, followed by one control byte, followed by 128 data bytes, and finally a checksum byte. The operating system takes care of the first three and the last byte.

The first frame's first fourteen data bytes deal with seven BASIC pointers: LOMEM; VNTP, or Variable Name Table Pointer; VNTD, or Variable Name Table Dummy end pointer; VVTP, or Variable Value Table Pointer; STMTAB, the Statement Table; STMCUR, the Current Statement pointer; and lastly STARP, the String/Array area Pointer.

As you can gather from the names, these pointers are necessary for the running of a BASIC program.

The **BASIC Recovery Program** opens the cassette, reads the above mentioned pointers, performs the necessary calculations and then proceeds to read as much of the file as it can, byte by byte. When it's done, it will tell you if the load was complete or incomplete, and wait for you to press a key. When you do, it jumps through the SYSTEM RESET vector, and you will have your program up to the error. It is just a matter of typing in the remainder of your program and resaving it. You can also load a CSAVEd program with no errors.

In order to use the **BASIC Recovery Program**, type in the BASIC program provided, save it (since it removes itself from BASIC), and type 'A=USR(1536)'. Any time until you turn off the machine or modify page six, you can use this program.

If you have any questions, please write to me at: Bob Fine, P. O. Box 76, Tomkins Cove, NY 10986. □

BASIC listing.

```
1 REM BASIC RECOVERY PROGRAM
2 REM by BOB FINE   May 5, 1983
3 REM SAVE PRIOR TO USE
4 REM CALL WITH [ A=USR(1536) ]
5 REM Each line of data has 16 numbers
10 TRAP 100:RESTORE :I=0
20 READ A:POKE 1536+I,A:I=I+1:GOTO 20
100 TRAP 40000:NEW
1000 DATA 104,165,128,133,0,165,129,13
3,1,230,1,162,16,169,3,157
1010 DATA 66,3,169,251,157,68,3,169,6,
157,69,3,169,4,157,74
1020 DATA 3,169,128,157,75,3,32,86,228
,48,75,160,0,140,255,6
1030 DATA 32,207,6,48,65,72,32,207,6,4
8,59,170,104,172,255,6
1040 DATA 24,101,128,153,128,0,200,138
,101,129,153,128,0,200,192,14
1050 DATA 208,219,165,140,133,142,133,
144,133,14,165,141,133,143,133,145
1060 DATA 133,15,32,207,6,48,15,160,0,
145,0,230,0,166,0,208
```

```
1070 DATA 2,230,1,76,98,6,192,136,240,
22,162,0,142,255,6,189
1080 DATA 218,6,32,164,246,174,255,6,2
32,224,11,208,239,76,163,6
1090 DATA 162,0,142,255,6,189,229,6,32
,164,246,174,255,6,232,224
1100 DATA 10,208,239,162,16,169,12,157
,66,3,32,86,228,162,0,142
1110 DATA 255,6,189,238,6,32,164,246,1
74,255,6,232,224,14,208,239
1120 DATA 174,252,2,224,255,240,249,16
9,255,141,252,2,76,116,228,162
1130 DATA 16,169,7,157,66,3,32,86,228,
96,73,78,67,79,77,80
1140 DATA 76,69,84,69,155,67,79,77,80,
76,69,84,69,155,80,82
1150 DATA 69,83,83,32,82,69,84,85,82,7
8,155,67,58,155,0,0
```

CHECKSUM DATA

(see p. 70)

```
1 DATA 409,435,115,334,512,276,813,663
,824,245,366,301,91,472,32,5888
1070 DATA 351,857,909,624,719,25,262,1
75,101,4023
```

Ask Sally Forth

by Sally Forth

My original plan for this month was to talk about the concept of modular programming, and what it means to you, the Atari FORTH programmer. But Joel Gluck's column **Our Game** (page 40) is coincidentally devoted to the same subject; and I've got lots of reader mail piling up in my hopper. So let's violate the "age before beauty" maxim just this once, and allow Joel to extoll the wonders of modular programming. Rest assured that everything he says applies in principle to FORTH as well as BASIC. Where's my letter opener?

Please review different implementations of FORTH for the Atari. This would enable new FORTH users to select their version, and would let present users know what features are available in other variations (I have **PNS-FORTH**).

Howard Brazee
Lakewood, CO

That's a tall order, Mr. Brazee. It turns out that there are no less than six commercial implementations of FORTH available for the Atari, plus a number of "underground" systems available through various grapevines and user groups. The cruel editors won't give me enough space to review all of them in depth, so you'll have to contact the publishers at the addresses provided if you need more information.

Extended fig-FORTH Rev. 2
by Patrick Mullarky
ATARI PROGRAM EXCHANGE
P.O. Box 3705
Santa Clara, California 95055
16K Cassette/24K Disk $39.95
Extension Packages Additional

This is an unspectacular but usable version of standard 6502 fig-FORTH, modified for the Atari. It comes complete with a Ragsdale-type assembler, fig's clumsy Editor, a handful of debugging tools and a nice little set of operating system and floating point definitions. The graphics words provided with **APX FORTH** give you about as much picture-drawing power as Atari BASIC. Thankfully, APX also offers a couple of extension packages (**fun-FORTH** by Joel Gluck for $24.95; **FORTH Turtle Graphics Plus** by William Volk for $17.95) which make the basic system a lot more attractive as far as graphics are concerned. Another accessory package, **FDOS** by George Schwenk, allows **APX FORTH** users to access standard Atari DOS files ($39.95 from Superware, 2028 Kingshouse Road, Silver Spring, MD 20904). **APX FORTH**'s 50-page reference manual assumes prior familiarity with FORTH; a copy of fig's *Handy Reference Guide* with printouts of all extension screens are thoughtfully included.

QS FORTH
by James Albanese
QUALITY SOFTWARE
6660 Reseda Blvd., Suite 105
Reseda, California 91335
24K Disk $79.95

QS FORTH was the first FORTH system offered commercially for the Atari. Like **APX FORTH**, it offers limited graphics support and very few bells and whistles. An assembler and LOCATEing screen editor are included, but there are no provisions for floating-point math and minimal debugging facilities. The documentation for **QS FORTH** is better than average. It's a 160-page notebook, professionally printed and arranged like a mini-tutorial on FORTH, with many useful examples and a good description of the internal workings of the system. Too bad the only printed source listing is for the assembler.

PNS fig-FORTH 1.4
by Robert Gonsalves
PINK NOISE STUDIOS
P.O. Box 785
Crockett, California 94525
32K Disk $90.00

This FORTH (marketed chiefly by Mountain View Press) is crammed full of goodies for the Atari stacker, including a sophisticated player/missile graphics system that uses vertical blank interrupts (VBIs) to simulate multi-tasking. Ragsdale's assembler, a very usable screen editor (second only to **valFORTH's**), a system glossary, complete source listings and a strange but serviceable string-handling vocabulary are also included. Surprisingly, there is no floating-point support. The photocopied reference manual is impressively thick and comprehensive but very poorly packaged; where did they find a stapler mighty enough to handle one solid inch of paper?

ECS/MVP-FORTH
by Phil Koopman Jr.
MOUNTAIN VIEW PRESS
P.O. Box 4656
Mountain View, California 94040
32K Disk $100.00
Extended Version $175.00

Unlike most Atari FORTHs, which are based on the familiar fig-FORTH subset, Mountain View Press's official FORTH system implements the full FORTH-79 standard. It's also the only Atari-compatible FORTH that offers any potential for transportability: the MVP dialect is available on just about every micro you can name. The basic $100.00 ECS/MVP system is competent and includes a reasonable screen editor, but it lacks the special audio-visual features that make Atari folks sit up and take notice. $75.00 more will get you a package of interesting graphics and animation extensions, a "multi-tasking" sound vocabulary and an unlimited run-time license. Neither system supports floating-point. The scant reference manual assumes that you have access to the Haydon book *All About FORTH* (available, like most important FORTH literature, from MVP). If Mountain View and ECS Software decide to support Atari **MVP-FORTH** as well as the IBM-PC and Apple II versions (with source listings, target compilers and other professional delights), MVP could become the FORTH of choice for serious Atari software authors.

Extended FORTH
ELCOMP PUBLISHING, Inc.
53 Redrock Lane
Pomona, California 91766
32K Disk $39.95

Aside from the usual editor and I/O access features, this fig-based system includes limited player/missile graphics support, a couple of sound definitions and a few handy debugging words. If you need floating point math, purchase the optional Floating Point Package ($29.95). Elcomp also offers a beginner's mini-system called **Learn-FORTH** on either disk or cassette; its $19.95 price tag makes it one of the least expensive introductions to FORTH programming on the Atari.

valFORTH 1.1
by Steven Maguire and Evan Rosen
VALPAR INTERNATIONAL
3801 East 34th Street
Tucson, Arizona 85713
32K Disk $59.95
Extension Packages Additional

I've saved the best for last. **valFORTH** is the most complete FORTH implementation available for the Atari. It's a language *system* in the truest sense of the word: everything you need to develop, debug and sell finished FORTH applications is included. Valpar's basic $59.95 introductory package includes a choice of two screen editors (fig's and a slightly better custom version), a souped-up 6502 assembler, a good floating-point package plus operating system I/O and memory allocation words. Documentation and packaging are superb; they even supply dividers for the notebook and little labels for the dividers!

Valpar's extension packages really make **valFORTH** sing. Must-haves include the Editor/Utilities package ($49.95, containing Valpar's magnificent 1.1 Video Editor and a nifty string-handling system) and the Player/Missile Graphics package ($49.95). Other interesting but less essential options include the Display Formatter (a custom display list generator, $39,95); valGraphics (high-speed turtles, $54.95); Text Compression/Formatting (an adventure-writing toolkit, $39.95); and valDOS ($59.95).

The power and accessibility of **valFORTH** have already established it as the de facto FORTH standard of the Atari community. Until somthing better comes along, I'll continue to use it for all of the FORTH demos appearing in this column.

The only underground FORTH system worth mentioning goes by a number of different names. "Free FORTH," "Sunnyvale FORTH" and "Team FORTH" all stand for the same public-domain package developed somewhere in the San Francisco Bay area by Steve Calfee, Harold Striepe, Peter Lipson and Robin Ziegler (among others). The system consists of two disks: a system disk with bare-bones kernel and several utility screens, and a tutorial disk with all the utilities in compiled form. It's not the best documented or most advanced FORTH system in the world, but it's dandy for experimenting and the price is hard to beat. Where to find it? Your local user group library is the likeliest place. Failing that, check your local bulletin board system or computer store for leads.

One of the main reasons I got an Atari was so that I could learn how to program games, and I don't doubt that this is true of many ANALOG *readers. Although I've read that FORTH is good for writing game programs, I have yet to see any code in FORTH for an "arcade type" game. So here is a question for you: Where can I find an example of a FORTH game program, similar to those which appear in every issue of* ANALOG *in BASIC and machine language? How about treating* ANALOG's *FORTH audience to a complete game, instead of just a few example screens and snatches of code?*

Joe Rockey
Highwood, IL

Your hopes are shared by many of my readers, Joe. I certainly have no objection to publishing a game in FORTH, and neither do the editors of ANALOG as long as the source code is of reasonable length. If any of my readers would like to share their FORTH shoot-em-ups with the rest of us, send me the source screens (on disk, please) in care of ANALOG and I'll be happy to publish them with your name on the top.

Old Sally's been working on a few game ideas of her own. Next month, I'll present a super-high-speed PLOT routine that will form the basis for a very ambitious entertainment program. □

Sally welcomes your comments about the FORTH programming language, and will publish her most interesting letters in future issues. Write to her c/o ANALOG Computing Magazine, P.O. Box 23, Worcester, MA 01603.

IF YOU HAVE THE ATARI, WE HAVE THE PRINTERS.

GP-700AT
Full Color Printer — $599.

COLOR — is the big feature of our GP-700AT Atari direct-connect printer. It's fast, quiet and the bright colors will make any presentation come alive. We provide cable and screen dump programs to eliminate any installation problems. 850 interface is NOT required, even if you daisy-chain a disk drive or cassette recorder. You can print, plot and screen dump in 25 colors with Axiom's GP-700AT.

GP-550AT
Dual Mode Printer — $399.

QUALITY — is the keynote in our near letter quality mode with a density normally costing twice as much. You get friction, fully adjustable tractors, and even bottom feed for label printing. 850 interface is NOT required. We supply cable, manual and screen dump programs to get you off to a flying start. You can even daisy-chain a disk drive or cassette with the GP-100AT Atari direct-connect printer.

GP-100AT
Economical Printer — $299.

COST — is small but features are many. You get graphics and adjustable tractors. Uses full size standard fan-fold paper. Just take it home, plug it in and you'll be printing in minutes. The GP-100AT Atari direct-connect printer comes with screen dump to provide spectacular graphics, and you can daisy-chain other peripherals. Again, NO 850 interface is required.

AT-WORDSTORE
Printer Buffer — $299.

EFFICIENCY — is what you get with our 32K printer buffer. Features include pause and copy. WORDSTORE can be connected to Atari computers WITHOUT an 850 interface. Other peripherals can be daisy-chained. Accepts any Centronics-parallel input printer. Two year warranty. Best of all, WORDSTORE is priced well below other printer buffers.

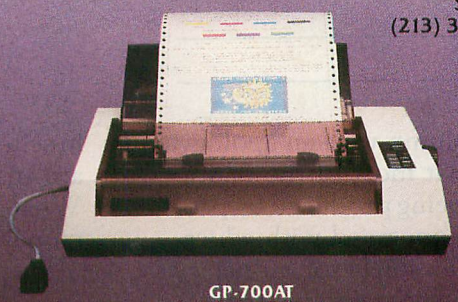
AT-846
Expansion Interface — $99.

SIMPLICITY — it's just like Atari's 850 interface, but without the four funny serial ports. Great for interfacing any parallel printer to your Atari computer. Can daisy-chain other peripherals, too. By the way, if you think up a name for this product, drop us a line. If we use it, we'll send you a color printer free. Contest ends 12-31-83.

AXIOM®
AXIOM CORPORATION

1014 Griswold Avenue
San Fernando, CA 91340
(213) 365-9521 • TWX: 910-496-1746

AT-846
Expansion Interface



GP-700AT
Full Color Printer

GP-100AT
Economical Printer

GP-550AT
Dual Mode Printer

AT-WORDSTORE
Printer Buffer

GP-Series Printers
Built for
Lasting Quality
by SEIKOSHA.

GRIFFIN'S LAIR

EDUCATIONAL PROGRAMS REVIEW

by Braden E. Griffin, M.D.

I must begin this issue's column with an apology. In the last issue, I implied that most of the educational software available was being produced by people who were not primarily concerned with education, but had entered this arena solely for monetary reasons. Though this may be true for some, there is an abundance of excellent educational software for the Atari being produced by a variety of manuacturers.

The editors of **ANALOG** have provided me with such an enormous volume and assortment of programs that it has become a chore just deciding what to review. If I feel a product is not worthwhile, I will not waste your time or this space reviewing it. For some software companies, this may be a good news-bad news scenario. The bad news is "we think your product stinks." The good news is "we aren't going to tell everyone." If I do not review some current item, it is either because I do not think it is well done, or I have not had the time to get to it yet. Next issue, I hope to show you what some real heavy-hitters, like Random House and others, are doing in educational software.

Before beginning the reviews, I would like to discuss an area of concern to me. We have all noticed the recent barrage of advertisements pushing the educational aspects of home computers. Most of these carry the message that if your child is struggling in school you should buy him or her a computer. We are made to feel guilty about our child's failure. What better way to assuage our guilt than to spend money?

Well, if a child is really having difficulty in school, the proper solution may be to spend more quality time with him or her. Many of the programs reviewed here may be of assistance to children, but they certainly will not solve these problems alone, and we should not expect them to. The development of children through adolescence offers an example of how we must take care in responding to problems. Acceptance by the social group in which he or she lives is a very dominant and personal drive for the adolescent. When deprived of this acceptance, there is often withdrawal into a shell.

A computer may act in a negative way, generating further isolation, a common, though mostly inaccurate, Hollywood characterization of teenage computer enthusiasts. However, if a computer can enhance the adolescents's self image by promoting success, whether with educational programs or otherwise, it may result in a much less tramatic period both for parent and child. We are responsible for much of what our children learn and how they learn it. Educational software is no exception. Now, off my soapbox and on to the reviews.

SAT PREPARATION SERIES
KRELL SOFTWARE
1320 Stony Brook Road
Stony Brook, New York 11790
48K Disk $299.95

Every winter, thousands of students take the "College Boards" hoping to improve their chances of college acceptance. Because of varying standards of grading in secondary schools, and even different grading techniques among teachers within the same school, college admissions committees have begun to rely heavily on the SAT scores. With greater competition at the entry level to college, success on this test is very important. This set of forty-two programs on ten disks from Krell is available to assist in the preparation for this test.

Of all the software I have reviewed, I feel the most comfortable in this arena. I have taken so many "Boards" (College, Medical entrance, National Medical, Pediatric, Neonatal-Perinatal, and recertification) that studying for this particular kind of exam has become second nature to me. I must say at the start that this series takes much of the drudgery out of the process.

The SAT is a test of general intellectual ability, not an achievement test. It is designed to measure verbal and mathematical abilities felt to be important for success in college. The scores in these two areas are used in the selection process by colleges. A measure of the ability to use standard written English is included in the test, but this is used for placement of students, not selection. There is also an additional part, either math or verbal, that is used for the writing of future tests and is usually quite different. This series of programs encompasses all of the areas included in the SAT.

Vocabulary.

Eight programs with a thousand questions each provide more than ample practice here. The format is the same as in the real exam; e.g., one is asked which of the following words is most nearly opposite in meaning of *inedible*: A. educe; B. increase; C. prototype; D. hale; E. comestible. After typing in the response, the correct answer is given and one's score is recorded. Although I did not go through all eight thousand examples, those that I did were challenging and frequently required use of a dictionary, a procedure suggested by the authors to enhance the learning process.

Word relationships.

Again, from one to one thousand questions in three programs test proficiency in this area. An example follows:

1. Which pair of words most nearly bear the same relationship to each other as:
fuel:gasoline

A. treason:ruler
B. behavior:psychologist
C. orthodoxy:heretic
D. disease:smallpox
E. insects:entomologist

After entering one's answer, the correct one is displayed and the relationship is presented. In the above example, the answer is "D", and the relationship, set:subset. A wide variety of relationships from synonyms to different verb tenses are used. This is a very difficult type of question for many and practice here will be quite beneficial.

Sentence completion.

Here are two programs which seek to find the word or word pair best used in the sentence shown. Example:

1. Because it was _____ the work was _____ by the critics.
A. vindictive — dismayed
B. opulent — denounced
C. implicit — decried
D. enigmatic — rebuked
E. reprehensible — bated

Another difficult type for some people and, as in the previous section, this exercise lends itself well to computer assisted instruction.

Reading comprehension.

This set requires use of an accompanying booklet which contains fourteen reading selections. There are two programs, and the questions are presented in groups of fifteen each. Each question begins by referencing the reading selection in the booklet. Then, one is asked to select one of five multiple choice answers to a question, such as:

The author states or implies all of the following except:

or,

The author's views on this issue cannot be determined from the reading.

In general, this type of question is not geared for the computer as the screen fills up quickly with the responses. It is a well done section, but skipping back and forth from booklet to screen is somewhat tiresome.

Math.

The mathematics portion of the SAT tests one's understanding of the following:

Fractions
Conversion of Units
Ratio and Proportion
Averages
Interest and Percent
Time and Work
Rate, Time, and Distance
Series
Decimals

Algebra
Geometry
Graphs, Charts, and Tables

No advanced knowledge in Algebra, Solid Geometry, Trigonometry or Calculus is required. This group of fifteen programs provides sufficient material for review and practice. I was amazed at how much I had forgotten as I began to review this section. The format used was extremely helpful in reacquiring atrophied skills.

One thousand examples in each program encompass the mathematical concepts mentioned above. Use of paper and pencil to draw diagrams and perform calculations, as with the actual test, is necessary. No extraneous materials are permitted while sitting for the SAT. This includes notes, slide rules (the sun dial of calculators) or calculators. This opportunity should be used to practice calculating without mechanical assistance. Time is a factor in the test and dependence on calulators must be eliminated. The questions are presented much in the same way as the others, with a notable exception. After being given the correct answer, one is given the option to see the solution. Unlike many other preparatory aids, where one must leaf through the back of the book for the answers and through another section for examples of solutions, the solution to the problem in question is presented immediately. The solutions are straightforward and clear. One is then given the opportunity to immediately select another problem of the same type. The same kind of problem can be repeated until its solution has been learned. After just a few examples in each area, I was back in stride with what used to be my best subject. I am quite sure that students will find this section of great assistance in reviewing and enhancing their mathematical abilities.

Test of standard written English.

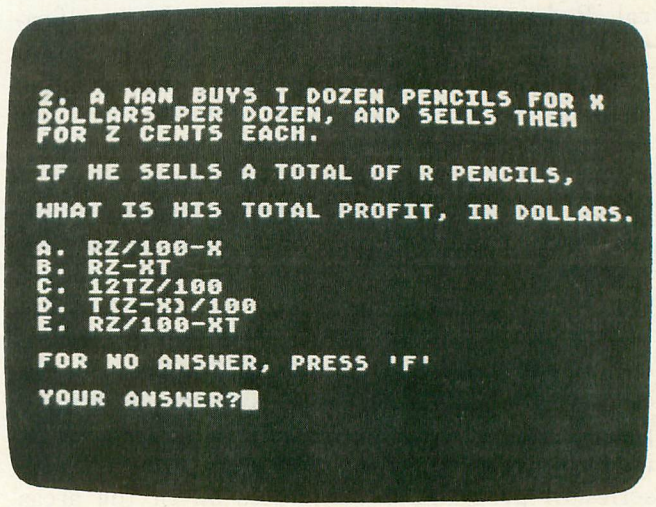
As mentioned previously, this section is not included in the overall score, but is utilized for placement. A good performance here may avoid having to take "Bonehead English" or whatever it is called now. There are twelve programs offering the student the opportunity to practice the recognition of the following:

Pronoun errors
Verb tense or mood errors
Diction errors
Sentence fragments
Dangling modifers
Completeness and consistency errors
Punctuation errors
Parallelism errors

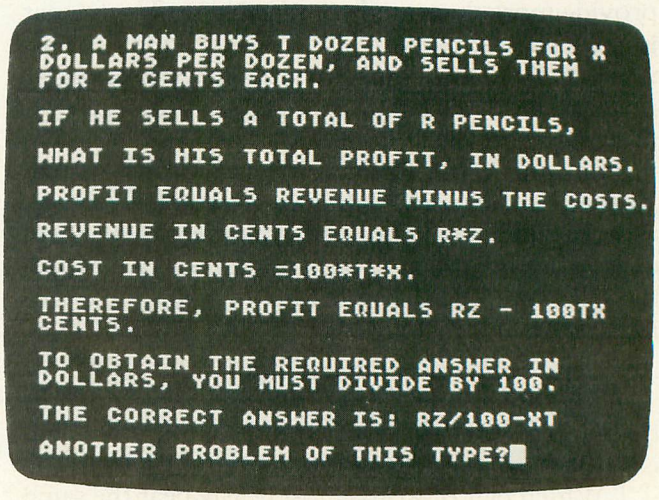
The first six lessons each consist of fifteen sentences in which there are four underlined and lettered (A-D) parts. If there is an error of standard written English in any underlined part, the appropriate letter is entered. There is no more than one error per sentence, or the sentence may be entirely correct. After answering, the correct response is supplied with an explanation and the error cited. Using *whom* instead of *who* might elicit the following:

The sentence should read "...who the man is."
"Who" is a subjective complement and thus in the subjective case.
****Pronoun Error****

In the second six programs, one is presented a long paragraph or series of paragraphs continuing the same story. One potential error at a time is presented and three words or phrases are offered as correct substitutes. The context of the story is quite important in these examples. Both parts of this section are well done and instructive. As may be obvious from my writing, I did well in math and not so well in standards of written English. This is a good review, whether you are taking the Boards or not.



Questions on Math.



Answers on Math.

Scoring.

The questions are scored in similar fashion to the SAT. Four points are awarded for a correct answer, none for an unanswered question, and one point is deducted for an incorrect response. After each question, a display of the total score, correct answers, errors, and unanswered questions is shown.

An additional feature may be used with the programs in Math and Vocabulary. If this feature is selected, Learning Coefficients are generated based on previously demonstrated success and are saved to the disk. Subsequently, problems on which there has been difficulty will be repeated. Conversely, those areas where the student has demonstrated success will be encountered less frequently.

Options.

A number of options are said to be available for additional cost. It is not specified how to obtain these, what the cost, or if they are available for every computer. They include a Worksheet Generator, which may be used with a printer to create worksheets, a Vocabulary Builder, and a Classroom Record Management System. Another seemingly useful option is the Diagnosis and Study Prescription which keeps track of the user's performance by skill types.

Is it worth $300?

This SAT preparation series is well done. The accompanying documentation includes sample worksheets and the reading selections mentioned. It is easy to use, and requires no special instructions. I compared it to a number of the readily available workbooks filling the shelves of bookstores. Basically, the same material is covered and, although the test format more closely resembles the books (it is on paper) it seems much easier to review using the computer. The quick response of the computer makes it much less tiresome and the immediate reward of a correct answer is encouraging. There is no doubt that much more material can be covered in a shorter period of time using this program than in any other way. Interestingly enough, a local TV station has just begun a series on preparation for the SAT. At the expense of missing a rerun of *Gilligan's Island*, I tuned in. Preparing for this test is more a matter of review than a new learning process, and an instructor slows the pace down. Attention spans shorten when an area is being covered that is known, and to suddenly kick it into gear when something new comes up is tough. The computer permits one to go at the pace desired.

A Limited Warranty is provided which guarantees a gain in score of seventy points (combined verbal and math), or the purchase price will be refunded. Transcripts of previous tests, proof of purchase, a statement certifying that the materials were utilized for a minimum of six hours, residency in North America, and the return of all materials are among some of the conditions necessary to receive the refund. This is probably not too bold an offer, since studying any of the available aids would most likely achieve similar results.

You may have noticed that I have been avoiding the question stated above. Is it worth the price? This is obviously an individual decision. This series of programs is of top quality and will certainly be of enormous assistance to anyone preparing for the SAT. It is easy to use and even makes the review process pleasurable. Spending this much money may ensure that the studying is done, just out of guilt. More importantly, if a high score is desired for admission to a particular school, or a higher score is sought after a mediocre performance, it may very well be worth the investment. Frequently, the stiffer the academic requirements, the higher the tuition. If one is willing to pay the higher tuition, the price of this package may be very reasonable. Krell Software has provided the Atari owner with a superb educational tool. I hope other producers of educational software emulate these high standards. □

MAGIC STORYBOOK
AMULET ENTERPRISES, Inc.
P.O. Box 25612
Garfield Heights, OH 44125
16K Cassette $29.95

by Tony Messina

Programs which utilize the Atari computer along with the cassette drive to produce music and instruction have been around for quite a while. **Moonbase Io** and the Atari language series are but a few examples. **Magic Storybook**, a new program from Amulet Enterprises, combines both recorded music and hi-res graphics to tell one of my favorite children's stories, "The Three Little Pigs."

The program.

Loading the program cassette was not a problem. The screen was soon filled with a **Magic Storybook** castle logo, and the program continued to load with music playing in the background. Finally, a voice announced the story and we were off.

I admit that sitting through the computerized edition of this story seemed a bit strange, and I did get a few heads popping into my office with strange looks. I was impressed with it nonetheless. The scrolling screens, color, music and animation of the pigs and "Wasco the Wolf" were all done very nicely. I wasn't impressed to the point of sitting through another replay, however.

It was at this point that I decided to call in the Newport Software Test Conglomerate. This includes my niece, her friends and the neighborhood kids. After they had sat through the program, they wanted to see it again...and still again...and again until I said "OK, I guess you really like this, huh?" A thundering "YES!!" was heard through the entire street. From the kids' point of view, the program was really great and, after all, **Magic Storybook** is aimed at the kids. Nevertheless, there were a couple of features that the kids themselves wished the program had.

What the kids wanted.

My niece wished that Wasco could be moved all around the screen. She wanted to play with Wasco to see if she could sneak him in the window, or around the back of the house. Her friends wanted to control the pigs to help them build their house. When asked if the story was too short, the majority of the kids said it was fine.

What I wanted.

The only major item that could be re-done would be to get a more enthusiastic narrator. Whoever narrated the soundtrack sounds like she was bored to death. As a matter of fact, the narrator got a D- from the kids when they answered the "How did you like the storyteller, A-F?" question.

The verdict.

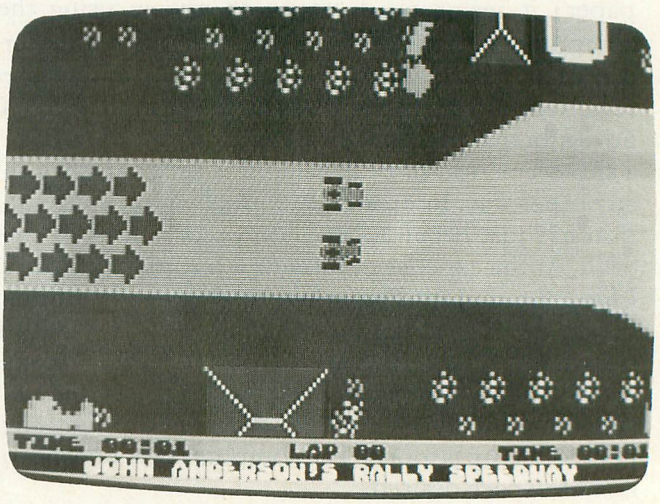
Overall, this program rated very high with the kids, as evidenced by the fact that I couldn't drag them away from the computer. The program could use an upgrade in the narration department, but otherwise **Magic Storybook** is certainly worthwhile and will keep your kids entertained until you can't stand it anymore! □

RALLY SPEEDWAY
by John Anderson
ADVENTURE INTERNATIONAL
P.O. Box 3435
Longwood, FL 32750
16K Cartridge $49.95

by Lee Pappas

Rally Speedway is a one or two player, 16K cartridge where you have a birds-eye view of a small portion of a large scrolling screen. In the one player version, your race car remains in the center of the screen as you negotiate hairpin turns and scream down straightaways. Should you "miss" a turn you'll likely pile-up into a tree, building, or someone's swimming pool, with the poor driver frantically rolling on the ground to extinguish his flames.

When the game first "boots up," you can select an option list allowing you to tailor the game to your own tastes and abilities. From here you select between a one or two player game, dry, wet, or icy roads, or speeds ranging from 40 to 100 MPH in 20 MPH increments. The player can also choose slow,



Rally Speedway.

normal or fast acceleration, and real life or *only in a computer* jaunts. *Only in a computer* means your

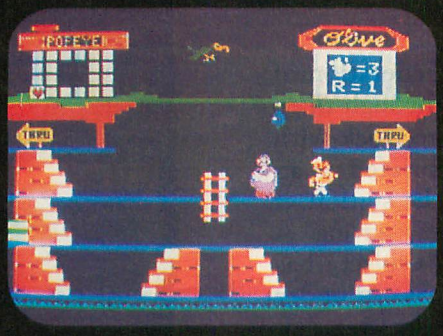
(continued on page 130)

ATARI 5200

ATARI 400

ATARI 800

ATARI 600XL



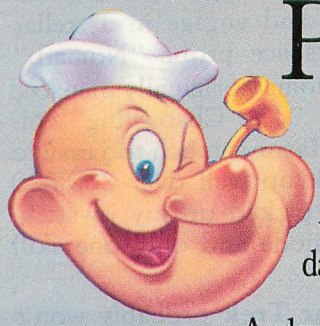
TI99/4A

ATARI 2600

INTELLIVISION

COLECOVISION

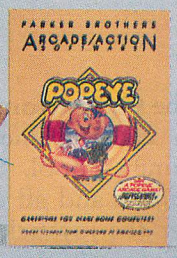
EVERYONE HAS THEIR OWN SYSTEM FOR PLAYING POPEYE.®



Atari.® Intellivision.® ColecoVision.™ T.I.™ Now you can play POPEYE, one of the most fun and challenging arcade games yet, on any one of them. Run through three screens of non-stop action, where you try to capture Olive Oyl's heart while avoiding untold dangers, including Brutus and the Sea Hag.

Run down to your local store for Popeye today. And while you're there, check out TUTANKHAM,™ FROGGER,™ Q*bert™ and SUPER COBRA,™ also from Parker Brothers' Arcade Action Series.

®PARKER BROTHERS



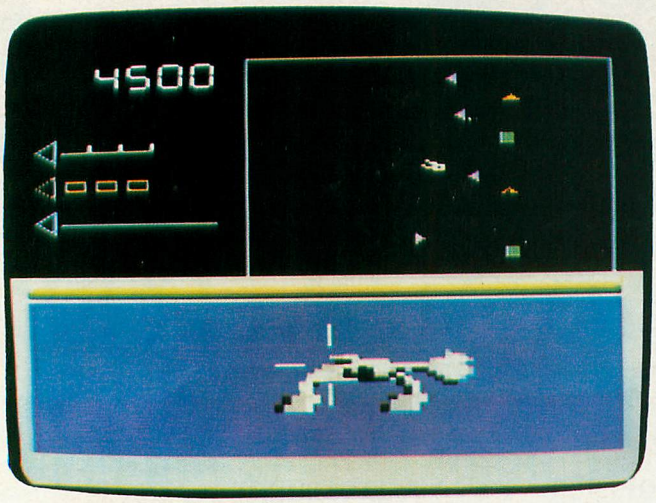
STAR TREK: STRATEGIC OPERATIONS SIMULATOR
SEGA CONSUMER PRODUCTS
5555 Melrose Avenue
Los Angeles, California 90038
16K Cartridge $39.95

by Patrick J. Kelley

It is the 23rd Century.

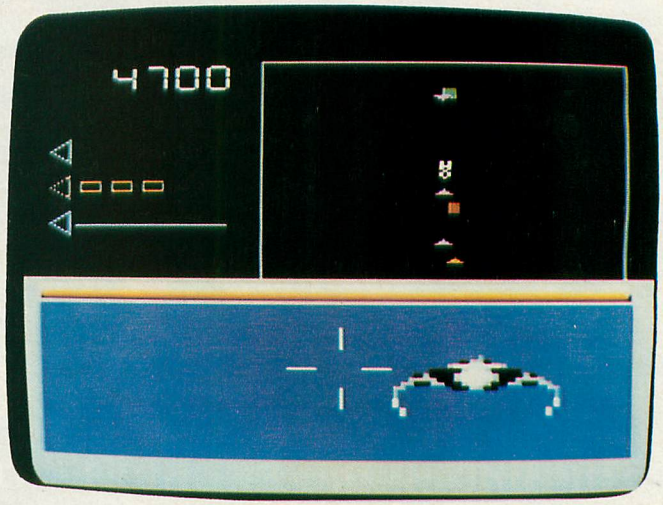
You are at the helm of the United Federation of Planets' Cruiser *Enterprise*, traveling through deep space. You have just received orders to dock with one of the Federation's starbases to be briefed on an upcoming mission. As you approach the station, it becomes obvious that something is very wrong. A score of Klingon Katinga-Class heavy cruisers has descended from warp space; the starbase is under heavy bombardment. You must make a decision that could spell intergalactic war on one hand, or the death of hundreds of innocent persons aboard the station on the other. Grimly, you prepare yourself and your ship to do battle with the Klingon invaders.

If this situation sounds like an outline for the next *Star Trek* film, you may be half right. Sega's new **Star Trek: Strategic Operations Simulator** takes many familiar elements from the *Trek* films and TV series and combines them into an arcade-style action game.



Star Trek.

The **Operations Simulator** puts you in the shoes of a cadet trainee at the Federation Academy's training school. You are in command of the *Enterprise* and all of its defensive weaponry. Using either the joystick or the keyboard, you must lead the Enterprise into the thick of battle, using photon torpedoes and phasers to combat the Klingons and save your beleaguered starbases. You also have the ability to use warp speed, a feature that allows you to outdistance your Klingon adversaries, or to speed to the defense of an overwhelmed starbase.



Klingon ship.

The display screen is divided into three parts. The lower third represents the *Enterprise's* bridge viewer. The upper right third is a bird's-eye scanner that shows your relationship with the Klingons, starbases and other nearby objects. The remaining third is a bar-graph display that indicates your remaining photon torpedoes, warp energy, and the power remaining in your deflector screens. This particular gauge is one to watch, because if your shields become depleted, a disruptor-torpedo from a Klingon cruiser will mash you and your ship to jelly.

To be a seasoned Atari space-gamer, **Star Trek** may sound a lot like **Star Raiders** (a classic worth aspiring to). Purists will shake their heads and say "The first is always the best," and in this case I must agree with them. Although I am a big fan of Sega's **Star Trek** arcade game, this home version does not live up to the promise of its big brother. Game play is gratifyingly fast as you blast Klingons, dock with starbases, avoid asteroids and engage in a stellar dogfight with the killer space probe "Nomad." (*Trek* fans will recall that Nomad originally menaced menaced the *Enterprise* in "The Changeling" episode.) But the home **Simulator** is much too easy to master. I was able to rack up well over 200,000 points the second time I picked up my joystick. There is no multiplayer mode and only one skill level.

This incarnation of **Star Trek** probably won't impress a hard-core Atari computer gamer. Its lack of challenge would soon relegate it to the "I'm Very Bored" pile of cartridges. As for myself, it will certainly be a long time before my next voyage on Sega's Starship *Enterprise*. □

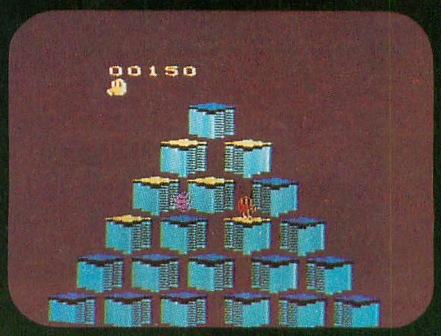
ATARI 5200

TI99/4A

ATARI 400/800/600XL

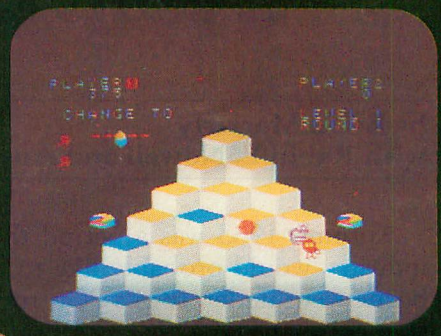
INTELLIVISION

COMMODORE VIC 20



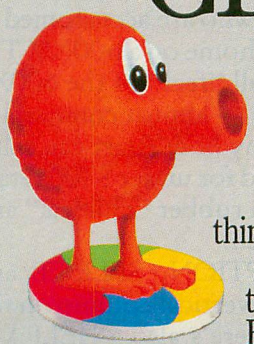
ATARI 2600

COMMODORE 64



COLECOVISION

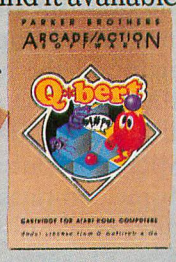
HOW TO GET Q*BERT™ OUT OF YOUR SYSTEM.

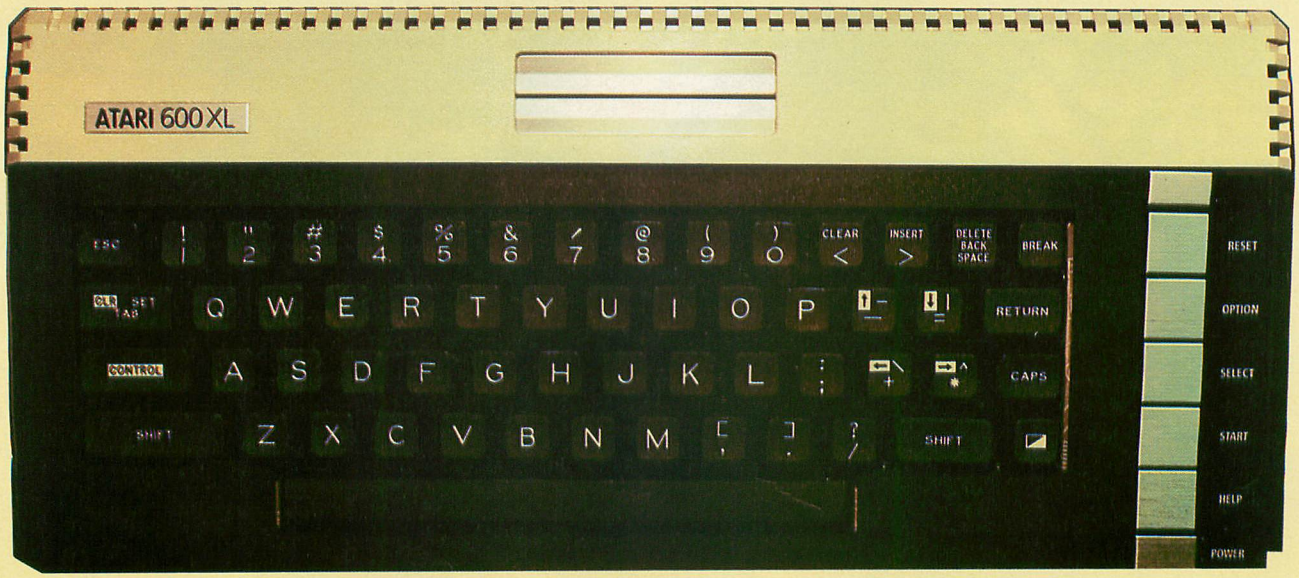


If you've been wanting to play Q*bert, but haven't been able to find it available for your home system, your time has come. Because now you can keep things hopping with any of these popular home video and computer formats.

Get going to your nearest video store and get Q*bert today. And while you're there, check out Parker Brothers' POPEYE,® FROGGER,™ TUTANKHAM,™ and SUPER COBRA.™ All the great Arcade Action® games, now in all the great home formats.

PARKER BROTHERS





INSIDE THE ATARI 600XL

by Brian Moriarty
with Robin E. Nowell and Austin Franklin

Autumn 1983 was a period the microcomputer industry would probably like to forget about. Osborne went under. IBM's PCjr turned out to be the best computer of 1978. TI threw the market into a feeding frenzy by dumping thousands of obsolete 99/4As at $49 each. And Atari delighted the hostile press by reporting massive operating losses, selling 400s and 800s at bargain-basement prices, putting the 1400, 1450 and 1600XL projects on "indefinite hold" and failing to keep up with Christmas orders for the 600 and 800XLs.

As this article goes to press in late November, the 800XL has yet to appear on the shelves of Massachusetts retail outlets. The 600XL is available, but quantities are so limited that it's almost impossible to obtain. We did manage to snare a single unit at a nearby Toys-R-Us store. (Woe for the days when Ataris were sold at real computer stores, by people who knew what they were talking about.) We pulled the 600 out of its surprisingly small carton with a mixture of hope and fear. Would the new machine maintain the high engineering standards set by the 400 and 800 systems? Or was it a rework of the ill-fated 1200XL, destined for a glass case in the Microcomputer Hall of Shame? The fate of Atari's home computer division (and yes, even of our magazine) could ultimately depend on the success of the new XL series. So we tore the beast to pieces and puzzled over its entrails.

On the outside.

The 600XL weighs about five pounds and measures 6½ x 15½ x 2½ inches, no bigger than an 800 from the front edge to the back of the cartridge slots. It feels encouragingly substantial when you pick it up. The cosmetics are superb; beige and dark chocolate with brushed aluminum accents. The Commodore 64 and Tandy CoCo look like toys by comparison.

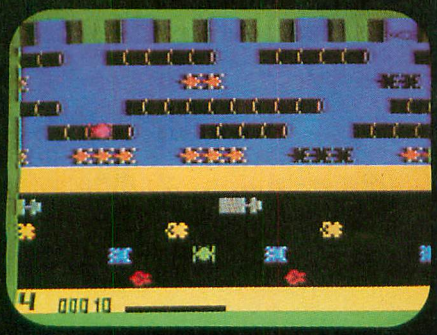
The full-stroke keyboard consists of 57 plastic keys mounted on springs, which in turn depress a flat flexi-circuit membrane. The entire assembly is backed by a solid metal plate to make it feel solid as a rock. Atari wisely abandoned the clumsy layout of the 1200XL keyboard; users who are accustomed to a 400 or 800 will feel right at home on a 600XL. The height, positioning and overall "feel" of the 600XL keyboard are very, very good — almost comparable to the old 800 systems. Funny how Atari can sell a unit with such a nice keyboard for under $200 retail, while IBM's PCjr comes with rubber "chiclets" and costs over three times as much.

The familiar RESET, OPTION, SELECT and START keys are located in a vertical column on the right edge of the console. There's also a HELP key which you can read by PEEKing location 732 decimal ($2DC hex), and an LED power-on indicator. The RESET key is backed by an extra-hard spring to make it difficult to press by accident. If you

ATARI 5200

TI99/4A

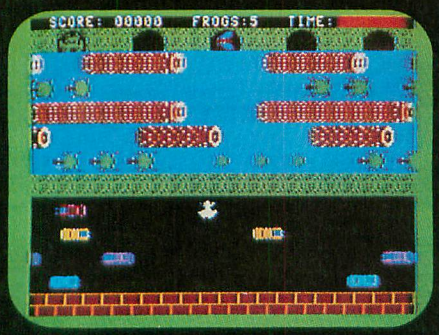
ATARI 400/800/600XL



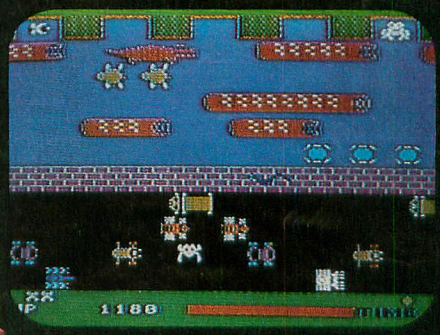
INTELLIVISION

COMMODORE VIC 20

ATARI 2600



COMMODORE 64

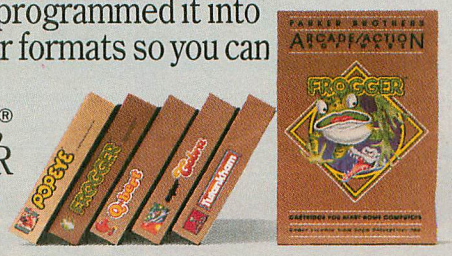


COLECOVISION

8 WAYS YOU CAN PLAY FROGGER*™ AT YOUR PAD.

FROGGER is one of the all-time great award-winning home video games. And now Parker Brothers has programmed it into all the most popular video and computer formats so you can keep things hopping in your own home.

Catch Frogger along with POPEYE,® Q*bert,™ TUTANKHAM™ and SUPER COBRA™ where you buy your video and computer games. You'll find it absolutely ribbitting. ®PARKER BROTHERS



Books&Software

BOOKS for ATARI 400/600XL/800XL 1200XL

ATARI BASIC – Learning by Using
An excellent book for the beginner. Many short programs and learning exercises. All important features of the ATARI computers are described (screen drawings, special sounds, keys, paddles, joysticks, specialized screen routines, graphics, sound applications, peeks, pokes, and special stuff).
Order-No. 164 $7.95

Games for the ATARI Computer
This book describes advanced programming techniques like player-missile-graphics and use of the hardware-registers. Contains many ready to run programs in BASIC and one called GUNFIGHT in machine language.
Order-No. 162 $7.95

How to program your ATARI in 6502 Machine Language
Introduction to machine language for the BASIC programmer.
Order-No. 169 $9.95

FORTH on the ATARI – Learning by Using
Introduction, programs, applications, learning exercises.
Order-No. 170 $7.95

All programs from book No. 170 on disk.
Order-No. 7319 $22.00 only!

A Look into the Future–ASTROLOGY on your ATARI 800. How to calculate your own horoscope. Including listing of the program.
Order-No. 171 $9.95

HACKERBOOK for your Atari computer
Tips+tricks–Very important subroutines in 6502 machine language. How to make bootable cassettes, disks, and EPROMs. Complete construction article and software on how to build an EPROM burner.
Order-No. 172 $9.95

SMALL BUSINESS SOFTWARE FOR ALL ATARI COMPUTERS

SUPERMAIL (500 addresses on 1 disk) Completely written in FORTH. Comes on autoboot disk. No cartridge, no DOS, no FORTH language required.
Order-No. 7312 $49.00

SUPERINVENTORY (1000 items p.disk) Completely written in FORTH. Same as above. (Disk only)
Order-No. 7320 $49.00

BUSIPACK-1 (written in FORTH). Complete order entry, inventory, mailing and invoicing. (Disk only).
Order-No. 7313 $98.00

Microcomputer Hardware Handbook
Order-No. 29 $ 14.95

ATCASH
Convert your ATARI 800 into a powerful cash register. (Disk only).
Order-No. 7303 $49.95

Invoicing program in BASIC
Order-No. 7201 (C) $29.95
Order-No. 7200 (D) $39.95

Mailing List in BASIC
Order-No. 7212 (C) $19.95
Order-No. 7213 (D) $24.95

Inventory control in BASIC
Order-No. 7214 (C) $19.95
Order-No. 7215 (D) $24.95

SOFTWARE IN MACHINE LANGUAGE FOR ATARI

ATMONA-1
Machine language monitor.
Order-No. 7022 (C) $19.95

ATMONA-2
This is a tracer (debugger) that lets you explore the ATARI RAM/ROM area. You can stop at previously selected address, opcode, or operand. Also very valuable in understanding the microprocessor. At each stop, all registers of the CPU may be changed. Includes Atmona-1.
Order-No. 7049 cassette $49.95
Order-No. 7050 disk $54.00

ATMAS
Macro-Assembler for ATARI-800/48K. One of the most powerful editor assemblers on the market. Versatile editor with scrolling. Up to 17k of source-code. Very fast, translates 5k source-code in about 5 seconds. Source code can be saved on disk or cassette (Incl. Atmona-1).
Order-No. 7099 disk $89.00
Order-No. 7999 cartridge $129.00

ATMS APPLICATION DISK
All programs and machine language subroutines from Book No.169 on disk.
Order-No. 7311 $20.00

ATAS
Same as ATMAS but without macro-capability. (32K and 48K RAM)
Order-No. 7098 $49.95

ATEXT-1
This wordprocessor is an excellent buy for your money. It features screen oriented editing, scrolling, string search (even nested), left and right margin justification. Over 30 commands. Text can be saved on disk or cassette.
Order-No. 7210 cassette $29.95
Order-No. 7216 disk $34.95
Order-No. 7217 cartridge $69.00

FORTH for the ATARI
FORTH from ELCOMP PUBLISHING, Inc. is an extended Fig-Forth-version, Editor and I/O package included. Utility package includes decompiler, sector copy, Hex-dump (ASCII), ATARI filehandling, total graphic and sound, joystick program and player missile. Extremely powerful!
Order-No. 7055 (D) reg. $99.00 sale $39.95

Floating point package with trigonometric functions (0 – 90°).
Order-No. 7230 disk $29.95

Learn FORTH
A subset of Fig-Forth for the beginner. On disk (32k RAM) or cass. (16k RAM).
Order-No. 7053 reg. 79.00 sale $19.95

HARDWARE-ADD-ONs for ATARI

PRINTER INTERFACE
This construction article comes with printed circuit board and software. You can use the EPSON printer without the ATARI printer interface. (Works with gameport 3 and 4).
Order-No. 7211 $19.95

300 Baud serial interface (RS232+5V)
Software with connector and construction article.
Order-No. 7291 $19.95

EPROM BURNER for ATARI 400/800 KIT. Printed circuit board incl. software and extensive construction article.
Order-No. 7292 $49.00

EPROM BOARD (CARTRIDGE)
Holds two 4k EPROMs (2532). EPROMs not included.
Order-No. 7043 $29.95

EPROM BOARD KIT
Same as above but bare board only with description.
Order-No. 7224 $14.95

Astrology and Biorhythm for ATARI
Order-Nol 7223 D/C $29.95

Birth control with the Atari (Knaus Ogino)
Order-No. 7222 disk only! $29.95

The APPLE in your Hand, BRAND NEW! Book includes introduction to 6502 Machine Language and FORTH. BASIC programs never published before!
Order-No. 178 (200 pages) $12.95

CP/M –MBASIC and the OSBORNE computer
Business Applications, compl. listings of mailing list, datablock, inventory, invoicing and more.
Order-No. 177 $9.95

SUPERSOFTWARE f.the Commodore-64
BLIZTEXT! – The best wordprocessor for the C-64 in the whole universe. Includes terminal software for electronic mail.
Order-No. 4965 (62 pages manual) $89.00

MACROFIRE – Editor/Assembler for the C-64
The best macroassembler you can buy!
Order-No. 4963 $89.00

SUPERBOOKS for your C-64

The Great Book of Games, Vol. I
64 programs for the Commodore-64.
Order-No. 182 $9.95
Programs from this book on disk $19.95

MORE ON THE SIXTYFOUR
Tips, tricks, hints, very important subroutines.
Order-No. 182 $9.95
Programs from this book on disk $19.95

How to program in 6502 Machine Language on your C-64
Order-No. 184 $12.95

Commodore-64 Tune-up, Vol. I
How to expand your C-64.
Order-No. 185 $12.95

Small Business Programs for the C-64
Order-No. 186 $12.95

HARDWARE ADD-ONs for your C-64

Parallel printer interface, KIT
Order-No. 4990 $19.95

Universal Experimenter Board
Order-No. 4970 $9.95

Expansion Board (holds 4 exp. boards)
Order-No. 4992 $29.95

BOOKS, SOFTWARE, ADD-ONs for VIC-20, APPLE II, OSBORNE, TIMEX + OSI computers

Tricks for VICs (Book)
Order-No. 176 $9.95

Universal Experim.board for the VIC-20
(Save money with this great board). This board plugs right into the expansion slot of the VIC-20.
Order-No. 4844 reg. $19.95 sale $9.95

Programming in BASIC and machine language on the ZX-81 (82) TIMEX1000
Order-No. 174 (book) $9.95

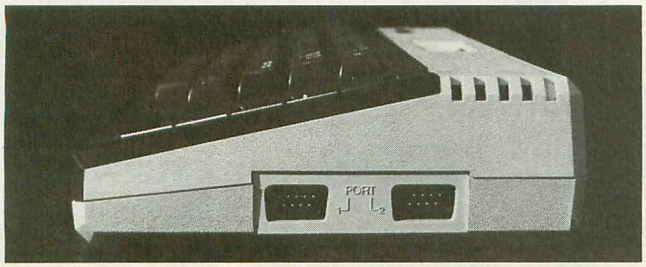
The Custom APPLE + Other Mysteries
A complete guide to customizing the APPLE software and hardware.
Order-No. 680 $24.95
We also stock the boards which are used in the book No. 680 (bareboards).

HOFACKER

Elcomp Publishing, Inc, 53 Redrock Lane, Pomona CA 91766, Phone: (714) 623-8314, Telex: 29 81 91

PAYMENT: Check, money order, VISA, MASTER CARD, Eurocheck, ACCESS, Interbank
Prepaid orders add $3.50 for shipping (USA)
$5.00 handling for C.O.D.
All orders outside USA: add 15% shipping, California residents add 6.5 % sales tax.

do press RESET (accidentally or otherwise), a true hardware reset is performed instead of the pseudo-reset of the 400 and 800 systems. That means you should be able to get out of even the nastiest lock-up without having to shut off the console.

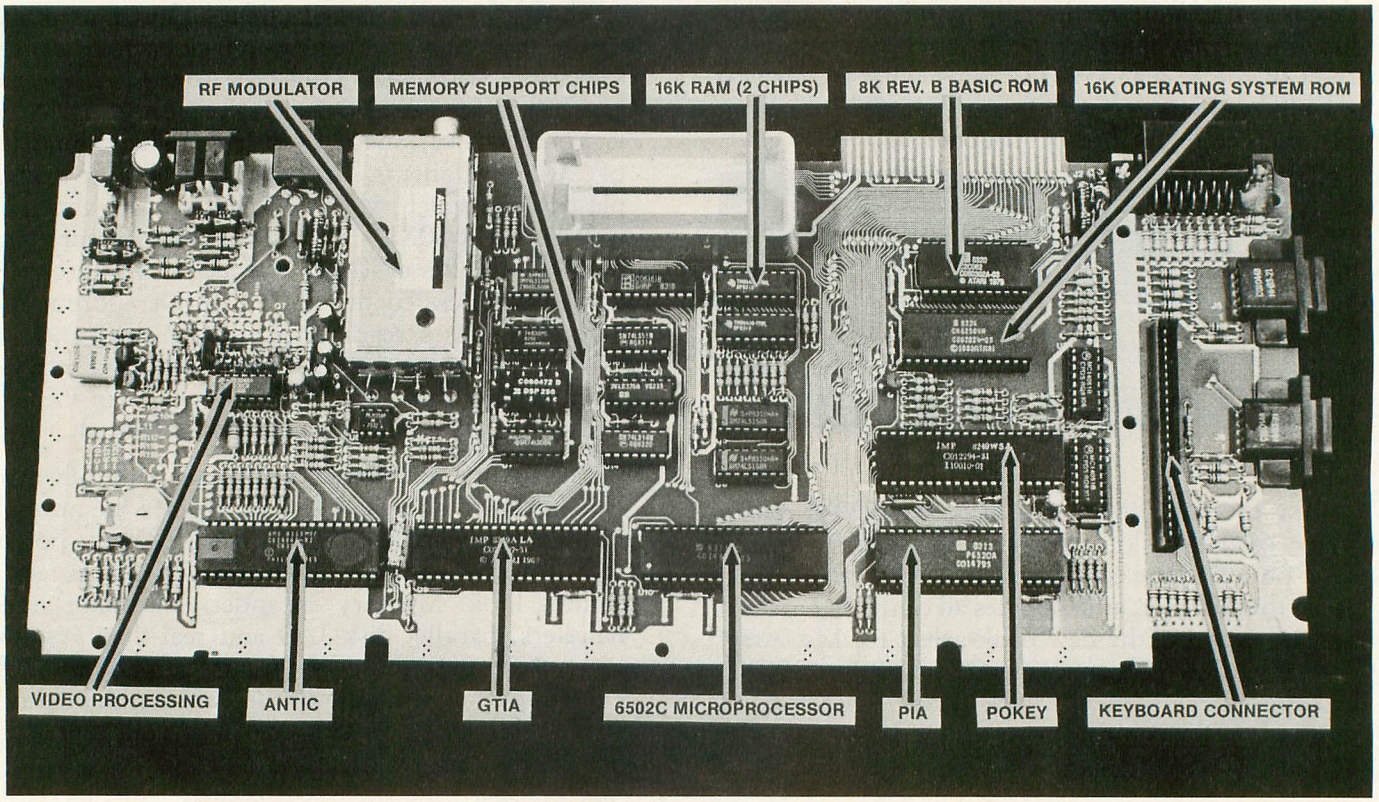


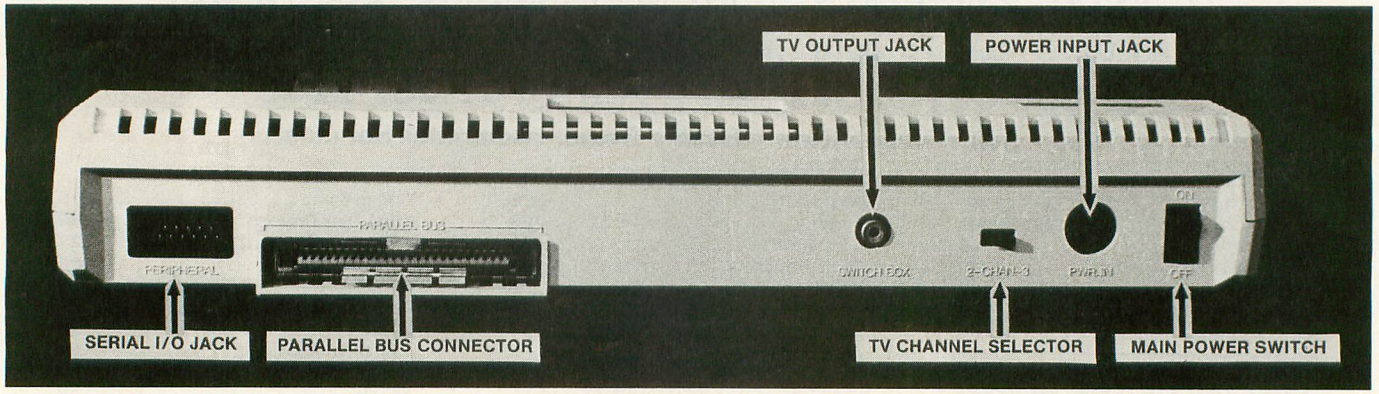
Side view

Like the venerable Atari 400, the 600XL has a single cartridge slot, now located under a hinged door on top of the console. It's electronically identical to the 400 slot, and includes all of the signal lines necessary to use the new bank-select ROMs. Atari continues the questionable tradition of the 1200XL by supplying only two joystick ports, located on the right side of the console. So much for 4-way games of **M.U.L.E.** or **Ali Baba**.

The back panel has the standard 13-pin serial I/O connector, an RF video output jack, TV channel selector (2/3), a 5-pin DIN-type power jack and the main on/off switch. No composite monitor hook-up is provided, and none is possible without adding a lot of extra components. The back panel also has a small plastic door which, when removed, reveals what may well be the 600XL's most promising feature: the parallel bus. It's a 50-pin connector that provides complete access to the 6502's 16-bit address and 8-bit data busses, read/write, phase 2 clock and status lines. There's also an audio input and various other signal lines that allow you to control the system's memory configuration from off-board.

The XL bus connector actually makes the 600 *more* expandable than a 400 or 800. It gives the new XL line the hardware flexibility it needs to make potential Apple IIe and Commodore 64 customers take a serious look. Memory expanders, alternate video handlers, parallel disk I/O and real-time systems control are just a few of the fascinating possibilities. Let's hope Atari and the third-party suppliers don't wait too long to come out with peripherals that make use of this long-overdue option.





On the inside.

All of the ICs, connectors and video circuitry are mounted on a single printed circuit board, which is enclosed in a vented metal chassis to reduce RF interference. To save space, Atari moved the AC line toroid out of the console and into the external power supply module, which is a little larger and heavier than its 400/800 counterpart.

The 600XL comes factory-equipped with 16K of RAM, implemented by a pair of 4416 8K x 8-bit ICs and a handful of support chips. Noise-reducing resistors and precision delay lines in the timing network make the 600XL memory circuits quieter and more reliable than the 400/800 systems. All of the chips are mounted in sockets for fast, easy servicing.

The keyboard is tied to the mother board with a flexible ribbon-type edge connector. The connection is relatively fragile; half of the contacts on our sample wore away after six plug/unplug cycles. This is the only significant wear point we could find on the unit.

The 600XL's 16K operating system is contained in a single 28-pin mask-programmed ROM. Contrary to popular belief, the 600XL OS is *not* based on the old and comfortably compatible 400/800 OS. It is a direct descendant of the 1200XL OS, incorporating many of the silly, memory-wasting bells and whistles that helped send its creators to the unemployment lines. Chief among these is the Self-Test Mode, a series of three routines that let you "check the computer's memory, sound, colors and keyboard to ensure that they are working properly," to quote the

Owner's Guide. It sounds great on paper, but in practice the routines constitute little more than a sales gimmick.

The Memory Test fails to inform you when it's done testing. It just keeps cycling. The simplistic Audio-Visual Test doesn't indicate which colors you're supposed to be seeing, and never turns on more than one sound channel at a time. Most annoying (and revealing) is the Keyboard Test, which displays a key pattern that does not match the layout on the 600XL console. Instead, it presents a totally different-looking keyboard that old-timers will immediately recognize as belonging to the 1200XL!

The 600XL's OS is also saddled with the foreign character set found in the 1200 system. This feature (which is not supported by any currently available word processor, including Atari's own **AtariWriter**) provides the oh-so-exciting ability to display many of those strangely-accented characters you've been dying to include in your letters to Aunt Matilda. Just POKE location 756 with 204 and amaze your friends.

We wouldn't gripe so much about the extra character set and Self-Test Mode if they didn't eat up so much space in the ROM chip. The Self-Test takes up about 2K (2,048 bytes); the funny characters, another K (1,024 bytes). Why not use those three thousand bytes to implement a nice little machine-language monitor, similar to the one supplied in every Apple II system? Or how about plain English error messages instead of meaningless numbers?

On the positive side, the 600XL OS eliminates the need for a separate BASIC cartridge by including the language right inside the console. It resides in an 8K ROM chip at location $A000, and is automatically activated whenever you turn on the computer unless you hold down the OPTION key while powering up.

The 600XL marks the first appearance of the long-awaited "Revision B" version of the Atari BASIC. Most of the bugs present in the original cartridge have supposedly been eliminated, including the notorious editing lockup problem. It remains to be seen whether Atari's fixes will cause any new bugs to appear; we've heard rumors about suspicious behavior in the ENTER and LOAD commands, among others. Let us know about anything you discover. Unfortunately, the changes wrought in Rev. B make Bill Wilkinson's *Atari BASIC Sourcebook* totally obsolete as far as the source listing is concerned.

Is the 600XL compatible with software written for the 400/800 OS? Our answer is a highly qualified yes. None of the "guaranteed" OS vectors have been changed in the new XL OS, so software using those vectors (and *only* those vectors) has a fair chance of working on the 600. Nevertheless, a lot of good stuff written for the 400/800 series makes use of undocumented ROM routines. Such programs will almost certainly crash on the new XL machines. Software publishers aren't likely to admit how their products misuse the OS; so the only foolproof way to find out if a particular piece of software is compatible is to try it out yourself.

Accessories, options, opinions.

The documentation provided with the 600XL computer is inadequate and frankly disappointing. The first third of the 20-page *Owner's Guide* tells you how to hook the thing up and turn it on; most of the remainder is devoted to thinly-disguised advertisements for all the wonderful Atari peripherals you didn't know you were going to need. Instead of the 120-page *Atari BASIC Reference Manual*, you get a 14-page pamphlet of BASIC keywords ("For Experienced Programmers") which contains a list of books you should consult in the apparently unlikely event that you want to learn more about your $200 investment. Another little pamphlet explains how you can buy a 1-year service contract for just $34.95. Also hidden in the styrofoam are the power supply module, the TV switchbox and a warranty card. And that's all, folks.

Atari has announced a Memory Expansion Module for the 600XL that will connect to the parallel bus connector on the back of the unit. This $100(?) device will expand the 600 to its full effective RAM capacity of 48K. An expanded 600XL should be functionally identical to an 800XL, except for the monitor output. None of our local stores is carrying the Expansion Module yet, and nobody knows when to expect it. Look for a report in the pages of **ANALOG** when and if it appears.

Our first impression of the Atari 600XL system is mixed but mostly optimistic. The hardware is nicely engineered and solidly constructed. The operating system is flabby but serviceable; having a bug-free BASIC on board is a welcome relief. Whoever made the decisions about documentation ought to be ashamed of themselves. We'd rather see the cost of the unit go up by five or ten bucks if it will buy a reference manual that won't shortchange beginners or insult the experienced user. An XL Series hardware manual and OS listing would also be welcome options in the near future.

The 600XL isn't as exciting or advanced as the 400/800 series was when it first appeared in 1979. Let's face it, these new XL machines are nothing more than repackaged 800s. That does not change the fact that Atari home computers are still the most versatile graphics machines you can buy for less than five thousand dollars. Atari's Alan Alda ads fail to communicate the essential value these products represent; it is up to us, the Atari community, whose word of mouth can do more to sell computers than a box of Hawkeyes, to educate our innocent neighbors and overcome Atari's sluggish marketing. Otherwise we'll all end up marching to the beat of a drummer whose initials are IBM. □

Tired of high prices, poor service and hidden charges you get from other mail order companies . . .

Try RISING SUN SOFTWARE!

ENTERTAINMENT

ARTWORX
Strip Poker (D) $25.00
S.P. Data Disks 20.00

ATARI (ROM only)
Centipede $32.00
Defender 32.00
Dig Dug 32.00
Donkey Kong 35.00
Eastern Front 35.00
Galaxian 32.00
Joust 35.00
Missle Command 27.00
Ms. Pac-Man 35.00
Pac-Man 32.00
Pengo 32.00
Qix 32.00
Robotron 32.00
Space Invaders 27.00
Star Raiders 32.00

BIG FIVE
Miner 2049er (R) $35.00

BRODERBUND
A.E. (D) $25.00
Arcade Machine (D) ... 42.00
Choplifter (D) 25.00
Choplifter (R) 32.00
Oper. Whirlwind (D) ... 25.00
Sea Fox (D) 21.00
Sea Fox (R) 28.00
Sky Blazer (D) 28.00

BUDGECO
Raster Blaster (D) $21.00

DATAMOST
Airstrike (D/C) $28.00
Bilestoad (D) 28.00
Mating Zone (D) 25.00

DATAMOST
Micropainter (D) $25.00
Pooyan (D/C) 21.00
Zaxxon (D/C) 28.00

DON'T ASK
Poker Sam (D/C) $20.00
S.A.M. (D) 42.00

EDU-WARE
Prisoner 2 (D) $28.00
Rendezvous (D) 28.00

ELECTRONIC ARTS
Archon (D) $28.00
Hard Hat Mack (D) 25.00
M.U.L.E. (D) 28.00
Murder on the Zunderneuf (D) 28.00
Pinball Construction Set (D) 28.00
Worms? (D) 25.00

EPYX
Jumpman (D) $28.00
Jumpman Jr. (D) 28.00

INFOCOM (disk only)
Deadline $35.00
Enchanter 35.00
Planet Fall 35.00
Starcross 28.00
Suspended 35.00
The Witness 35.00
Zork I 28.00
Zork II 28.00
Zork III 28.00

INTELLIGENT STATEMENTS
Pro. Blackjack (D) ... $49.00

LIGHTNING SOFTWARE
Master Type (D) $28.00

MUSE
Castle Wolfenstein ... $21.00

ODESTA
Chess (D) $49.00
Checkers (D) 35.00
Odin (D) 35.00

PARKER BROTHERS
Astrochase (R) $35.00
Chess (R) 42.00
Frogger (R) 34.00
Popeye (R) 35.00
Q-Bert (R) 35.00
Risk (R) 42.00
Super Cobra (R) 35.00
Tutankam (R) 35.00

ROKLAN
Deluxe Invaders (R) .. $28.00
Gorf (D) 28.00
Gorf (R) 32.00
Wizard of Wor (R) 32.00
Wizard of Wor (D) 28.00

SIERRA ON-LINE
Crossfire (R) $25.00
Crossfire (D/C) 21.00
Frogger (D/C) 25.00
Mission Asteroid (D) .. 20.00
Sammy Lightfoot (R) .. 27.00
Ultima II (D) 42.00
Ulysses (D) 28.00
Wiz & Princess (D) 25.00

HARDWARE DEALS

MODEMS
Apple Cat II .. $299.00
Micromodem II 265.00
Micromodem w/terminal prog. 295.00
212 Apple Cat . 580.00

MONITORS
Amdek
Color I $299.00
Color II RGB ... 599.00
Color III RGB .. 399.00
RGB Card 149.00

USI
Pi1 9" Green . $119.00
Pi2 12" Green . 149.00
Pi3 12" Amber . 159.00
Pi4 9" Amber .. 129.00
Color 1400 .... 299.00

PRINTERS
C. Itoh
GX-100 ..... $229.00
Prowriter 399.00
Okidata
Microline 80 ... 349.00
Microline 82A . 449.00
Microline 92 ... 549.00

STRATEGIC SIMULATIONS
Battle for Normandie (D/C) .. $28.00
Battle of Shiloh (D/C) 28.00
Combat Leader (D) 28.00
Cosmic Balance (D/C) 28.00
Cosmic Balance II (D/C) 28.00
Cytron Masters (D) 28.00
Galactic Gladiator (D) . 28.00
Knights of the Desert (D/C0 ..... 28.00
Shattered Alliance (D) . 28.00
Tigers in the Snow (D/C) 28.00

SYNAPSE
Blue Max (D/C) $25.00
Dimension X (D/C) 25.00
Ft. Apocalypse (D/C) .. 25.00
Necromancer (D/C) ... 25.00
Pharoah's Curse (D/C) 25.00
Shadow World (D/C) .. 25.00
Shamus (D/C) 25.00
Shamus (R) 32.00
Shamus II (D/C) 25.00
Survivor (D/C) 25.00
Zepplin (D/C) 25.00

THORN EMI
Hockey (R) $28.00
Jumbo Jet Pilot (R) 35.00
River Rescue (R) 28.00
Soccer (R) 35.00
Submarine Commander (R) 35.00

UTILITIES & LANGUAGES

ADVENTURE INTERNATIONAL
Diskey (D) $35.00

ATARI
Assembler Editor (R) . $45.00
Atari Basic (R) 42.00
Macro Assembler (R) .. 68.00
Microsoft Basic II (R) .. 68.00
PILOT (R) 60.00

DATASOFT
BASIC Compiler (D) .. $63.00
Lisp Interpreter (D) ... 70.00
Edit 6502 (R) 125.00

OPTIMIZED SYSTEMS
Basic A+ (D) $56.00
Bug 65 (D) 25.00
Mac/65 (D) 56.00

EDUCATION

ATARI
Conversational Languages
French (C) $42.00
German (C) 42.00
Italian (C) 42.00
Spanish (C) 42.00
Educator Kit 117.00
Inv. to Programming 1 ..... 18.00
Inv. to Programming 2 ..... 23.00
Inv. to Programming 3 ..... 23.00
Juggles House (D) 23.00
Juggles House (C) 18.00
Juggles Rainbow (D) .. 23.00
Juggles Rainbow (C) .. 23.00
My First Alphabet (D) . 27.00
Programmer Kit 53.00
Tough Typing (cass) .. 18.00

EDU-WARE
Compu-Read (D) $21.00
Compu-Read (C) 15.00

SPINNAKER
Face Maker (D) $28.00
Hey Diddle Diddle (D) . 21.00
Kindercomp (D) 21.00
Most Amazing Thing (D) 28.00
Rhymes & Riddles (D) . 21.00
Snooper Troops 1 (D) . 32.00
Snooper Troops 2 (D) . 32.00

RISING SUN SOFTWARE
4200 PARK BLVD.
OAKLAND, CALIFORNIA 94602
(415) 482-3391

Ordering Information: We'll accept any form of payment—cash, personal check, money order, VISA/MasterCard, or C.O.D. Send cash at your own risk. Add $2.00 for UPS shipping; $3.00 for Blue Label Air. California residents add applicable sales tax. ALL orders shipped same day received. If we are out of stock on a particular item we will include a special bonus with your order when shipped.

master charge

CALL TOLL FREE 24 HOURS (ORDERS ONLY)
(800) 227-1617, ext. 114 (Outside California)
(800) 772-3545, ext. 114 (Inside California)

VISA

A TOOL SO POWERFUL, SO CONVENIENT, YOU WILL FIND IT INDISPENSABLE!

OMNIMON!

ANTIC, July '83: "Every microcomputer should have this sort of hardware monitor."

ANALOG, Aug '83: "A godsend for serious programmers. Our staff programmers swear by it."

WHAT IS OMNIMON!

OMNIMON! is a PC board which plugs into your 400/800 and gives you **complete control** of your computer. Even though it is **always available** (by pressing SELECT and SYSTEM RESET) it takes up no user memory! Use it to interrupt, examine and manipulate any program whether it be disk, cassette, or cartridge based. The flexible disk I/O allows you to write to or read from disk in either **single or double density**. One of the most powerful features is the ability to **recover from system lockup**, meaning that when your computer freezes, you don't have to worry about losing the program or text file in memory because you can always go into OMNIMON! and dump memory to disk! You can also do things like load a file without DOS and **edit raw sector data**. Many **debugging tools** are available: Display/Alter memory or registers, Disassemble memory, Search memory, Hex/Char modes, Single Step execution, JSR or GOTO address, Push/Pull stack, Printer dump, etc. After intrrupting a program with OMNIMON!, many times it is possible to return to the program as if you had never left it (e.g., BASIC, DOS, etc.). Instructions are provided for the addiiton of a a simple toggle switch to make OMNIMON! invisible, thus making it **compatible with all software**.

HOW TO ORDER

Base price of OMNIMON! is $99.95. We accept checks, money orders, credit cards, or COD. Our toll free order desk is (800) 227-3800 ext. 561. If you have questions please call us direct or write:

CDY Consulting
421 Hanbee
Richardson, TX 75080
(214) 235-2146

OPTIONS:

Advanced version ($15 extra): Same as standard version with the addition of Hex conversion, Verify memory, **Happy drive** upload/download, and **lockup recovery.**

Ramdisk version ($25 extra): Same as advanced version but it allows you to **use your AXLON 128K Ramdisk with any DOS** which uses standard SIO calls (e.g. ATARI, OSA+, MYDOS, etc.). Even reboot (coldstart) without disturbing the contents of Ramdisk! Single step is deleted.

OMNIMON 2 (8K version) early 1984 standard version required. Dealer/Distributor inquiries solicited.

Wordman

NOW YOU REALLY **ARE** THE BOSS!
WORDMAN is WORD MANagement at its best!
LETS YOU MANAGE EVERY WORD YOU EVER WANTED TO PROCESS.

Word Processing used to be a puzzle, a pain, and a printer predicament, but now, with WORDMAN, here's what you can do . . .

* CREATE **YOUR OWN** PRINTER COMPATIBILITY
* PRINT **FROM** OR **TO** DISK FILE
* CONTROL RIGHT MARGIN TO **YOUR** SPECIFICATION
* SELECT FUNCTIONS FROM 3 MENUS
* CREATE, ADD, EDIT, AND COPY
* PROGRAM SINGLE KEY FOR AUTOMATIC TEXT REPETITION
* AUTOMATICALLY CENTER TITLES AND FORMAT PAGES
* USE MIXED DENSITY DISK DRIVES
* MERGE MULTIPLE FILES FROM DISK FOR DOCUMENT CREATION
* CREATE DOCUMENTS WITH EMBEDDED GRAPHICS
* SEARCH FOR AND REPLACE WORDS OR PHRASES
* DELETE, INSERT, OR COPY BLOCKS OF TEXT

WORDMAN - THE NEW WORD MANAGEMENT PROGRAM WITH POWER, PREDICTABILITY, AND P-ZAZZ!

ONLY $39.95
JIM CHANEY & ASSOCIATES
916 E. BERKELEY, RICHARDSON, TX 75081
Add $2.00 PER ITEM For Shipping On Quantities Less Than 5
DEALER INQUIRIES INVITED

REQUIRES 40K RAM, BASIC CARTRIDGE, AND DISK DRIVE

SALE SALE SALE

FASTCHIP - ATARI 400/800* $41.95 This month $29.95
A custom floating point RDM that can give you up to 3½ times the speed of the original ROM. It can be installed in minutes.

RAMROD MMOS - ATARI 800* $189.95 This month $149.95
A new operating system board that replaces the existing board. Allows use of standard roms or EPROMS. Comes with OSN O.S. Features FAST CURSOR. GRAPHICS MODES 12-15. VARIABLE CASS. BAUD RATE. CONTROL ON POWER UP. Also addresses 4K block of memory that is not currently accessbile in the system. PLUS, every board comes with OMNIMON!, a resident machine language monitor with commands like display alter memory, disassembler, printer and disk I O. etc.

4KRAM $15.95 **2732 EPROM 350ns. $10.00**
OMNIMON! - ATARI 400/800* $99.95
As described above, but comes on a piggyback board that plugs into the operating system socket.

52K RAM BOARD - ATARI 400/800* SPECIAL $89.95
REPLACES EXISTING RAM BOARD(s). FEATURES: 64K RAM, GOLD CONTACTS, SOCKETED IC's, 5 YR WARRANTY

MYDOS 3.X DISK OPERATING SYSTEM $29.95
USES LESS MEMORY than 2.OS, and supports 5¼ & 8 inch drives. A MUST.

PRO BOWLING VERSION THREE (disk) $17.95

SD/DD SECTOR COPIER (disk) $17.95

ATARI SOFTWARE SPECIALS:

STARRAIDERS CARTRIDGE	$20.00
MISSILE COMMAND CARTRIDGE	$20.00
STOCK ANALYSIS	$11.00
BOND ANALYSIS	$11.00
MORTGAGE & LOAN ANALYSIS	$10.00
MAILING LIST CASS.	$10.00
BUSINESS COMMUNICATIONS	$12.00
JAWBREAKER DISK	$12.00
COMPUMATH DECIMALS CASS.	$12.00
COMPUMATH FRACTIONS CASS.	$12.00
JOYSTICKS (PAIR)	$15.00

Call 214-423-1781 to order C.O.D. or send check or money order to NEWELL INDUSTRIES, 3340 Nottingham Ln., Plano, TX. 75074. Texas residents add 5% sales tax where applicable. Orders outside the U.S. add 3% ($3 min) for shipping. C.O.D. orders add $1.65. Call or write for additional information.

ATARI 400/800 are Trademarks of ATARI INC.

OUR GAME

by Joel Gluck

This is **Our Game**, the column in which you and I collaborate in writing a video game. You mail in ideas, suggestions, criticisms and comments. I carefully read each letter I receive, always watching for original ideas and viewpoints. Some of the mail actually ends up in **Our Game**. But all of it, even the letters scrawled in crayon, affect what our game is going to be like.

Viewer mail.

Reader response has not exactly been overwhelming, but **Our Game** has definitely been receiving increasing amounts of mail. Here are some thought-provoking excerpts:

Derek McClintock of Pueblo, Colorado writes:

"I often wonder how my ship, person, frog, etc., can become so totally obliterated and yet, through the magic of VLSI, a showroom model will inevitably reappear without a scratch. Why not, instead, have a sequence so that when the ship is blown to bits, a new one appears next to the wreckage and a little blip floats from one to the other, like the escape pods NASA is developing for their spaceships."

Derek is right — games in which a player has multiple lives usually leave the transition from one life to another unexplained; it would be interesting to see more elaborate or realistic rebirths.

From Manchester, New Hampshire, Scott Mitchell has this to say:

"A game I got to thinking about one day involves a day in the life of a bee. The main idea is to pollinate all the flowers and get back to the hive to start the next screen. Some flowers, would be more difficult to reach due to wandering enemies such as spiders, preying mantises and birds. Some flowers are poisonous — so don't touch the side or any part but the pollen. Some flowers clamp shut as well. There might be spider webs to become tangled in or flowers that open up only briefly and present a timing problem to pollinate."

Now there's an original idea! Although it is somewhat violent, the violence is not of the standard arcade shoot'em-up style — the pollination is constructive, not destructive as are most video games.

Dave Mundy of St. Ann, Missouri, has his mind made up about one aspect of our game:

"I would like to see the game playable by two people at once, rather than two people alternately as is common in most every game on the market. I remember the old **Pong** *game where two could play at once and both players were against each other, rather than against the machine. Half of the satisfaction in winning a game is making your opponent feel four inches tall. So let's make the opponent human, rather than the computer or some high score."*

I feel the same way, Dave. Computer games are much more fun if you can play head-to-head with another person, whether it be competitive or co-operative.

George Price, of East Detroit, Michigan, proclaims himself to be a tried and true wargamer:

"My computer to me is my constant adversary, a fellow gamer who is always willing and waiting to thwart my latest and sometimes not so brilliant strategy. Unfortunately, except for a few companies, there are very few programmers writing to satisfy my gaming needs. That is where you come in. Hopefully, with the tools you will provide, I will be able to write my own simulations, filling a 155mm void in my computing life."

George, I like your attitude. Even if **Our Game** doesn't develop the sort of game you enjoy, you say that you can learn from this column and write games to your own liking. That is part of the aim of **Our Game**, not to create the ultimate video game but rather to write a game in such a way that the reader can really learn from the process.

Chris Sicks of Orchard Park, New York, wants to

see a game in which spirits — the servants of the "Supreme Being," as he puts it — travel down to earth to fight devils who are picking up and carrying off the populace (similar to "Landers" in **Defender**). The spirits' weapon? Writes Sicks: "*The spirits summon down lightning bolts to destroy the devils...*"

Chris's game idea is a refreshing change — the themes of the game are derived from myths of the supernatural instead of the usual science-fiction. I'm getting tired of space ships, myself.

Bob Mulholland writes from Jersey City, New Jersey:

"*Everyone has seen at least one movie with a big spectacular stunt in it, and marveled at the way these stuntmen can coordinate such a stunt. So why not make a game that will include a wide range of stunts in which the player can participate. The stunts can range from a barroom brawl to a human fireball jumping out of a helicopter 20,000 feet in the air into a bucket of kerosene.*"

Not bad, Bob. I like the idea of play-acting a role from real life, whether it be stuntman, surgeon, mail carrier, or politician. More computer games may take advantage of this role-playing fantasy in the future — such games can be educational as well as fun.

You also brought up the possibility of a game comprised of several small games. One of the best of this genre was a game for the Apple called **Olympic Decathlon**. In the arcades, **Tron** has been the most popular "multi-game." The advantage of such a game is variety — players confront different challenges in different environments. We'll probably be seeing more of these games, too, as arcade machines and home computers are built with increasingly larger memories.

This is how Larry Nocella, a Woodbury Heights, New Jersey resident, described the abilities of Herman, a user-controlled creature in his game idea, **The Adventures of Herman**: "*...he can turn red with anger for ten seconds — this chases away the meanies — or he can turn blue and take 3 big steps.*"

Arcade games would have us believe that we spend our time shooting and blowing up people we dislike. Turning red with anger, though comical, is a real-world touch.

Claude Galinsky, of Watertown, Massachusetts, has this to say:

"*My girlfriend thinks video games are for nerds. I can see her point. They exclude other people and can give socially inept players an easy way to avoid relating to others. There is no reason why an absorbing game has to be single-player.*

"Suppose we write a simulation of a road rally. Having driven a rally single-handed, I know it can be done, but you really need one person to drive and another to navigate. You have to cooperate and communicate; sometimes you even have to argue, but it's very absorbing and a whole lot of fun. Or how about a WWII bomber simulation where one player is the pilot, another is the navigator, and a third is the bombardier? Or a bank robbery game in which one player drives the getaway car while another shoots at the pursuers? Or a football game where one player passes while the other receives?

"Another thought on video/computer games: girls don't play them! This is an extremely bad state of affairs, for the simple reason that computer games are the primary means by which kids start to become computer-literate. Now, I would never claim to understand any member of the opposite sex, but it seems to me that they are less attracted to games involving violence and mayhem than are boys. Their hand-eye coordination seems to be directed more to achieving precision rather than speed. They seem to be creative more in interpersonal ways (making up stories about other people, for instance) than in person-machine ways (e.g., building a bridge). Something, I don't know what, should be done to appeal to them in **Our Game**."

The combination of player cooperation with the challenge of unique roles is intriguing. This is the kind of game that arcade manufacturers unfortunately have avoided (it simply doesn't seem to have the money-making potential) and that is perfect for home computers. It is an exciting possibility for **Our Game**.

As for the benefits of our game being equally enjoyable for both males and females, I agree totally. so that an 8-year-old can play as easily as his 30-year-old mother.

The **Our Game** mailbag is pretty empty at the moment, so don't hesitate to send in anything you think might be fun and different.

The hard facts.

This may come as a shock, but the computing power of your Atari is not infinite. As a matter of fact, the 400, 600, 800 and 1200 suffer from a severe case of limited resources. This may not seem obvious when playing a game written in machine language, the built in "language" of your machine's microprocessor. The balls and paddle in **Super Breakout** have no trouble moving at very high speed. But even machine language has its limitations — in **Star Raiders**, for example, things slow down quite a bit during explosions.

Atari BASIC is written in machine language. Its major purpose is to take the BASIC commands you type in, interpret them (figure out what they mean),

and execute them (carry them out). BASIC is a middleman between you and the computer — you tell BASIC what you want, and BASIC tells the machine what you *really* want, in terms the machine understands.

Middle-men increase the cost of things. In the case of BASIC, you pay the price in speed. Machine language, for most tasks, runs at least 100 times faster than BASIC.

Our game, the game that we are going to write together, is going to be written in BASIC, not in machine language or some other language. There are several reasons for this:

1) Almost everyone has BASIC;
2) Almost everyone knows or is learning BASIC;
3) Teaching machine language is such a big job it could be the subject of a whole other column. [*It already is. See* **Boot Camp** *on page 124.—Ed.*]

BASIC is obviously the way to go, but we are up against the speed problem. Some would suggest that a BASIC compiler — software that turns a BASIC program into faster code — would be the answer. And often it is. But **Our Game** can't expect readers to go out and buy a BASIC compiler, and neither should we limit our program only to statements that a compiler can handle.

The answer to this mess is to stick with plain old Atari BASIC, and to write efficient code that avoids using up the resources of the machine.

This, however, gives us a new mess. With the limited resources available, we are *very* limited when it comes to moving objects — as a matter of fact, I'd say that three moving objects exceeds the limit for an ordinary BASIC game. For an object to move at a decent speed, we're limited to one, count'em, *one* object.

When writing **Attank!**, a two-player action game in BASIC, I found that two tanks moving simultaneously travel rather slowly. And when it came time to write the routine that fires a bullet, I had to stop movement of the tanks just to get a passable bullet speed.

Limited resources also means limited computation time. The computer does not enjoy thinking a lot in the midst of an action-packed game. This means that any computer-controlled opponent(s) or objects must have a fairly simple algorithm to control them; more than a few IF-THEN statements from top to bottom in the decision-making tree makes your action-packed game start to crawl.

Other elements of a program can bog your game down. For example, overly-complex sound effects can turn a silent, speeding space ship into a swooshing but soggily-slow space ship. Another slower-downer is to program a visible timer, or the like, to run constantly on the screen. Keeping time (and displaying it) takes time.

Now don't get me wrong. There is nothing I enjoy more than sound effects and animation. But when writing our game, we're going to have to be careful when it comes to speed.

Keep these limitations in mind when thinking up ideas for our game. Remember, the more feasible your idea, the greater chance it has of appearing in these pages.

Fun 'n' games.

There have been requests in the **Our Game** mailbag that I print a small, sample game. After all, this is the fourth **Our Game** and not a single listing has appeared.

Feast your eyes on **Listing 1**, a program called **Bounce.**

Bounce is not strictly a game, even though it has graphics and sound-effects and is controlled with a joystick. **Bounce** does not give you a score, or allow you to achieve higher levels of difficulty. You do not play **Bounce**, you play with **Bounce.**

When I first wrote the program, I couldn't get enough of it. I must have played with the thing for a solid week. There are nuances, strategies, and goals you can set for yourself. But I'm not going to tell you any of them.

Play with it, learn from it, and enjoy it.

One more thought on this subject: If you've written a program along the lines of **Bounce** — short, fun, and interactive — don't hesitate to send it in. If it's any good, I won't hesitate to print it.

Programming with the top down.

Last time we discussed the idea of prototype — which is kind of a "warm-up" for writing a major program. But there's more to programming than warming up. Are there right and wrong methods of programming?

There is no clear answer. It all depends on your reasons for writing the program, and the conditions under which the program is going to be used. You may, for example, be writing a short program to add some numbers. In that case, whatever gets the job done most simply is the "right" method.

But what about when you are writing a large program, like a game? There are a few important points that should be kept in mind:

1) Efficiency of the program in terms of memory and speed;
2) Debug-ability;
3) Ease of repair and/or modification;
4) Clarity;
5) Future usability.

Point one is pretty clear — if your program is efficient in terms of memory, you have more space to add features and make fixes. If it is speed-efficient, you have more processor time left over for extras. For a game, this could mean more elaborate special effects or game logic.

Point two refers to the relative ease in locating bugs in the program. How easy it is to repair those bugs and make other modifications is the purpose of point three.

Point four asks the question, "How easy would it be for someone else to read and understand the program?" When you look at the program a year after you write it, you are, effectively, someone else — you probably don't remember how you wrote it.

Finally, point five asks "How reusable are the individual parts of the program?" For example, would a certain routine from program A be easily transportable to program B?

Points two through five are frequently ignored by programmers who are in a rush or don't care. In the long run, though, such careless programming can cost you time — in debugging and revising your software. There are, however, techniques to make careful programming easier.

Top-down programming is one such technique. When you program in the top-down manner, you start with the whole problem and proceed to break it down into smaller and smaller pieces, until what you have is essentially the outline of a program.

We do this all the time in everyday life — although we don't always think it through. For example, the simple task of giving Rover a bath can be broken down into sub-tasks:

Giving Rover A Bath

A) Prepare the bathtub.
B) Catch Rover.
C) Get Rover into the tub.
D) Clean Rover.
E) Dry Rover.
F) Thank Rover and clean up.
G) RETURN.

RETURN simply means to return to whatever you were doing. If giving Rover a bath was part of a bigger list of things to do, you would continue with the next thing on the list.

Each of these steps can, in turn, be broken down. Here's a possible breakdown for item "B":

Catching Rover

1) Walk around the house yelling "Yoo-hoo, Rover, time for your bath!"
2) IF Rover isn't in sight, THEN GOTO 1.
3) Say "There you are, you naughty boy!"
4) Make a grab for Rover, avoiding the tusks.
5) IF you got him, THEN RETURN.
6) Lead Rover to the bath with a trail of peanuts.
7) IF it worked, THEN RETURN.
8) Give up. POP. RETURN.

Notice that the list of sub-tasks begins to look like a program. There are IF-THENs, loops (the THEN GOTO 1), and of course RETURNs to allow you to continue back at step C (getting Rover into the tub).

The POP command is a little strange. It means to pop out of the current level (or list of things to do) and continue the current list *as if we were on the next higher level list*, which in this case is the list "Giving Rover a Bath." So, in order to "give up" in step eight, we have to first POP out to the main list and then RETURN to exit that list. If we had said RETURN without the POP, it wouldn't have meant giving up — it would have meant, incorrectly, that we had succeeded in catching Rover (because it would only exit us from the current "Catching Rover" list.)

When thinking in a top-down fashion, there are a few problems, like the POP mechanism, but in general it is a good, logical way to solve problems — and therefore a good way to program; for what is programming besides a systematic approach to problem solving?

Of course, for us game writers, the top level problem is not "Giving Rover a Bath" but rather "Telling My Computer How To Execute My New Game Idea, **Martian Big-Game Repairman**," or the like. But breaking a game idea down into steps differs little from what we did above to the problem of cleaning Rover. If anything, top-down design of the game is easier.

(continued on page 46)

128K SOLID STATE DISK EMULATOR SYSTEM

FOR

ATARI 800™

NOW

$299.00

RAMPOWER 128™

- *DISPLAYS DATA IN MICROSECONDS*
- *20x FASTER THAN ATARI—810*
- *NO MOVING PARTS*
- *SLOT COMPATIBLE*
- *160K POTENTIAL*

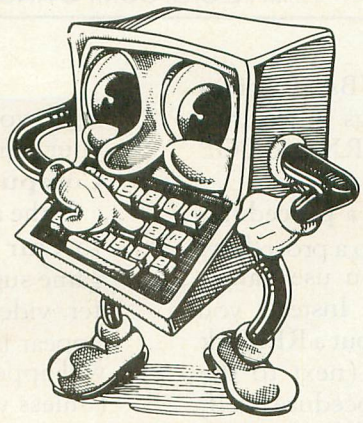
RAMPOWER 128 is actually eight - 16K banks of memory . . . lots of room for graphics, data, tables and program overlays. "Omnimon" by CDY CONSULTING, or the "Integrator" by ADS allows you to customize your own DOS, AUTO-BOOT, and some copy-protected programs. RAMPOWER 128 also enhances SYNAPSE Business Software with added memory and speed; "File Manager 800+", "Syn-FILE", and "Syn-CALC." VALFORTH and OSS are also compatible. (Note: RAMPOWER 32K when modified for the first slot makes RAMPOWER 128 compatible with 80 column boards.

VISA MasterCard

AXLON

1287 N. Lawrence Station Road
Sunnyvale, CA 94089

(408) 747-1900



INTEC

RAM BOARDS

Lifetime Warranty

ATARI

32K	400 / 800	59.95
48K	400	84.95
64K	400	109.95
16K	Vic 20	69.95

PRINTERS

EPSON FX 80	529.95
EPSON FX 100	729.95
PROWRITER 8510 AP	369.95
GEMINI 10 X	294.95

INTERFACES

Microbits MPP 1100 *Atari*	79.95
Grappler + Apple	124.95
Cardco . . . VIC 20 - '64'	59.95

MODEMS

MPP 1000 Atari Direct-Connect W/Term. Software *(No. 850 needed)*	159.95
Hayes Smart Modem 300 Baud	209.95
Rana 1000 Drive	349.95

Ordering Information:

We accept M/C, VISA, Money Orders, and Cashier Checks. Sorry No C.O.D.'s.

SHIPPING: Add 3% UPS ($3.00 Minimum). APO/FPO 5% ($5.00 Minimum). International Orders 10% ($10.00 Minimum). Credit Cards add 3%. California Residents add 6% Sales Tax.

MICRO MERCHANT

898 Via Lata Suite 'H'
P.O. Box 1516 A
Colton, CA 92324

ORDERS ONLY

800-652-8391

Customer Service

714-824-5555

MasterCard VISA

An indispensable tool for your 400/800

dy **OMNIMON!**

Incomparable power & convenience!

Here is what the reviews say:

ANTIC, July '83:
"Every microcomputer should have this sort of hardware based monitor."

ANALOG, Aug. '83:
"...a godsend for serious programmers."
"Our staff programmers swear by it."

OMNIMON! is not detectable by other software with the addition of a simple toggle switch. The many features include: Complete system control - Resident: Uses no memory but always available! Recover from lockup - Flexible disk I/O Single or double density - Load file w/o DOS - Edit raw sector data - Dump any program to disk - Supports Happy! Many debugging tools - Very user friendly

Price: $99.95 Check or MO

Dealer/Distributor inquiries solicited

Credit Card or COD: (800) 227-3800 ext 561 Orders only please!

CDY Consulting 421 Hanbee Richardson, TX 75080-214-235-2146

RATED X

XBASIC by George Schwenk

An extension of ATARI ® BASIC. **XBASIC** provides 30 new functions, including: All this power for only **$29.95**

- **Integer and String Arrays**
- **Player Missile Graphics**
- **Special Graphics Modes**
- **Vertical Blank Sound**
- **Requires less than 3 K RAM**
- **Detailed Manual / 6 Examples**

XBASIC is ideal for tape users.
Available as disc or cassette— Please specify

FINE SCROLLING IN BASIC

SCROLL IT by George Schwenk

SCROLL IT is a relocatable machine language program which allows the BASIC programmer to do variable speed horizontal, verticle and diagonal fine scrolling with a simple POKE. SCROLL IT supports all 6 ANTIC character modes and handles the intricate details of display list maintenance. Available for ATARI® computers on tape or disk — **$19.95**

Also available — **FDOS — $39.95**

FDOS gives FORTH users named file handling (DATA FILE, FNT) and normal DOS functions. Special features allow direct loading and/or conversion of DOS 2.0S files to FDOS Files, sector and screen copying. Requires APX Fig — FORTH or compatible FORTH.

ORDER NOW!

SUPERware
2028 Kingshouse Rd.
Silver Spring, MD 20904
(301) 236-4459

Dealer Inquiries Invited

MC, VISA, AMX, Check, M.O. accepted.
Include $2.00 shipping, MD residents add 5% tax.

One reason it's easier is that Atari BASIC gives you special commands to break things down into procedures. They are GOSUB, RETURN and POP (although we'll try to avoid POP).

The RETURN gets you back from a procedure, like we saw above. GOSUB gets you into a procedure. Unfortunately, BASIC doesn't let you use names like "Catching Rover" for procedures. Instead, you have to use line numbers. But you can put a REMark on the line that calls the procedure (next to the GOSUB) and on the first line of the procedure itself.

If "Giving Rover a Bath" were a BASIC computer program, here's what parts of it might look like:

```
1000 REM - Giving Rover A Bath
1100 REM - Version 1.0
1200 GOSUB 2000 : REM - Prepare The Bathtub
1210 GOSUB 3000 : REM - Catch Rover
1220 GOSUB 4000 : REM - Get Rover Into Tub
1230 GOSUB 5000 : REM - Clean Rover
1240 GOSUB 6000 : REM - Dry Rover
1250 GOSUB 7000 : REM - Thank Rover/Clean Tub
1260 END
2000 REM - Prepare The Bathtub
.
.
.
2280 RETURN
.
.
.
3000 REM - Catch Rover
3100 GOSUB 3200 : REM - Walk And Yell "Yoohoo!"
3110 IF (Rover Not Found) THEN GOTO 3100
3120 GOSUB 3300 : REM - Say "Naughty Boy!"
3130 GOSUB 3400 : REM - Grab Him
3140 IF (Rover Caught) THEN RETURN
3150 GOSUB 3500 : REM - Lay Trail Of Peanuts
3160 IF (Rover Caught) THEN RETURN
3170 POP : END : REM - Give Up, End Program
3200 REM - Walk And Yell "Yoohoo!"
.
.
.
etc.

•
```

Notice that in BASIC, END means RETURN on the highest level; it returns control to BASIC, exiting your program/procedure.

Don't laugh, but the listing for our game is going to resemble the above fictional program quite a bit. Top-down programming is quite advantageous, making programs easier to read, understand, and debug, simply because it encourages a *modular* programming style. Not only that, these modules or procedures that you write are easier to transport to other programs; they are not tied to the rest of your program in the usual spaghetti of GOTOs. (Note that in truly modular programming, GOTOs are used only to perform loops that FOR-NEXT can't handle — never are they used to jump around.)

Next time in this tutorial part of **Our Game**, we'll be discussing the arcane art of debugging — don't forget to bring along your favorite insecticide!

Instant fame.

Do you crave fame? Have you always wanted to see your name in lights? Or, at least in **A.N.A.L.O.G. Computing**?

If the answer is yes, then all you have to do is write to **Our Game**. If you send in a decent game idea, game suggestion or comment on the state of computer/video games in general, it will almost definitely appear in these pages. And when it does, your name will appear with it, not to mention your home town (unless you are embarrassed about the whole thing, in which case we'll also print your social security number, shoe size, and favorite brand of toothpaste).

Share those ideas with us!

Send your mail to:

Our Game
c/o A.N.A.L.O.G. Magazine
P.O. Box 23
Worcester, Mass. 01603

Don't miss the next **Our Game** — we may actually start writing THE GAME itself. □

Listing 1.

```
100 REM - Bounce rev 2.2
110 REM - by Joel Gluck / June'83
120 GOSUB 160:REM - INIT
130 GOSUB 280:REM - DRAW
140 GOSUB 470:REM - BOUNCE
150 GOTO 130
160 REM - INITIALIZE
170 DIM XD(15),YD(15)
180 FOR Z=5 TO 15:READ A,B
190 XD(Z)=A:YD(Z)=B:NEXT Z
200 DATA 1,1,1,-1,1,0,0,0
210 DATA -1,1,-1,-1,-1,0,0,0
220 DATA 0,1,0,-1,0,0
230 GRAPHICS 5:POKE 752,1
240 SETCOLOR 0,0,13:SETCOLOR 1,9,2:SET
COLOR 2,0,0
250 X=40:Y=24
260 COLOR 3:PLOT 0,0:DRAWTO 79,0:DRAWT
O 79,39:DRAWTO 0,39:DRAWTO 0,0
270 RETURN
280 REM - DRAW
290 SETCOLOR 2,12,6
300 ? :? "* Use stick to draw walls,"
310 ? "* Hold trigger to erase,"
320 ? "* Hit ESC to clear screen,"
330 ? "* Hit SPACE to bounce.";
340 COLOR 1:PLOT X,Y
350 J=STICK(0)
360 IF PEEK(764)=33 THEN POKE 764,255:
RETURN
370 IF PEEK(764)=28 THEN POKE 764,255:
GOSUB 230:GOTO 290
380 IF J<>15 THEN 400
390 LOCATE X,Y,G:COLOR 3-G:PLOT X,Y:GO
TO 350
400 A=XD(J):B=YD(J)
410 IF X+A<1 OR X+A>78 OR Y+B<1 OR Y+B
>38 THEN 350
420 SOUND 0,(200-X-Y)*STRIG(0),8+2*STR
IG(0),4
430 COLOR 3*STRIG(0):PLOT X,Y
440 X=X+A:Y=Y+B
450 COLOR 1:PLOT X,Y
460 SOUND 0,0,0,0:GOTO 350
470 REM - BOUNCE
480 ? :? :? "* Hit SPACE to draw.":?
490 COLOR 1:PLOT X,Y:A=1:B=1:L=0
```

```
500 IF PEEK(764)=33 THEN POKE 764,255:
RETURN
510 LOCATE X+A,Y+B,G:IF G<3 THEN COLOR
 2:PLOT X,Y:X=X+A:Y=Y+B:COLOR 1:PLOT X
,Y:L=L+1:GOTO 500
520 SOUND 0,L*4+20,10,8:LOCATE X+A,Y,P
A:LOCATE X,Y+B,PB:SOUND 0,0,0,0:L=0
530 IF PA>2 THEN A=-A:GOTO 570
540 IF PB>2 THEN B=-B:GOTO 590
550 IF PEEK(53770)>127 THEN B=-B:GOTO
500
560 A=-A:GOTO 500
570 IF PB>2 THEN B=-B:GOTO 500
580 COLOR 2:PLOT X,Y:Y=Y+B:COLOR 1:PLO
T X,Y:GOTO 500
590 IF PA>2 THEN A=-A:GOTO 500
600 COLOR 2:PLOT X,Y:X=X+A:COLOR 1:PLO
T X,Y:GOTO 500
```

CHECKSUM DATA

(see p. 70)

```
100 DATA 598,71,185,175,412,707,787,59
8,256,624,974,244,446,437,453,6967
250 DATA 982,576,604,886,743,256,63,30
8,794,719,920,756,796,739,751,9893
400 DATA 646,918,530,884,194,724,169,4
06,305,145,742,399,231,901,913,8107
550 DATA 925,144,895,257,898,229,3348
```

EAGLES
Strategic Simulations, Inc.
883 Stierlin Road
Building A-200
Mountain View, CA 94043-1983
40K Disk $39.95

by Edward Bever

Have you ever dreamed of flying a World War I biplane, twisting and turning in desperate combat against an enemy ace? If so, then **Eagles** was intended for you. Once you boot this introductory level game into your system, you and up to nineteen friends can dogfight to your hearts' content, and the only thing that can get hurt is your pride.

Before the game begins, the computer presents a wide variety of choices about the situation to be played: the number of pilots on each side, the period during the war the combat is to take place, the scale of the battle, the planes' initial positions, the weather conditions, and whether or not special aircraft like two-seaters and balloons are to be included. Each situation can be played as a discrete game, or they can be linked together as a "campaign" game. If you like, the computer will do enough of this decision-making itself that a beginner need not feel overwhelmed; at the same time, an experienced player can create just about any imaginable situation in the air during the Great War.

Once initialized, the program displays a "bird's eye view" of the first player's plane and any friends and enemies in the vicinity. It also presents a list of all the possible moves: straight ahead, climbs and dives, turns, diagonal slips, skids and rolls. Movement is calculated in terms of hypothetical "squares," and the probability that a plane will execute a maneuver successfully is a function of its maneuverability and speed ratings. Most maneuvers involve movement of one square at a time, but one allows two forward moves, while some others span two turns and may not be interrupted in the middle.

If a plane ends its movement heading at a nearby enemy, it may attempt to shoot it down. The computer displays the probability of a hit on all potential targets, and the player indicates which one his plane will attack and the length of the burst it will fire. The longer the burst, the greater the chances of damaging the target, but also the faster the machine guns use up their ammunition, and the greater the danger that they will jam. The computer then calculates and reports on the result, which will be a miss, a hit that merely leaves a few holes in the canvas, a hit that causes some kind of serious damage, or a kill. The computer may also report that the guns have jammed. The pilot may be able to clear them in a subsequent turn, or he may have to pray and head for home.

The computer can control all the planes on one side and/or wingmen for the players. Naturally, its moves are somewhat less elaborate since it does not have to interact with a person. Regardless of who controls it, though, each plane moves and fires in succession, first all the Allied aircraft, then all the Germans.

The birds eye view display shows the horizontal relationships between the moving plane and its neighbors, but it cannot convey their verticle relationships, and it does not stretch far enough to indicate all planes that might intervene in its battle. Therefore, players can flip to another display that lists all the aircraft in a wider area, along with each one's relative altitude, horizontal position, heading, and attitude (e.g. whether it is banking or executing a loop). This display is accessed by a single keystroke, and the player can return to the main display by tapping the space bar.

Some of **Eagle's** features are quite good: the variety of situations and aircraft, the simplicity of the game system, and the quality of the graphics on the main display. However, it suffers from a number of faults that seriously detract from its value as a simulation and as a game.

First of all, the program loads too slowly. It takes approximately one minute before the program asks which graphics chip is in the machine, a second full minute before it begins requesting decisions about the situation, and a third full minute after this information is supplied before the game begins. While such sluggishness would not be so bad if the program only had to be loaded once per session, **Eagles** must re-load before each dogfight, going all the way back to determining which graphics chip is in the machine. Particularly when a player is first learning the game, it can seem that more time is spent waiting for the game to load than playing it. This seriously detracts from its value for beginning gamers, its target audience. *(continued on next page)*

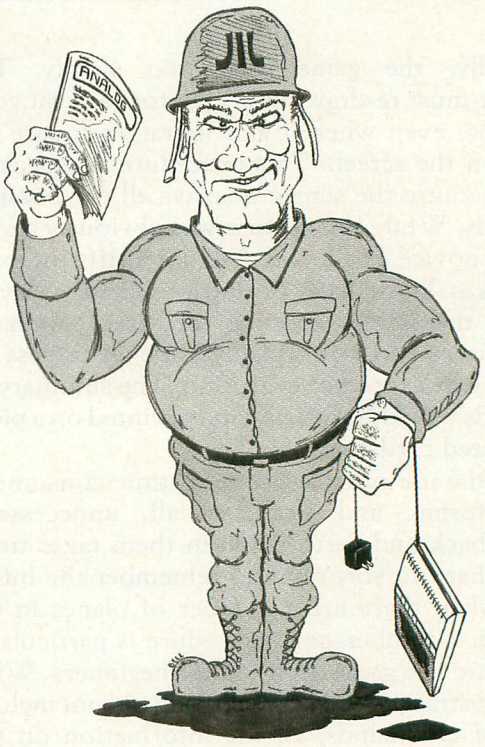
Secondly, the game plays too slowly. The computer must re-draw the screen for each move of each plane, even when it and its target square are already on the screen. This procedure is especially tedious because the screen displays all the possible commands. While this feature was obviously added to aid the novice, I believe it does not justify the extra time it takes. Particularly during long chases, players will find themselves entering the same command over and over, and would gladly trade the useless list for some extra speed of execution. The summary of commands would be just as handy printed on a piece of laminated cardstock.

Third, the use of two screens is time consuming and confusing, and worst of all, unnecessary. Flipping back and forth between them takes time, and it is hard to sort out and remember the information when there are a number of planes in the area. Such a cumbersome procedure is particularly unfortunate in a game designed for beginners. What is most frustrating, if the main screen did not include the list of commands, all the information on the screen could fit in its space. Or, better yet, a single display with smaller pictures could hold all the information by itself.

Finally, the game is not realistic. While a great deal of care obviously went into the research and design, the historical commentary that comes in the documentation itself mentions several maneuvers that were used to let a pursuing plane overshoot its target (and thereby become the pursued) that are not possible in the game. Furthermore, the commentary emphasizes the danger of overstressing the aircraft, but in **Eagles** the planes' maneuvers are rigidly bound by the limits of safety. Nevertheless, these problems are relatively minor, and would be unimportant if the game gave the "feel" of the situation. As the commentary itself notes: "Once the dogfight was joined, ability to maneuver quickly and correctly became all important. Moves had to be made so fast that there was no time to consider — you merely had to sense the right action and, by using the stick and rudder-bar almost instinctively, to bring it off. Dogfighting seemed...a matter of elemental, nearly automatic reaction." Nothing could be further from the play of this game.

Perhaps the author set himself an impossible task trying to design a computerized dogfighting game that moves at the speed of Atari BASIC. **Eagles** is authentic without being particularly realistic; it is simple without being especially playable. I find it mildly enjoyable as a sort of three-dimensional chess, but no one else whom I have asked to try it has reacted at all favorably to it. SSI has brought us many fine war games, but **Eagles** does not rank among them. □

BASIC TRAINING



by Tom Hudson

With this issue, **ANALOG** begins a new column: **BASIC Training**. This column will concentrate on one or more features of Atari BASIC, while presenting a useful program or subroutine to illustrate the techniques discussed in the column. Readers are invited to send topic suggestions to **BASIC Training, ANALOG Computing**, P.O. Box 23, Worcester, MA 01603.

Just a souped-up calculator?

It has been said that computers are nothing more than glorified adding machines. True, computers can't do anything without explicit instructions, but they can do one thing most calculators can't: They can make decisions.

The computer's ability to make decisions is what allows it to crank out hundreds of paychecks one minute and play a championship-level game of chess the next. This article will examine BASIC's decision-making statement, "IF."

Go with the flow.

To illustrate some of the concepts presented in this column, I will be using a programming tool known as the FLOWCHART. Some readers may be familiar with flowcharts already, but for those not acquainted with this method, I will briefly summarize the technique.

For our purposes, we will use only three flowchart symbols, shown below.

Figure 1

The leftmost figure, a rectangle, is the general PROCESSING symbol. This symbol will be used to represent most operations performed by the computer.

The center figure, a circle, is the CONNECTOR symbol. It is used, not surprisingly, to show a logic connection between parts of a program.

The rightmost figure, a diamond, is the DECISION symbol. Decisions are what this article is all about. Whenever a decision must be made by the program, the DECISION symbol is used.

In order to see how these flowcharting symbols are used, let's look at a situation we're all familiar with: Making a phone call.

Figure 2

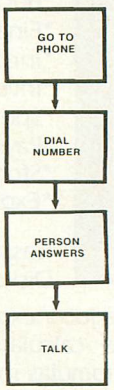
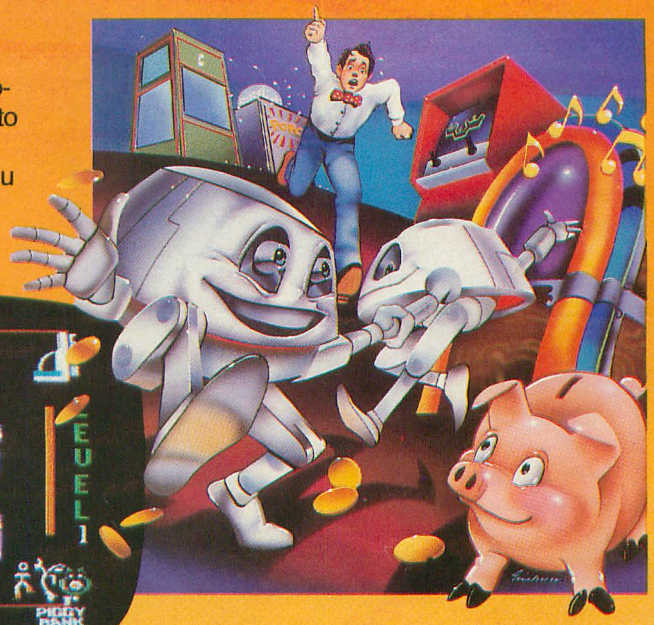


Figure 2 shows a simple flowchart for making a phone call. If you look at the flowchart closely, you will see that it is not very practical. What if there is an

A million laughs

SPARE CHANGE™ You are the game-happy owner of the Spare Change Arcade. Two fun-loving, but overworked Zerks—the main characters in your most popular game—have broken loose and are trying to retire from the business. You try madly to stop them. If you can get a coin into the juke box, the Zerks get so caught up in the music, they drop everything and start dancing. You also try popping popcorn and making a pay phone ring—which immediately makes the Zerks stop, answer and start a wild conversation. If you "win" the game, there are rib tickling cartoons by the Zerks to reward your efforts. It's a game full of sight gags, surprises and comedy. From the best. Brøderbund! **For the Apple® II/II+/IIe, Atari® Computers, and Commodore 64™ in disk format.**



and an endless challenge

LODE RUNNER™ Here's a game that will never stop challenging you. That's because Lode Runner is more than a spellbinding, fast-action game with its 150 different mind-boggling game screens. Lode Runner is also an easy-to-use Game Generator that lets you create your own games. Without any knowledge of programming, you can easily design unique Lode Runnner screens, then bring them to action-packed life. You will maneuver through scene after scene, running, jumping, drilling passages and outfoxing enemy guards in a secret underground hideaway as you pick up chests of gold stolen from citizens of the Bungeling Empire. There's no end to the thrills, chills and challenge. Of course, it's from Brøderbund! **For the Apple® II/II+/IIe; Atari® Computers; Commodore 64™ (disk and cartridge); VIC-20™ (cartridge); IBM® PC.**

Ask your Brøderbund dealer for sneak previews.

Brøderbund Software **Discover the Difference**

17 Paul Drive San Rafael, CA 94903

ATARI SINGS YOUR FAVORITE SONGS!!!

THE *Original* VOICE BOX Speech Synthesizer by the ALIEN GROUP has received rave reviews:

MICRO COMPUTING–"The VOICE BOX injects an endearing personality to your computer. The possibilities are enormous."

COMPUTE–"The VOICE BOX offers more human-like tones and does not blank out the screen."

CREATIVE COMPUTING–"English text and phonetic code may be freely intermixed rather than requiring separate modes as is the case without exception with every other speech system. A mode called talking face displays an animated face with impressive lip sync animation."

ANTIC–"There is a great potential for teaching children to spell and an added dimension to games overall. I believe the VOICE BOX is well worth the price tag."

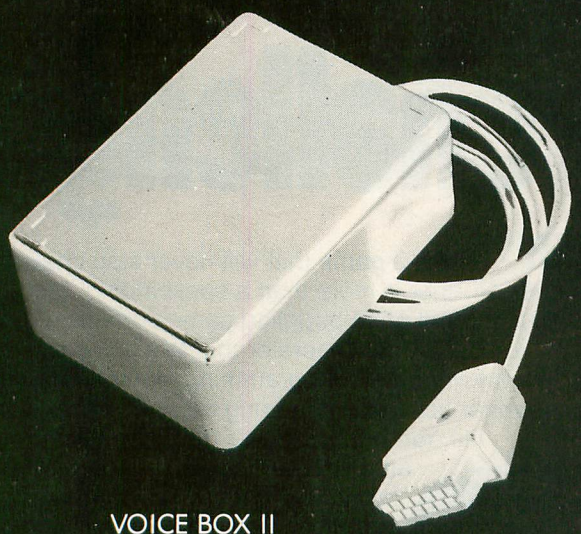
ANALOG–"For ATARI owners who want to add speech to their programs, the Alien Group VOICE BOX is probably the best choice."

POPULAR SCIENCE–"The speech quality is excellent. Besides creating speech, the software has a bit of fun with graphics."

and on the new VOICE BOX II.....

TIME MAGAZINE–"Machine of the Year" "The VOICE BOX by the Alien Group enables an ATARI to say aloud anything typed on its keyboard in any language. It also sings "Amazing Grace" and "When I'm 64" or anything else that anyone wants to teach it.

INCORPORATE THE SINGING HUMAN FACE
INTO YOUR PROGRAMS AND GAMES



VOICE BOX II
Speech & Singing Synthesizer

The New **VOICE BOX II** for **ATARI** plugs into the serial port of the **ATARI** 400/800 with sound coming out of the TV/monitor. 48K DISK is required. It has all of the features of the original **VOICE BOX** plus many exciting new hardware and software features:

- The ability to sing with voice and 3 part music.
- A library of 30 famous songs.
- A comprehensive music system that allows the user to easily enter or modify new songs.
- Software that can convert the bottom two rows of the **ATARI** keyboard into a piano with a range of 3½ octaves using the shift and control keys.
- Programmable musical sound effects such as tremolo, vibrato, glissando and click track.
- A singing *human* face with lip-sync animation designed by Jerry White.
- A talking or singing ***ALIEN*** face with software that allows the user to change the face and 8 mouth patterns as he sees fit.
- The ability to speak with inflection and feeling.
- Can speak in a foreign language with correct foreign spelling as input.
- A talk and spell program by Ron Kramer. Users can program any vocabulary for this spelling game. In fact, this program can even speak in a foreign language like French, where the user must spell the correct word in English, or vice versa.
- **GREEN GOBLINS**–A talking arcade game by John Wilson.
- Random Sentence Generator–An amusing grammar game that helps teach school children to identify parts of speech and recognize a variety of sentence structures.
- **NUMBER SPEAK**–A subroutine by Scott Matthews that converts up to a 9 digit number into normal English pronunciation. Ideal for building your own math games.
- **STUD POKER**–A talking poker game by Jerry White.
- The screen never blanks out while talking or singing.
- Singing or speaking subroutines can be incorporated into your programs, requiring as little as 100 bytes of RAM plus 5 bytes for each word.
- Price $169.00 includes **VOICE BOX II** and all of the above software.
- Inquire about our discounts for educational institutions.

To order by mail send a check or money order to the ALIEN GROUP for $169.

THE ALIEN GROUP
27 West 23rd Street (212) 741-1770
New York, NY 10010

ALSO AVAILABLE AT LEADING COMPUTER STORES THROUGHOUT THE WORLD.

answer but the person you want to talk to is not home at the time? It soon becomes obvious that even a simple phone call requires many decisions.

Figure 3

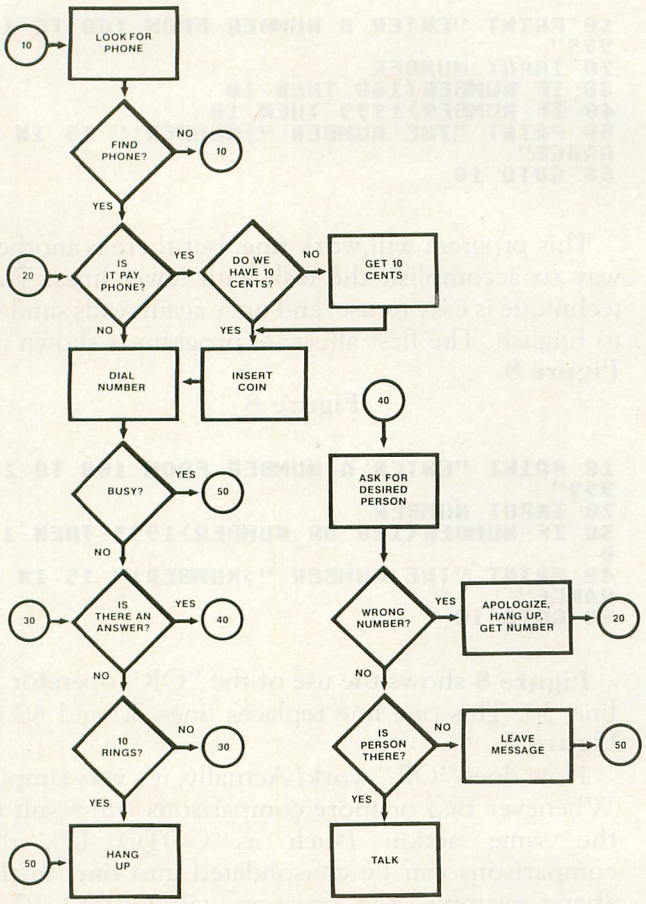


Figure 3 shows a more detailed phone call flowchart which contains 8 decisions. Note that each decision is structured to have a YES or NO answer. For example, the decision "IS IT A PAY PHONE?" has only two possible answers. If the phone is a pay phone, we must have 10 cents to make it operate. If it is not a pay phone, we can go ahead and dial the desired number.

Flowcharts are a valuable tool for outlining a program's vital functions. Unfortunately, most programmers are more excited about actually writing programs than sitting down and figuring them out. I hate to admit this, but I am as guilty as anyone of sitting down and writing programs at the keyboard, neglecting to plan the program out beforehand. The result is usually that a major restructuring of the program is necessary later to resolve unforseen problems. The moral: PLAN your programs—you'll be glad you did.

Decisions, decisions.

As pointed out earlier, even a simple phone call requires that several decisions be made by the caller in order to accomplish the task. It is not unrealistic to say that the normal person probably makes thousands of decisions every day.

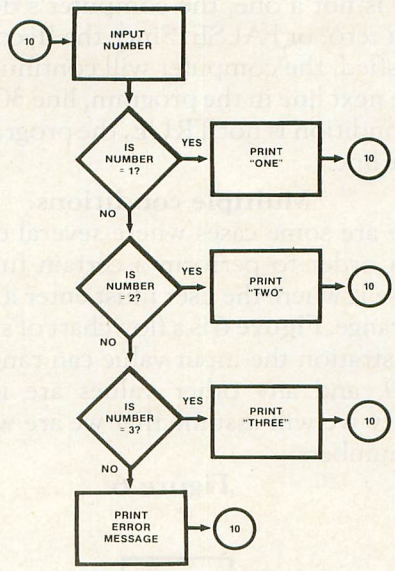
Your computer can make decisions, too, but quite a bit faster than any human. In fact, the ATARI computers, in machine language, can make hundreds of thousands of decisions every second! Of course, BASIC can't work this fast, but you can get the general idea — computers "think" circles around almost any human.

To let you help your computer make decisions, BASIC provides a program statement called "IF/THEN." This statement is incredibly versatile, and has many possible uses, depending on how it is structured. The general format of the IF/THEN statement is:

IF expression THEN (line number)
(statement [:statement. . .])

The best way to learn how the IF/THEN statement is used is to actually try it. Let's start with the simple example shown in the flowchart in **Figure 4.**

Figure 4



This program will accept a number from the keyboard. After accepting the number, three comparisons are made. The first checks the number to see if it equals one. If it does, a message is printed verifying the number and the program branches back to accept another number. If the number is not one, the computer checks to see if the number is a two. If so, an appropriate message is printed and another number accepted. If the number is neither one nor two, a comparison is made to see if the number is a three. If it is, a message is printed and the program branches back to get another number. If the number was not a one, two or three, a message is displayed and another number is accepted. **Figure 5** shows a

BASIC program corresponding to the flowchart in **Figure 4.**

Figure 5

```
10 INPUT NUMBER
20 IF NUMBER=1 THEN PRINT "ONE":GOTO 1
0
30 IF NUMBER=2 THEN PRINT "TWO":GOTO 1
0
40 IF NUMBER=3 THEN PRINT "THREE":GOTO
 10
50 PRINT "NUMBER NOT 1, 2 OR 3--TRY AG
AIN":GOTO 10
```

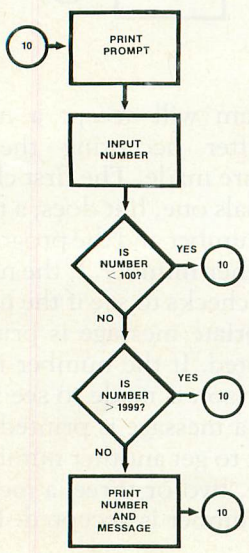
This program shows what is probably the simplest example of the IF/THEN statement. Lines 20, 30 and 40 are the comparison lines. Look at line 20. This line checks to see if the number entered was a one. If the number is equal to one then the program prints the word "ONE" and goes to line 10 to get another number. This comparison code reads almost like English, making the program very easy to understand.

What happens at line 20 if the number is not a one? Inside the computer, the results of comparisons are stored as either a one or zero, the binary number equivalent of TRUE or FALSE. In line 20, if the number is not a one, the computer's decision value will be a zero, or FALSE. Since the IF condition was not satisfied, the computer will continue processing with the next line in the program, line 30. Whenever the IF condition is not TRUE, the program will go to the next line.

Multiple conditions.

There are some cases where several decisions are made in order to perform a certain function. One such case is where the user must enter a number in a certain range. **Figure 6** is a flowchart of such a case. In this illustration the input value can range from 100 to 1999, and any other values are ignored. For simplicity we will assume that we are working with whole numbers.

Figure 6



We can handle this situation in several ways through BASIC. The first and most obvious is to have two separate IF/THEN statements, as shown in **Figure 7.**

Figure 7

```
10 PRINT "ENTER A NUMBER FROM 100 TO 1
999"
20 INPUT NUMBER
30 IF NUMBER<100 THEN 10
40 IF NUMBER>1999 THEN 10
50 PRINT "THE NUMBER ";NUMBER;" IS IN
RANGE"
60 GOTO 10
```

This program will work fine, but there is another way to accomplish the task with fewer lines. The technique is easy to use, and once again reads similar to English. The first alternate program is shown in **Figure 8.**

Figure 8

```
10 PRINT "ENTER A NUMBER FROM 100 TO 1
999"
20 INPUT NUMBER
30 IF NUMBER<100 OR NUMBER>1999 THEN 1
0
40 PRINT "THE NUMBER ";NUMBER;" IS IN
RANGE"
50 GOTO 10
```

Figure 8 shows the use of the "OR" operator in line 30. This one line replaces lines 30 and 40 in **Figure 7.**

How does "OR" work? Actually, it's very simple. Whenever two or more comparisons will result in the same action (such as GOTO 10), the comparisons can be consolidated into one. In the above example, the program will GOTO 10 if NUMBER is less than 100 OR NUMBER is greater than 1999.

The "OR" operator will result in a TRUE result if any condition in the IF/THEN statement is TRUE.

The comparison we made in the above example branched to line 10 if the number was out of the specified range. Another way to program the above test would be to see if the number is INSIDE the specified range. The flowchart for such a test is shown in **Figure 9**, and the BASIC code in **Figure 10.**

Figure 9

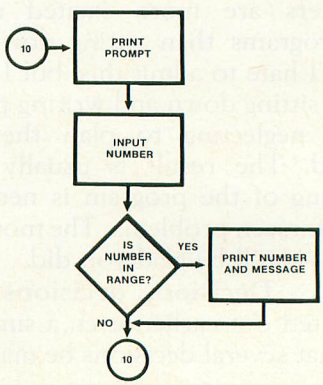


Figure 10

```
10 PRINT "ENTER A NUMBER FROM 100 TO 1
999"
20 INPUT NUMBER
30 IF NUMBER)=100 AND NUMBER(=1999 THE
N PRINT "THE NUMBER ";NUMBER;" IS IN R
ANGE"
40 GOTO 10
```

Looking at **Figure 10**, you will notice the user of the "AND" operator in Line 30. Unlike the "OR" operator, ALL conditions in a multiple comparison MUST be TRUE.. For example, if the number entered is 5, it satisfies NUMBER<=1999 but does not satisfy NUMBER>=100. Since the AND operator requires that both conditions be TRUE, the final result is that the IF statement is FALSE, and the program falls through to line 40. If the number entered is 540, it satisfies both NUMBER>=100 AND NUMBER<=1999, so the "in range" message will be printed. After the message is printed, the program continues to line 40, where the GOTO sends the program to line 10.

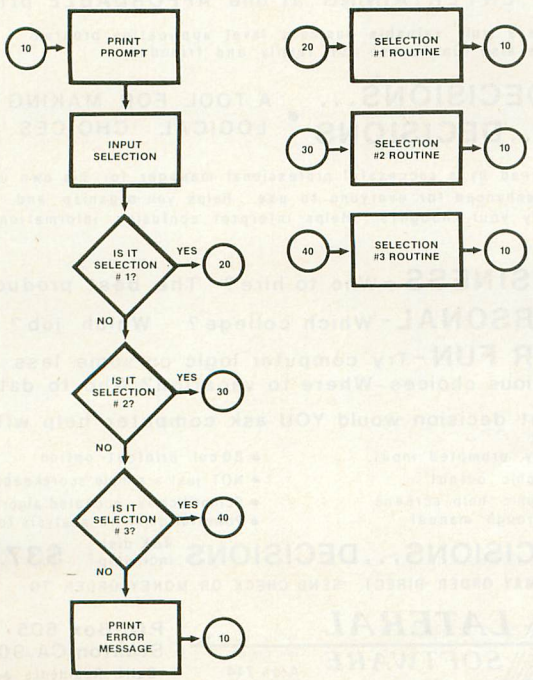
In this particular case, the AND operator saves us an additional line, with the program only using 4 lines total.

ON-ward and upward.

Many programs give the user several choices as to what the computer is to do. The list of options is usually referred to as a MENU. "Menu-driven" programs provide ease of use for people who are not familiar with computers.

Programming menu structures is very easy using the IF/THEN statement **Figure 11** shows the flowchart for a simple three-choice menu.

Figure 11



Now that we have covered IF/THEN statements, writing this routine in BASIC should be easy. **Figure 12** shows one of the ways this flowchart can be handled in BASIC.

Figure 12

```
10 PRINT "WHAT DO YOU WANT TO DO?"
20 PRINT
30 PRINT "   1. CALCULATE 2+2"
40 PRINT "   2. SAY 'HELLO'"
50 PRINT "   3. DO NOTHING"
60 PRINT
70 PRINT "ENTER THE NUMBER YOU WANT"
80 PRINT "AND PRESS RETURN."
90 INPUT NUMBER
100 IF NUMBER=1 THEN 1000
110 IF NUMBER=2 THEN 2000
120 IF NUMBER=3 THEN 3000
130 PRINT "OOPS -- TRY AGAIN!"
140 PRINT
150 GOTO 10
1000 PRINT "2+2 = 4"
1010 GOTO 10
2000 PRINT "HELLO!"
2010 GOTO 10
3000 PRINT "O.K."
3010 GOTO 10
```

Three IF/THEN statements are all that are needed for this simple application, but let's consider what could happen with other circumstances. What if there were 20 choices instead of only three? Using the IF/THEN technique would require 20 lines of code for a simple menu! Surely there must be a better way, right? Luckily for us, ATARI BASIC contains two handy statements, ON/GOTO and ON/GOSUB. These statements are essentially a series of IF/THEN statements wrapped into a single statement. Let's take a close look at how they work.

The format of the ON/GOTO and ON/GOSUB statements is as follows:

ON expression (GOTO) lineno [,lineno . . .]
(GOSUB)

The interesting thing about the ON/ statements is that, depending on the value of the expression, they will automatically GOTO or GOSUB the corresponding line in the line number list. If the expression evaluates to 1, the program will GOTO/GOSUB the first line number. If the expression evaluates to 2, the program will pass control to the second line number in the list, and so on. If the expression evaluates to 0 or is larger than the number of line numbers in the list, the program will fall through to the next statement in the program after the ON statement.

The ON/GOTO statement is perfectly suited for menu applications, especially when many selections are available. **Figure 13** shows the program from **Figure 12** which has been modified to use the ON/GOTO statement.

Figure 13

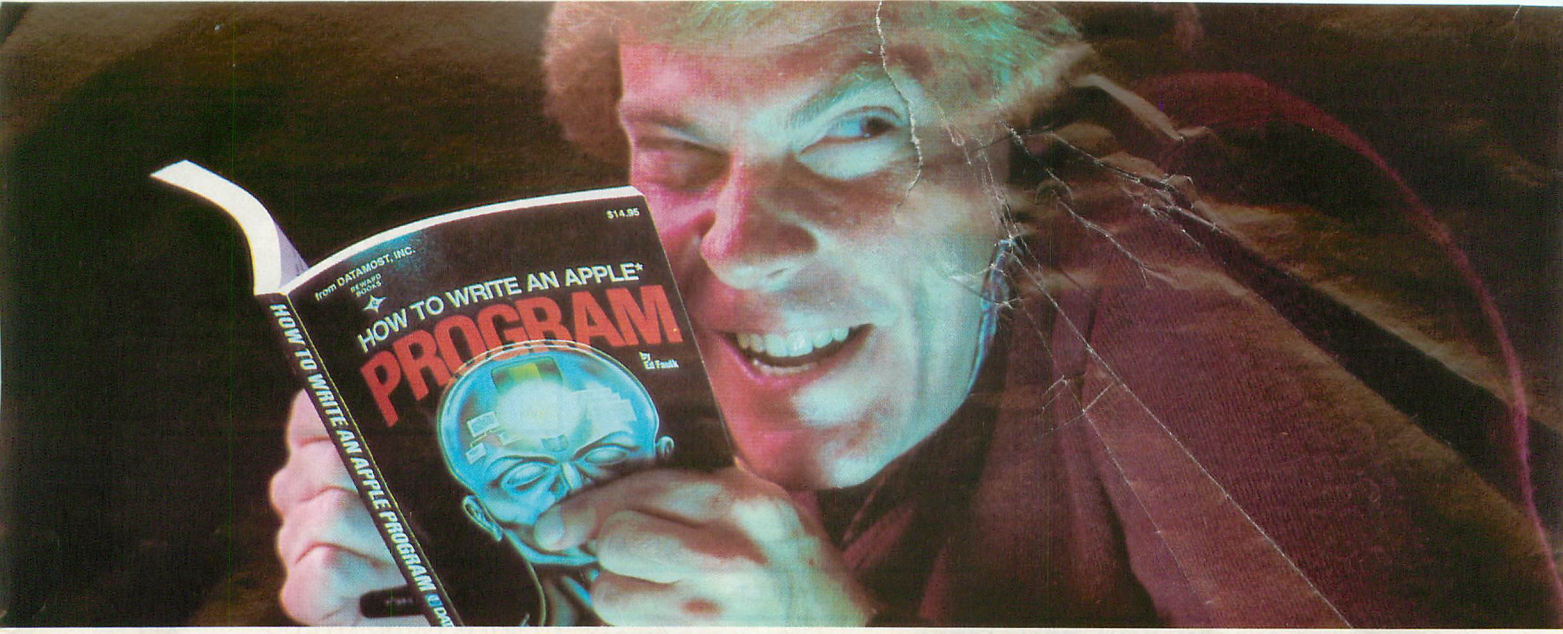
```
10 PRINT "WHAT DO YOU WANT TO DO?"
```

```
20 PRINT
30 PRINT "    1. CALCULATE 2+2"
40 PRINT "    2. SAY 'HELLO'"
50 PRINT "    3. DO NOTHING"
60 PRINT
70 PRINT "ENTER THE NUMBER YOU WANT"
80 PRINT "AND PRESS RETURN."
90 INPUT NUMBER
100 ON NUMBER GOTO 1000,2000,3000
110 PRINT "OOPS -- TRY AGAIN!"
120 PRINT
130 GOTO 10
1000 PRINT "2+2 = 4"
1010 GOTO 10
2000 PRINT "HELLO!"
2010 GOTO 10
3000 PRINT "O.K."
3010 GOTO 10
```

As you can see, lines 100, 110 and 120 in **Figure 12** have been replaced by line 100 in **Figure 13**. If NUMBER is 1, control passes to line 1000; if it is 2, control passes to line 2000; and if it is 3, control goes to line 3000. If NUMBER was not 1, 2 or 3, control passes to line 110, where the error message is printed.

It is easy to imagine how much memory (and typing!) can be saved by using the ON/GOTO statement whenever possible.

Next issue we'll look at non-relational and string comparisons and apply all we've learned with a program that gives Atari BASIC a PRINT USING capability. Until then, try to apply some of the concepts we've already covered in your own programs. □



Outsmart your computer.

Show your computer who's boss. Earn its respect. With a Datamost book.

No matter what age or ability level you're at, we have the right book that talks just to you. And your computer. Whether you own an Apple.* An Atari.* Or just about any brand.

All our books are incredibly easy to understand.

Which will make it incredibly easy to understand your computer.

What If You're Scared Of Books About Computers?

Don't be.

Our books are written in friendly, familiar American English. Highlighted with cartoons. And illustrations. So they're fun to read. As well as educational.

And there's over 30 books to choose from. Basic computer learning books to programming books to coloring books.

You won't be bombarded with complicated programming routines until you're ready for complicated programming routines. And no funny technical talk until you've reached the level where you don't think it's funny.

We'll tell you what you need to know. And then, when that's understood, you can take the next step toward outsmarting your computer.

Before You Spend Big Bucks On A Computer, Spend Little Bucks.

Once you've decided, kind of, almost, nearly, what type of computer you think you'd like to buy, buy a Datamost book.

It'll help you understand the Atari or the Commodore* or the Apple of your eye.

Before you spend a lot of money. Before you take your computer home.

You'll make a better purchase decision. Because you'll understand what you're doing. And what you and your computer can do together.

So there won't be any misunderstandings to ruin your new relationship.

How To Get The Most Out Of Your Computer.

Get the most out of our minds.

DATAMOST™ INC

The most out of our minds.™

Datamost books for every computer. Everybody. Every level. Pretty smart, huh?

Datamost, Inc., 8943 Fullbright Ave., Chatsworth, CA 91311, (213) 709-1202
*Atari is a trademark of Atari Computer.*Apple is a trademark of Apple Computer.
*Commodore 64 is a trademark of Commodore Business Machines, Inc.

ComputAbility™

Your ATARI Discount Software Store That's As Close As Your Phone!!

ATARI 800 48K	**CALL**
ATARI TOUCH TABLET	69.95
ATARI TRAK-BALL	49.95
COMMUNICATOR II	CALL
REMOTE CONTROL JOYSTICK	CALL
ATARI 800XL 64K	CALL
ATARI 1050 DISK DRIVE	CALL
ATARI 850 INTERFACE	CALL
ATARI 1010 RECORDER	77.00
HOME MANAGER KIT	64.95
ATARI 1027 PRINTER	289.00

MOSAIC

64K Ram/400	149.00
64K Ram + Cable Kit/400/800	169.00
48K Ram Kit	94.00
16/32 Expander	64.95
32K Ram	77.95
Mosaic Adaptor	49.95
64K Ram Utility Software - Superdrive & Handyman	Call!!

KOALA TOUCH TABLET 69.95

DISK DRIVES: **PERCOM, RANA, TRAK, ASTRA** ..... CALL FOR PRICES.

INTERFAST I
Programmable Printer Interface
w/4K Buffer 139.00

PRINTERS

PROWRITER 8510 AP	CALL
AXIOM AT-100 w/Parallel Port	229.00
GEMINI 10X	CALL
OKIDATA PRINTERS	CALL

Every month, we select the Super Specials

This month, YOU CAN PICK YOUR OWN!!!

USE THE CHART BELOW TO DETERMINE THIS MONTH'S SPECIAL PRICE!!

Koala Technologies Corporation · Broderbund Software · subLOGIC · SSI · Sirius · EPYX Computer Games Players Play · Thorn EMI Video · microcomputer games® a division of The Avalon Hill Game Company · Penguin Software The graphics people · SPINNAKER We make learning fun · FIRST STAR SOFTWARE · Roklan Software · Sierra On-Line Inc · DesignWare · SYNAPSE SOFTWARE · CBS Software · Datasoft Inc. Computer Software · MicroProse Software · EDUWARE® · MUSE® SOFTWARE · BRAM Inc. · DATAMOST Inc. · GAMESTAR · ACTIVISION® · SEGA THE ARCADE EXPERTS · Adventure International · MICROGRAPHICIMAGE · Lightning Software

IF THE RETAIL PRICE IS: *	SUPER SALE PRICE IS:
99.95	69.95
89.95	64.95
79.95	55.95
69.95	49.95
59.95	41.95
49.95	34.95
44.95	30.95
39.95	27.95
34.95	23.95
29.95	20.95

* **The following companies do not qualify for the Super Sale:**
- **ATARI**
- **APX**
- **PARKER BROTHERS**
- **ELECTRONICS ARTS**
- **LJK**

And still no surcharge for credit cards

Prices effective now through January 31, 1984

Mastercard/VISA Order Toll Free MasterCard **800-558-0003** VISA **In Wisc. Call 414/351-2007**

COMPUTABILITY
P.O. Box 17882
Milwaukee, WI 53217

ORDERING INFORMATION: To order by mail, send money order, certified check, or personal check (allow 14 days to clear) to COMPUTABILITY. Include $2.50 shipping on all software orders. Include 3% shipping on all hardware orders, minimum $2.50. Mastercard and Visa please include card number and expiration date. WI residents please add 5% sales tax. Canadian, APO & FPO software orders include 5% shipping minimum $5.00. All other foreign software, please add 15% shipping, minimum $7.00 (US Funds only). Prices subject to change without notice.

ORDER HOURS
Mon-Fri 11-7 PM (CST)
Sat 12-5 PM (CST)

TRANSPORTING ATARI COMPUTER PROGRAMS TO THE 5200

by Claus Buchholz

When Atari designed the 5200 "Supersystem" as a successor to the aging 2600 VCS, they made use of the state-of-the-art hardware they put into their 400/800 home computers. As a result, the systems are quite similar. The differences are great enough, however, that transporting programs from one system to the other requires some effort.

The 5200 is a single-board machine with four controller jacks, a cartridge slot, an I/O expansion connector and a power/RF cable. It shares the same VLSI chips with the 400/800, except for the 6520 PIA (joystick/parallel I/O ports). The other chips (ANTIC, GTIA, POKEY and the 6502 CPU) are in the 5200, although some of their registers are in different memory locations than those in the 400/800. Also, some of the registers serve slightly different purposes. **Figure 1** is a block diagram of the 5200. Let's look at each section of the hardware in turn.

When a specific register is mentioned in the article, the name is taken from the Hardware Manual in the 400/800 Technical Reference Notes. With the exceptions listed in this article, the Hardware Manual applies also to the 5200.

6502 CPU.

Although it is a standard 6502 from the programmer's view, Atari has reworked this chip to save four support chips. Those four chips mediate access of the system bus by the 6502B and ANTIC in the 400/800, but that function is built into the 6502C in the 5200. Atari also uses the 6502C in the new XL series of computers.

16K RAM.

The 5200 contains 16K bytes of RAM addressed from $0000 to $3FFF, just as in an unexpanded 400. The memory circuit is nearly identical to that in the 400, except it uses 4516s, a 5V-only version of the 4116 16K-bit dynamic RAM.

The 5200 monitor program reserves locations $0000 through $0018 and $0200 through $021B for shadows and RAM vectors. And, of course, page $01 is reserved for the 6502 stack. The rest of the RAM is available to the cartridge program.

ANTIC.

This is the same ANTIC chip used in the 400/800. Since ANTIC shares the address bus with the CPU and has no chip select input, its registers reside in the same place in the 5200's memory as in the 400/800's, page $D4.

The 5200 has no System Reset key, so bit 5 of NMIST is useless and the 5200's interrupt handler ignores it.

GTIA.

The GTIA and its registers perform the same functions in the 5200 and 400/800, except as noted below. The registers, however, appear at page $C0 in the 5200's memory, not at $D0 as in the 400/800.

The trigger inputs, TRIG0 through TRIG3, are wired to the controller ports, one to a port. The bottom button on either side of the leftmost controller zeroes the TRIG0 register when pressed, and likewise for the other ports.

The bits in CONSOL, the 400/800's console switch port (START, OPTION, SELECT and speaker), are used as outputs in the 5200, Bit 3, the 400/800's

speaker control, can still be toggled in the 5200 to produce sounds through the TV speaker. Bit 2 controls the pots in the joystick controllers. It must be set high to enable the pots.

Bits 1 and 0 select which controller port is to be active at one time. 00 selects port #1 (the leftmost), 01 selects #2, 10 selects #3, and 11 selects #4. The trigger buttons and pots are independent of this selection; it applies only to the keypads and top side buttons on the controllers.

POKEY.

POKEY's registers are all addressed at page $EB in the 5200 as opposed to $D2 in the 400/800. Its functions are unchanged, however, except for two.

The eight pot inputs used for paddles in the 400/800 are wired to the 5200's controller ports, two to a port. Each controller has an analog joystick, using one port to sense horizontal position and a second pot for vertical position. The even pots (POTO-POT6) give the horizontal positions of range from 1 to 228; the maximum readings are to the right for the horizontal pot and at the bottom for the vertical pot. **Figure 2** shows the pinout for the 5200 controllers.

The keyboard scanning lines in the 400/800 are used in the 5200 to read the keypad keys to the one controller that is selected by bits 1 and 0 in CONSOL. Only four lines are used, though, so only bits 1 through 4 of KBCODE are valid. **Table 1** gives the keycode for each key on the controller. The top side buttons on the selected controller act like the 400/800 shift keys and also cause a BREAK-key interrupt, if that's enabled. Bit 0 of SKCTL, the debounce enable bit, need not be set in the 5200.

Table 1. Keypad Codes.

Key	KBCODE bits 4-1	Keypad code
none	0000	$FF
#	0001	$0B
0	0010	$00
*	0011	$0A
Reset	0100	$0E
9	0101	$09
8	0110	$08
7	0111	$07
Pause	1000	$0D
6	1001	$06
5	1010	$05
4	1011	$04
Start	1100	$0C
3	1101	$03
2	1110	$02
1	1111	$01

POKEY's serial I/O lines are unused in the 5200, but they are wired to the I/O expansion connector, an edge connector hidden behind a small door in the rear of the 5200. This connector allows for more hardware registers addressed at page $EO, and for peripherals using the serial port. See **Figure 3** for the pinout of this connector. Its existence demonstrates Atari's original plans to expand the 5200 system.

ROM.

The 5200 has a 2K ROM on board which holds the character set and monitor program. The character set, which is an exact copy of the 400/800's set, resides at pages $F8 through $FB, and the monitor sits at $FC through $FF.

The cartridge ROM can be 32K bytes long and resides in memory from $4000 to $BFFF. **Figure 4** shows the pinout of the cartridge slot. The two interlock connectors are wired together on a cartridge board. The 5200 uses this as a switch for the cartridge's power connections and as a Reset signal. Therefore, a cartridge may be safely removed or inserted while the 5200 is powered on.

The 5200 monitor program.

The 1K monitor program in ROM has three functions: (1) to initialize the system before running the cartridge program, (2) to service interrupts as they occur, and (3) to maintain shadows of some important hardware registers. Recall that the 400/800 Operating System is 10K bytes long and performs the above functions. It also provides peripheral handlers, predefined graphics modes, a screen editor, and floating-point math routines. Those utilities do not exist in the 5200.

Table 2 shows the RAM locations used by the monitor for shadows and RAM vectors.

Table 2. 5200 Monitor RAM Locations.

Page Zero Locations

$00	Shadow for IRQEN
$01	Real time clock (high byte)
$02	Real time clock (low byte)
$03	Critical code flag (if non-zero, VBI routine is abbreviated)
$04	Attract mode timer/flag
$05	Shadow for DLISTL
$06	Shadow for DLISTH
$07	Shadow for DMACTL
$08-$10	Shadows for COLPMO through COLBK
$11-$18	Shadows for POTO through POT7

Page Two Vectors

$200	Immediate IRQ vector
$202	Immediate VBI vector
$204	Deferred VBI vector
$206	DLI vector
$208	Keyboard IRQ vector
$20A	Keypad routine continuation vector
$20C	BREAK key IRQ vector
$20E	BRK instruction IRQ vector
$210	Serial Input Data Ready IRQ vector
$212	Serial Output Data Needed IRQ vector
$214	Serial Output Finished IRQ vector
$216	POKEY Timer 1 IRQ vector
$218	POKEY Timer 2 IRQ vector
$21A	POKEY Timer 4 IRQ vector

Upon Reset, the 6502 vectors through $FFFC to the initialization routine. This routine performs the following sequence:

1. Disable maskable interrupts, clear the 6502 decimal flag, and set the stack pointer to $01FF.

2. If the cartridge address $BFFD contains $FF, then jump immediately through the vector at $BFFE (diagnostic cartridge).

3. Zero all hardware registers and page $00, set CHBASE to point to the character set at $F8, and initialize the first six RAM vectors starting at $0200.

4. Set up the Atari logo rainbow display. The cartridge title (20 characters) and copyright year (2 characters) in ANTIC mode 7 display code are taken from cartridge addresses $BFE8 through $BFFD.

5. Enable VBI (Vertical Blank Interrupt) and DLI (Display List Interrupt), and enable key scan.

6. Wait four seconds, then jump through the vector at $BFFE to the cartridge program.

When the 6502 receives a non-maskable interrupt (NMI), it vectors through $FFFA to the NMI handler. The following steps take place:

1. Check NMIST and strobe NMIRES to reset the interrupt status.

2. If a DLI is pending, jump through the DLI vector (initialized to point to the rainbow effect routine).

3. If a VBI is pending, jump through the immediate VBI vector (initialized to point to the VBI routine).

4. Else, return from the interrupt (no System Reset).

A cartridge program can change these vectors to point to its own DLI and VBI routines, if it must. The default VBI routine takes the following action:

1. Push A, X, and Y onto the stack, increment the real time clock, and update the attract mode timer.

2. If the critical code flag byte is non-zero, then pop Y, X, and A from the stack and return from the interrupt.

3. Update DLISTL, DLISTH, and DMACTL from their shadows.

4. Maintain the attract mode flag and update the GTIA color registers from their shadows.

5. Update the pot shadows from POTO through POT7, and strobe POTGO to start another pot scan.

6. Jump through the deferred VBI vector (initialized to point to the end-of-interrupt routine, which pops Y, X, and A, and returns from the interrupt).

If maskable interrupts (IRQs) are enabled and one is received, the 6502 vectors through $FFFE to an instruction which jumps through the immediate IRQ vector. That vector is initialized to point to the IRQ routine, which performs the following tasks:

1. Push A and check IRQST.

2. For each of the 8 bits in IRQST, check for a pending interrupt. If found, then clear and status bit, update IRQEN from its shadow, and

jump through the appropriate IRQ vector.

3. If no interrupt found, then push X and check for a BRK instruction interrupt. If found, then jump through the BRK instruction IRQ vector.

4. Else, pop X and A and return from the interrupt.

The only IRQ vector that is initialized is the keyboard IRQ vector, which points to the keypad read routine. That routine does the following:

1. Push X and Y.

2. Read KBCODE and mask bits 1 through 4.

3. Convert to the keypad code given in **Table 1**, leaving that code in A.

4. Jump through the keypad routine continuation vector (initialized to point to the end-of-interrupt routine).

Comparing the 5200 monitor's vectors to the 400/800's OS vectors, we see that Atari paid no attention to compatibility between the two. This further complicates the task of converting a program from one system to the other.

Transportability.

It would not be difficult, given the information in this article, to write a program in two versions, one for the 400/800 and another for the 5200. Nor would it be difficult, given the source code, to convert a finished program from the 5200 to the 400/800. The reverse is more difficult if the program takes advantage of special features in the 400/800 OS. Otherwise, the only task, aside from redefining some addresses, is to convert the keyboard/joystick input routines from one system to the other.

I acquired the information in this article by dissecting a 5200 and disassembling its ROM. The 400/800 schematics in the Hardware Manual were quite helpful. It is interesting to note the differences between the two machines and to guess Atari's motives for the design differences. But the similarities grossly outweigh the differences, so that a 5200 program can be developed and almost entirely debugged before testing on a 5200. With the addition of an EPROM burner, a 400/800 can be a powerful development system for 5200 programs. An adventuresome hacker can even bypass the EPROM by putting dual-port RAM on the cartridge board and downloading programs from the 400/800 development system into the 5200 for testing. □

(See charts on next page.)

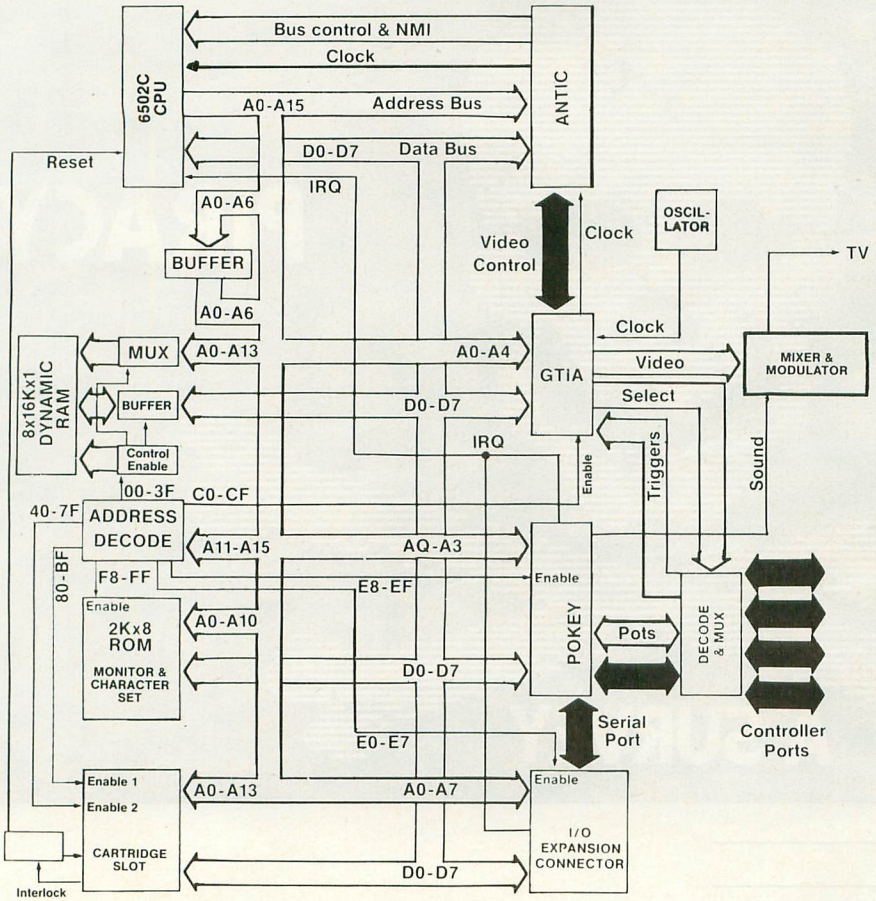
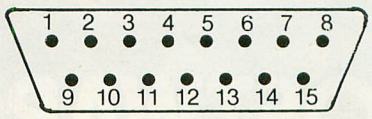


Figure 1.



Pin	Function
1	Keypad - right column
2	Keypad - middle column
3	Keypad - left column
4	Start, Pause, and Reset common
5	Keypad - third row and Reset
6	Keypad - second row and Pause
7	Keypad - top row and Start
8	Keypad - bottom row
9	Pot common
10	Horizontal pot (POT0,2,4,6)
11	Vertical pot (POT1,3,5,7)
12	5 volts DC
13	Bottom side buttons (TRIG0,1,2,3)
14	Top side buttons
15	0 volts - ground

Figure 2.

Controller Port Pinout.

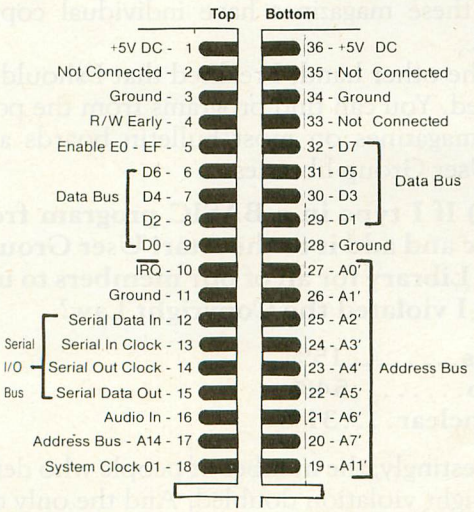


Figure 3.

I/O Expansion Connector

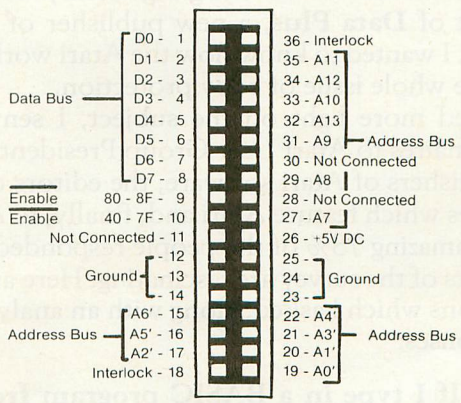
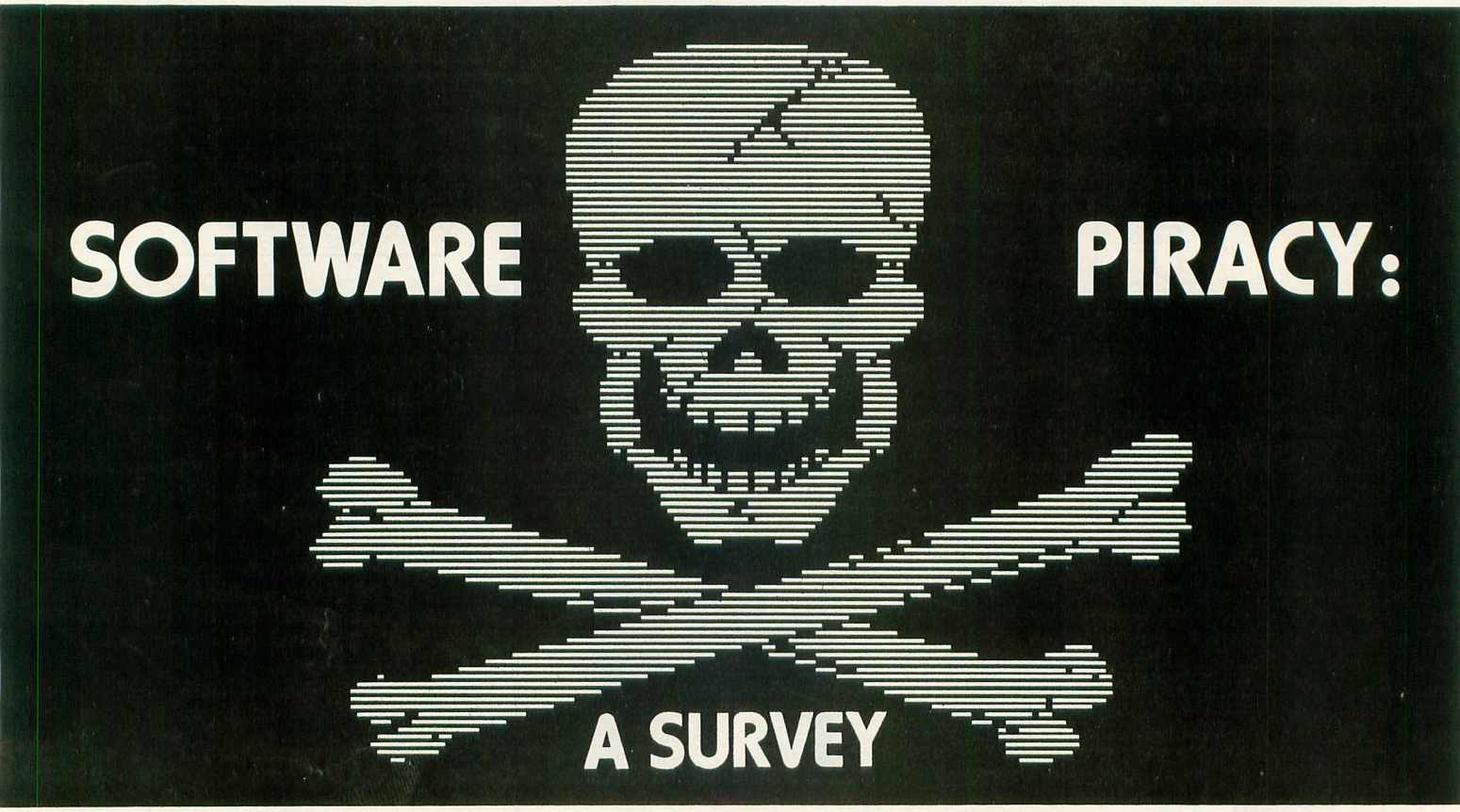


Figure 4.

5200 Cartridge Slot



by Allen Harberg

Is software piracy, like beauty, in the eye of the beholder? Is there a consensus in the Atari community as to what constitutes a violation of the copyright laws?

I needed answers to these questions for a couple of reasons. As President of our local Atari Users' Group, I needed to know which programs we could properly include in our fledgling library. And as President of **Data Plus**, a new publisher of Atari software, I wanted to know how the Atari world felt about the whole issue of copy protection.

To shed more light on the subject, I sent 100 questionnaires to Atari User Group Presidents, fellow publishers of Atari software, the editors of the magazines which feature Atari and, finally, to Atari, Inc. An amazing 78% of the people responded, and the results of the survey are fascinating. Here are the 7 questions which I asked, along with an analysis of the responses:

(1) If I type in a BASIC program from Compute!, save it on disk, and give the disk to a friend, have I violated the Copyright Law?

Yes 9%
No.....62%
Unclear...29%

On the one hand, I was amazed that almost two-thirds of the respondents felt that there was nothing wrong with giving a friend a program which you had typed in from a magazine. After all, **Compute!**, like most of the microcomputer magazines, is copyrighted. In addition, the majority of programs within these magazines have individual copyright notices.

On the other hand, I realized that I shouldn't be surprised. You can find programs from the popular micro magazines on most bulletin boards and in Atari User Group libraries.

(2) If I type in a BASIC program from Antic and add it to the Atari User Group's Disk Library for all of our members to use, have I violated the Copyright Law?

Yes.....15%
No.....54%
Unclear....31%

Interestingly, the number of people who detected a copyright violation doubled. And the only difference was the number of people who had access to the program. Evidently, a number of people believe that small give-aways are okay, but you run into trouble when your volume increases. Almost one third of the respondents continue to be unclear about the

legality of swapping programs which were typed in from magazines.

(3) If I buy the Disk Version of Softside and give one of its BASIC programs to a friend, have I violated the Copyright Law?

Yes.....45%
No.....16%
Unclear...39%

While a lot of people remained uncertain, a number of people "got religious" when they thought about the Disk Versions which some magazines provide. Unfortunately, there are 2 differences between this question and question #2. First, the consumer didn't type in the program; they received it on disk. Second, **Softside** includes a program on their Disk Version whose listing does not appear in the magazine. Despite this complication, people seem to have more respect for the copyright status of programs which they receive on an electronic medium.

(4) If I buy the Disk Version of ANALOG Computing and add it to the Atari Users Group's library for all of our members to use, have I violated the Copyright Law?

Yes.....55%
No.....31%
Unclear...14%

The good news is that most people have a 'clear' answer to this question. The bad news is that there is no consensus as to what that answer should be!

Almost one third of the respondents feel that it is okay to move **ANALOG Computing's** Disk Version directly into their club library...despite the copyright notice in the magazine and in a large portion of the individual programs contained on the disk. More than half of the people feel that it is improper to distribute these programs through the club.

(5) If I buy a copy of Byte and lend the magazine to a friend, have I violated the Copyright Law?

Yes.....0%
No.....77%
Unclear...23%

While I was happy to see that nobody considered this to be a copyright violation, I was truly surprised to learn that nearly one quarter of the respondents weren't sure. I'll bet that **Byte** sends thousands of

subscriptions to university-, business-, and public-libraries for the sole purpose of allowing them to lend the magazine to their 'friends.' Most authorities are in agreement that there is no violation of the law here.

(6) If I buy a copy of Micro and make a photocopy of an article for a friend, have I violated the Copyright Law?

Yes.....54%
No.....16%
Unclear...30%

Photocopying of copyrighted magazine articles was a major problem in the early sixties, one which received as much attention as the 'software piracy' issue receives today. At that time, the question was asked, "How could the small-sized, small-circulation, high-priced journals survive if libraries had photocopy machines?" Rarely will you find a photocopy machine in a public library which does not have a summary of the Copyright Law nearby.

The Copyright Law states that you have the right to make a personal copy of a copyrighted article; you do not have the right to give that photocopy to a friend.

(7) In their normal, day-to-day activities, do Atari User Groups violate the Copyright Laws?

All the time.....0%
Most groups do; some don't.....38%
Some groups do; most don't.....46%
Never knowingly.....16%

Statistically, you'd have to call it a tie. Most people believe that a lot of Atari User Groups spend a lot of their energy swapping copyrighted code.

I hope they're wrong. User Groups who overtly exchange copyrighted programs are leaving themselves extremely vulnerable to prosecution. And anyone who has followed the way in which Warner Communications, Atari's parent company, addressed the issue of pirated videotapes would realize that Warner has no sense of humor when it comes to people stealing their products.

Some of the survey results don't pop out at you when you examine the raw data. For example, in analyzing the survey results, I found a strong correlation between:

- *People who felt that the situations described in questions 1-4 did not constitute violations of the copyright law, and*
- *People who gave the Atari User Groups a clean bill of health*

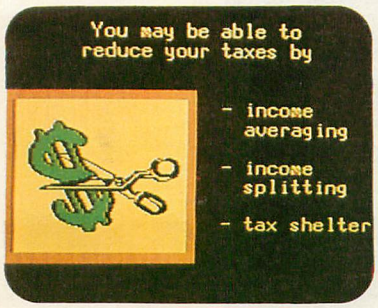
Essentially, people who took the 'see no evil' approach to software piracy were consistent in their perceptions.

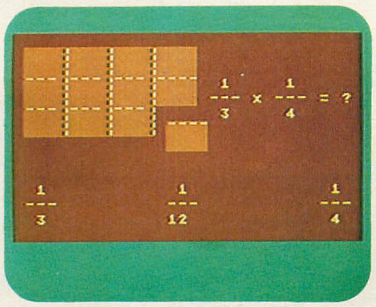
I received a response from Atari, Inc. More specifically, I received a response from an employee from Atari, who pointed out in no uncertain terms that he was expressing his own feelings and not those of his employer. Understandably, his response reflected the conservative attitude which Atari takes in the field of software piracy.

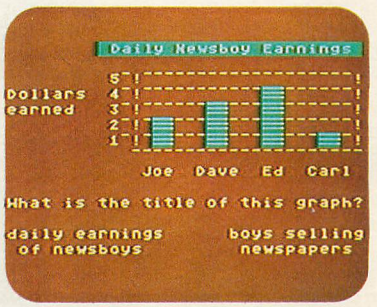
I continue to be amazed by the tenacity of the pirates. There are software publishers who market disks which contain 500 unformatted disk sectors. To make an illegal copy of such a disk, you would have to search a sea of unformatted sectors to find a single sector of binary zeroes. Yet I know that there are individuals who will spend untold hours listening to their drives make the most horrifying sound imaginable while they search for an oasis in a desert of unformatted tracks. Eight hours and 1000 I/O errors later, they've created a back-up of a program which retails for a $29.95! I guess that using leisure time effectively, like beauty, is in the eye of the beholder. □

NEW! For Your Atari Computer

TRS 80 Color, I, III, 4 or Apple Computer







Over 1000 Programs with Full Time Audio Narration, Pictures & Text!

Interactive Tutorial Programs

All along you've heard there isn't much of a selection of low cost microcomputer courseware. Well, do you want to learn Physics, Psychology, or Philosophy? Accounting or Auto Mechanics? Sociology, Supervision or Statistics? Economics or Electronics? How about English as a Second Language? Would you like to teach your children Math or Reading? If you do, then we have the educational programs for you—and at affordable prices.

It's true there aren't many companies that offer full-length courses in subjects other than reading and arithmetic, and what is offered seems to be drills, tests, games, or simulations. What you really want is a course that covers the subject with, say, 16 full-length lessons called tutorial programs, where you interact with an expert programmer backed by a staff of experts. That's exactly what we have.

Is there something wrong with our software? Well, we don't ask you to enter your name so we can drop it into some later text. We don't ask you to type in your answer and refuse to accept it if it's not spelled just right. And, we don't branch around a lot when you make an error. Our programs simply let you know if you're wrong by proceeding only when you select the right multiple-choice answer.

This proven learning-by-positive-reinforcement method lets you proceed quickly and smoothly through the programs, without a lot of cute tricks. But, if you're a devoted computerist or game freak you may be disappointed at the lack of motion in most of the pictures, or our special visual effects. We have some dandies, like the mushroom cloud that rises over Hiroshima in our History series, but your learning is not distracted by needless special effects.

OVER 1000 PROGRAMS

We have 64 courses of 16 half-hour programs: 1024 programs! All are easy-reading, upper- and lower-case. All are in color. All are illustrated by frequent graphics composed of special and regular characters. And, best of all, every frame of every program is accompanied by high-quality, full-time audio narration by professional voice talent. David Stanton, James Mathers, Pam Barrymore. Recorded and played back, not synthesized or digitized!

— If your child is having trouble with Reading or Math, or if he or she is exceptional, and could benefit from professionally programmed lessons or courses, then you need our educational programs for your Atari, Apple, or TRS 80 to help your child.

— If you want to build your math skills, we have programs on Numbers and their meanings, Addition, Subtraction, Multiplication, Division, Fractions, Decimals, Percents, Angles, Graphs, Word Problems, Algebra, Statistics. There are ten series in all, 160 math programs. Again, all with color, pictures, and a friendly tutor's voice—all the time.

— If your verbal skills need polishing, try our Talk & Teach programs on the alphabet, spelling skills, and every level of vocabulary; Sight Words or Learn by Phonics; develop Reading Comprehension skills from stories and articles. There's even a series on the Great Classics.

— For self-development, try a 16-program course in Economics or Psychology. Or Supervision, Sociology, Counseling (Personal, School, or Employee), Writing, Business, Philosophy, Government, World History, U.S. History, or Accounting.

— For vocational skills, study our 16-program courses in Electronics, Fluid Power, Auto Mechanics, Shop, Carpentry, Construction, Meat Processing, Military Skills (64 programs), and there are many more.

There are 832 programs for your Atari, Apple and TRS 80 Mod I/III/4 and 160 for the Color Computer. Send for a free catalog.

Would you like to do something more than play games on your computer? Tired of squeaks and robotics and want to hear a human voice? Do you want to further your education or help your children along in theirs? Now you can. For just **$8.80**, we'll send you one cassette with 2 programs from the course of your choice, 100% guaranteed to work in your Atari, Apple, or TRS 80 computer, and you can try us out. Better yet, get one full course of 16 programs on 8 cassettes for only **$59.90**. Atari Master Cartridge, **$9.95**; Apple T/T Board, **$99.00**; and T/T Player, **$79.00**. That's less than books and tuition for most college classes. And we offer a 10 day, 100% exchange allowance. Does your alma mater? Send us your check, or call us at **1-800-654-3871** with your Visa/Master Card number. We'll send your educational software pre-paid. Please allow 15 days delivery. Or see your computer dealer. He may have some of our courses in stock. You've got nothing to lose but your games!

We're Your Educational Software Source.

Course	No. of Programs
Reading	256
Mathematics	128
Comprehension	48
History	32
Algebra	16
Spelling	16
Government	16

16 Programs in each of the following:
Carpentry-Electronics-Health Services
Office Skills-Statistics-First Aid/Safety
Economics-Business-Accounting-Psychology
And Many More!

For your Atari 400/600/800/1200, you will need the Atari cassette recorder and the Dorsett 4001 Educational Master Cartridge, $9.95. For your Apple II, you will need the Dorsett M402 T/T plug-in board, $99.00, and the M401 stereo cassette player, $79.00. All programs listed are available for TRS 80, I, III, 4, which require the M203 speaker converter, $99.00, and 401 stereo cassette player, $79.00.

Send for a catalog of over 1000 programs for Atari, TRS 80, Apple, etc.

For more information, write or call: **TOLL FREE 1-800-654-3871**

Dealer inquiries welcome

VISA MasterCard

DORSETT
Educational Systems, Inc.
P.O. Box 1226, Norman, OK 73070

Craftsmen Need Precision Tools . . . Programmers! Demand Precision Software!

BASIC XL has twice the speed and twice the power of Atari® BASIC. And yet, as befits a fine craftsman's tool, BASIC XL is even easier to use and more dependable, while including such outstanding major additions as structured programming, string arrays, programming aids, enhanced graphics, and business capabilities.

Atari BASIC is a good starting point. We should know. We wrote it in 1978. Buy BASIC XL. Take advantage of five more years of experience!

So, prepare yourself for some exploration into imaginative programming with BASIC XL! Cartridge, excellent tutorial, reference manual . . . $99.

Functional, honest, and beautiful describe the simple lines of a craftsman's tools. For the jeweler these tools are an extension of the human hand to better execute complex designs. For you, the programmer, Precision Software tools keep complications out of your programming while allowing you to produce intricate programs.

See the complete collection of OSS Precision Software™ tools!

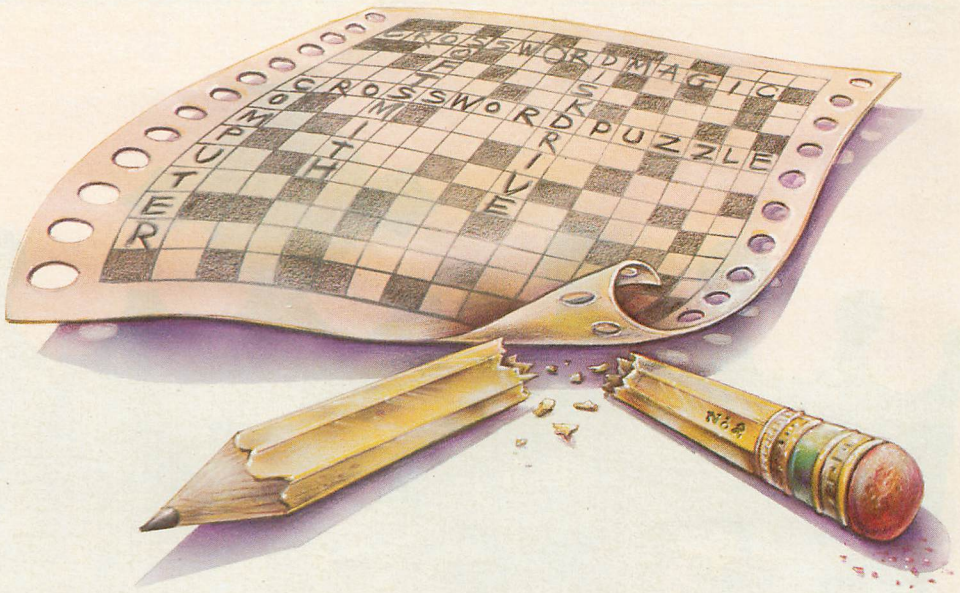
MAC/65: The fastest 6502 macro assembler/editor package on cartridge . . . $99.

BUG/65: A powerful debugger. On disk, with OS/A+ . . . $35.

C/65: The first native mode "small c" compiler for Atari and Apple® computers. On disk . . . $80.

ACTION!: The fastest, small computer language ever. A feature-packed cartridge at only . . . $99.

PRECISION SOFTWARE
A DIVISION OF O.S.S., INC.
1173D S. Saratoga/Sunnyvale Road
San Jose, CA 95129 • (408) 446-3099



CROSSWORD MAGIC™
A GAME DESIGNED TO GET YOU DOWN (AND ACROSS)

What's a five-letter word meaning report? The capital of a Baltic country, 6 letters, starts with W? If these questions get you thinking, you've obviously got the crossword bug. There is no cure, but Crossword Magic, from Softsmith™ Corporation, is just the treatment the doctor ordered. It is a challenging, intelligent game that will help you improve your vocabulary while you're having fun.

With Crossword Magic, you create your own puzzles to challenge your friends. First you choose a topic; then you enter words using the keyboard. Crossword Magic automatically fits the common letters together to form a familiar crossword grid. If there is no opening for the word you enter, the program saves it and fits it in later.

Next comes the fun. After the puzzle is full, Crossword Magic prompts you for a clue to go with each word. Make your clues as obscure, humorous or serious as you want. You can save the puzzle and clues on disk for later play on the screen, or have them printed out to send to friends.

Crossword Magic runs on the Apple® II, II + and IIe; Franklin Ace™ and Atari 800® computers with 48K RAM. A disk drive is required; printer is optional.

More than just a game, Crossword Magic is also an excellent educational tool for helping children of all ages improve their spelling and vocabulary skills. It is one of the programs in the Softsmith library of quality software — the largest library of programs under one brand name.

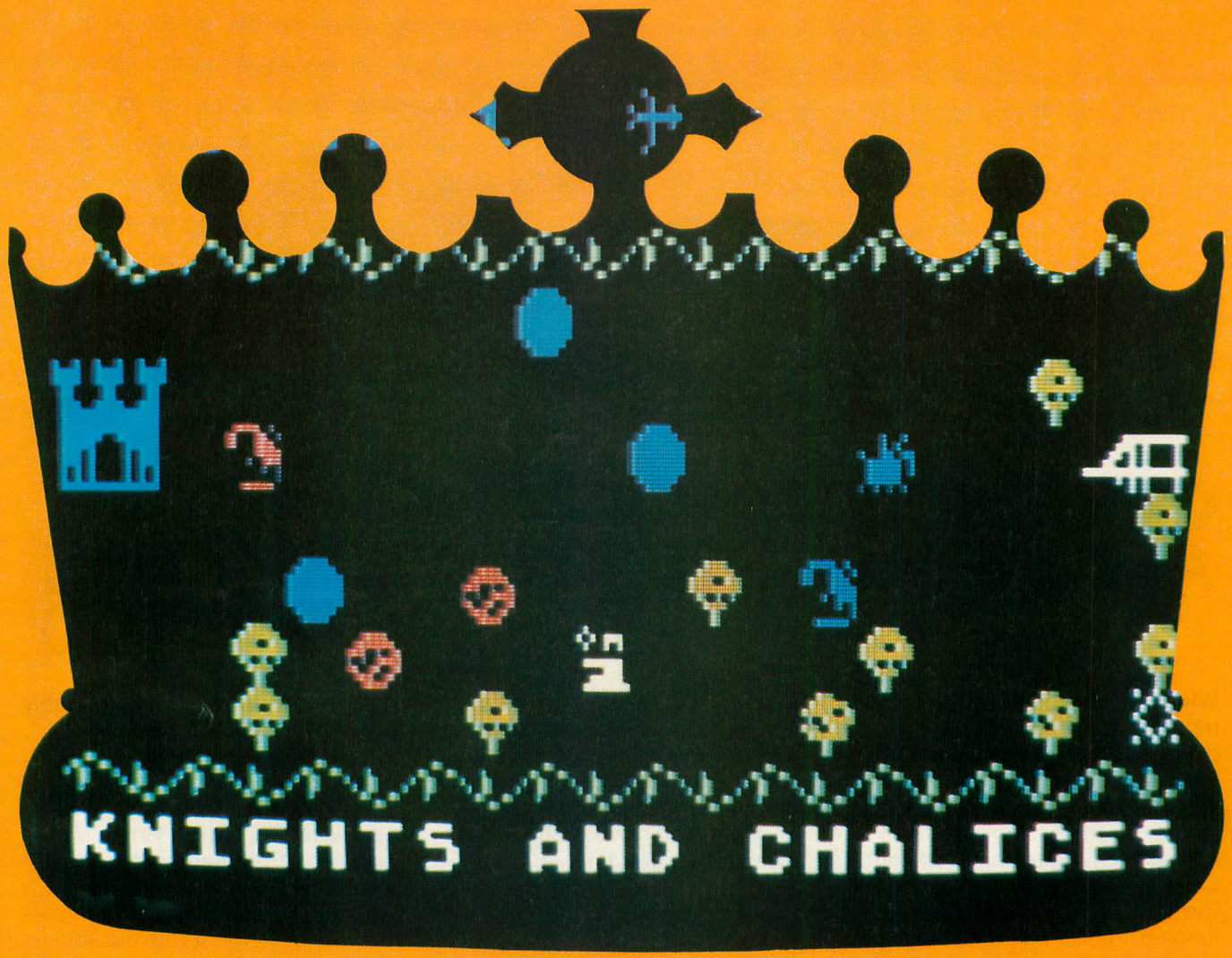
Softsmith has the most thoroughly tested, refined and clearly documented brand of personal computer software you can buy. You can depend on Softsmith programs for Education, Home Management, Entertainment, Word Processing, Business, Communications and Programming. And we back them with our Toll-Free customer service number.

Ask for Softsmith brand software wherever computers or software are sold. Or call us Toll-Free at (800) 341-4000 for the name and location of the dealer nearest you.

Dealer inquiries are invited.

SOFTSMITH™

SOFTWARE LIBRARY



Knights & Chalices

24K
Cassette
32K
Disk

by
Bruce Willard

Your King has been robbed by an evil Sorcerer of his Magic Chalice and, as a further insult, locked out of his own castle.

You are selected as his favorite Lord (or Lady) to retrieve the Chalice and the key to open his castle gates.

There are many obstacles in your path. Dragons, enemy archers, a troll, lakes and lava pools abound, not to mention the Evil Sorcerer himself. Of course there are also objects to aid your search, such as swords, a bow and arrow, and a magic necklace. However, you must have the right defense for the danger threatening you.

In the One Player game, your objective is to obtain the Magic Chalice and castle key while being delayed as little as possible by the dangers awaiting you. Then you must return to the castle where you will receive a Coat of Arms appropriate to your efforts.

In the Two Player game, you must find the Magic Chalice and key and return to your Castle before your opponent. If you should run into something nasty without the proper protection, you will lose a number of turns or be transported. An enemy archer will send you to the beginning of the screen you are on, and the Evil Sorcerer will send you to his abode at the corner of the screen.

There are two methods of using the joystick. The first, advised for more accurate movement and when first playing the game, is to move your joystick in the direction you wish to travel. Your Knight will then appear, flashing in that position. If that is the position you wish, then push the button. Your move is then made, If not, then just move your joystick in another direction.

The second method is to keep your button depressed while moving the joystick in the desired directions. Your Knight will then complete his move immediately. This will result in much faster play.

This game uses strings to display the playfield, and also to manipulate the playfield to show different areas for each player at the same time. All objects on the screen are part of an altered character set. Motion of individual characters, dragonfire, sorcerer's wisp and running motions are performed by changing the bytes in the character set making up that character during the motion routine.

One last note: when using a string as a screen display, make sure no part of the string crosses a 4K boundary or garbage may be displayed on the screen. If this problem is encountered, changing the order in which your strings are dimensioned may help. □

A$ - Machine language routine to move character set.

SCR$ - Full screen display.

SC1$, SC2$ - Holds obstacles for players.

SC1A$, SC1B$ - First and second screens for player one.

SC2A$, SC2B$ - First and second screens for player two.

TEMP$ - Holds reference characters for obstacles.

PL1$, PL2$ - Player titles and names.

SCREEN(N) - Which screen each player is on.

P1$, P2$ — Holds players' possessions.

KN - Which Knight is moving.

PLR - Number of players in game.

POS1, PPS2 — Players' positions.

POS - Player position during turn.

PR — Which players' turn.

MOVE - Which space to move to.

LT1, LT2 - Lost turns.

MR1, MR2 - Player sent to sorcerer's abode.

CHAL1, CHAL2 - Player has Chalice.

KEY1, KEY2 - Player has Key.

S - Joystick position.

P - Location of move in SCR$.

TPOS - Flashing Knight on or off.

S1, S2 - Offset of player playfield from SCR$.

SC - Position of obstacle in SC1$ or SC2$.

D - Position of dragonfire.

M1, M2 - Positions of sorcerer's abode.

ADDR - Address of new character set.

CHBAS - High byte address of new character set.

RAMTOP — New top of RAM.

PAGE - Number of pages Ramtop moved.

DLL, DLH - Display list Low byte, High byte.

SA, SL, SH - Address of SCR$, Low byte, High byte.

A, B, C, D, I, L, X, Y, Z - Common variables.

Line documentation.

0-20 — Go to start.

30-130 — Joystick and lost turn.

140-150 — Disable break key.

160-192 — Galloping knight.

200-350 — Flash knight, check for obstacles and make move.

360-800 — Check for invisible objects, win and bridge.

810-917 — Dragon, check for right sword, breath fire.

1000-1060 — Enemy archer, check for bow and arrow.

1070-1080 — Lake or lava pool, lose turn.

1090-1190 — Sorcerer.

1200-1210 — Find sword.

1220-1280 — Troll, check for necklace.

1290-1300 — Find necklace.

1310-1365 — Find chalice.

1500-1530 — Position knight to beginning of screen.

1540-1570 — Display positions.

1600-1630 — Position knight after non-movable obstacle.

2000-2380 — Construct new character set.

2400-2940 — Initialize, place obstacles.

3000-3250 — Initialize, display winner.

3400-3490 — If one player, score, coat of arms, high score.

```
0 REM >>KNIGHTS AND CHALICES<<
1 REM >>  by Bruce Willard  <<
3 REM
10 GOTO 3000
30 KN=C2:POS=POS1:IF PR=C1 THEN POS=PO
S2:KN=C1
40 MOVE=POS
50 S=STICK(PR):IF S=C15 THEN GOSUB 160
:GOTO 50
60 MOVE=POS+(S(C8)-C1*(S)C8 AND S(12)+
(S=C5 OR S=C9 OR S=13)*C20-C20*(S=C6 O
R S=C10 OR S=14)
65 GOSUB C200+80*PR
70 GOTO 50
80 IF  NOT PR AND LT2 THEN PR=C1:LT2=L
T2-C1:GOTO 100
90 IF PR AND LT1 THEN PR=C0:LT1=LT1-C1
100 IF LT1 AND LT2 THEN 80
110 IF PLR()C1 THEN PR= NOT PR
120 IF MR1 OR MR2 THEN GOSUB 2540
130 POKE 77,C0:GOTO 30
140 I=PEEK(16):IF I)128 THEN I=I-128:P
OKE 16,I:POKE 53774,I
150 RETURN
160 RESTORE 191+CHAL1*( NOT PR)+CHAL2*
PR:IF KN=C3 THEN RESTORE 192
170 B=ADDR+(KN)*C8+C7:FOR Y=C8 TO C4 S
TEP -C2:READ C:POKE B,C:SOUND C0,125,C
4*PR,Y
180 FOR Z=C1 TO C4:NEXT Z:SOUND C0,C0,
C0,C0:FOR Z=C1 TO C5*RND(C0)+C3:NEXT Z
:NEXT Y:RETURN
191 DATA 34,68,136
192 DATA 68,34,17
200 P=S1-C1+MOVE:IF P(S1 OR P)S1+C139
THEN 360
210 IF (MOVE-C1)/C20=INT((MOVE-C1)/C20
) AND POS/C20=INT(POS/C20) THEN 360
220 IF MOVE/C20=INT(MOVE/C20) AND (POS
-C1)/C20=INT((POS-C1)/C20) THEN 360
230 T=ASC(SCR$(P)):IF T AND T()194 THE
N 370
240 TPOS= NOT TPOS:SCR$(P,P)=CHR$(194*
TPOS):S=STICK(C0):IF S=C15 AND STRIG(C
0) THEN GOSUB 160:GOTO 240
250 IF  NOT STRIG(C0) THEN GOSUB 500:P
OP :GOTO 80
260 SCR$(P,P)=C0$:MOVE=POS:RETURN
280 P=S2-C1+MOVE:IF P(S2 OR P)S2+C139
THEN 360
290 IF (MOVE-C1)/C20=INT((MOVE-C1)/C20
) AND POS/C20=INT(POS/C20) THEN 360
300 IF MOVE/C20=INT(MOVE/C20) AND (POS
-C1)/C20=INT((POS-C1)/C20) THEN 360
310 T=ASC(SCR$(P)):IF T AND T()129 THE
N 370
320 TPOS= NOT TPOS:SCR$(P,P)=CHR$(129*
TPOS):S=STICK(C1):IF S=C15 AND STRIG(C
1) THEN GOSUB 160:GOTO 320
330 IF  NOT STRIG(C1) THEN GOSUB 700:P
OP :GOTO 80
340 SCR$(P,P)=C0$:MOVE=POS:RETURN
360 FOR Y=C8 TO C0 STEP -C1:SOUND C1,C
200,C8,Y:FOR Z=C0 TO C5:NEXT Z:NEXT Y:
MOVE=POS:RETURN
370 T=ASC(SCR$(P)):IF (T()22 OR POS()5
8) AND T()24 THEN 410
380 SCREEN(PR)= NOT SCREEN(PR):GOSUB 1
500+PR*C20+SCREEN(PR)*C10
385 FOR X=C1 TO C10:SOUND C0,C100,C8,C
8:FOR Y=C1 TO C5:NEXT Y
390 SOUND C0,C0,C0,C0:FOR Y=C1 TO C10:
NEXT Y:SOUND C0,C200,C4,C8:FOR Y=C1 TO
 C5:NEXT Y
395 SOUND C0,C0,C0,C0:FOR Y=C1 TO C10:
NEXT Y
400 NEXT X:POP :GOTO 80
410 T=ASC(SCR$(P)):IF KEY1 AND CHAL1 A
ND (T=205 OR T=206) THEN 3300
420 T=ASC(SCR$(P)):IF KEY2 AND CHAL2 A
ND (T=141 OR T=142) THEN 3300
430 GOTO 360
500 SC=MOVE+C140*SCREEN(PR):IF ASC(SC1
$(SC)) THEN 530
510 IF SCREEN(PR) THEN 520
515 SC1A$(POS1,POS1)=C0$:SC1A$(MOVE,MO
VE)=CHR$(194):POS1=MOVE:SCR$(S1,S1+C13
9)=SC1A$:RETURN
520 SC1B$(POS1,POS1)=C0$:SC1B$(MOVE,MO
VE)=CHR$(194):POS1=MOVE:SCR$(S1,S1+C13
9)=SC1B$:RETURN
530 IF ASC(SC1$(SC))()207 THEN 560
540 KEY1=C1:GOSUB 1540:FOR X=C90 TO C1
00:SCR$(P,P)=CHR$(207):SOUND C0,C100,1
4,C10
550 FOR Y=C1 TO C15:NEXT Y:SOUND C0,C2
00,C10,C10-X/C10:SCR$(P,P)=C0$:NEXT X:
GOTO 510
560 IF ASC(SC1$(SC))()208 THEN 580
570 GOSUB 1540:FOR X=C90 TO C100:SCR$(
P,P)=CHR$(208):SOUND C0,C150,14,C10:GO
TO 550
580 FOR X=C1 TO C20:IF ASC(SC1$(SC))=A
SC(TEMP$(X)) THEN 600
590 NEXT X:IF ASC(SC1$(SC))=C8 THEN 13
10
600 ON X GOTO 810,810,810,810,1000,100
0,1070,1070,1090,1090,1200,1200,1200,1
200,1070,1070,1070,1070,1220,1290
610 RETURN
700 SC=MOVE+C140*SCREEN(PR):IF ASC(SC2
$(SC)) THEN 730
710 IF SCREEN(PR) THEN 720
715 SC2A$(POS2,POS2)=C0$:SC2A$(MOVE,MO
VE)=CHR$(129):POS2=MOVE:SCR$(S2,S2+C13
9)=SC2A$:RETURN
720 SC2B$(POS2,POS2)=C0$:SC2B$(MOVE,MO
VE)=CHR$(129):POS2=MOVE:SCR$(S2,S2+C13
9)=SC2B$:RETURN
730 IF ASC(SC2$(SC))()143 THEN 760
740 KEY2=C1:GOSUB 1560:FOR X=C90 TO C1
00:SCR$(P,P)=CHR$(143):SOUND C0,215,14
,C10
750 FOR Y=C1 TO C15:NEXT Y:SOUND C0,C1
00,C10,C10-X/C10:SCR$(P,P)=C0$:NEXT X:
GOTO 710
760 IF ASC(SC2$(SC))()144 THEN 780
770 GOSUB 1560:FOR X=C90 TO C100:SCR$(
P,P)=CHR$(144):SOUND C0,120,14,C10:GOT
O 750
780 FOR X=C1 TO C20:IF ASC(SC2$(SC))=A
SC(TEMP$(X)) THEN 600
790 NEXT X:IF ASC(SC2$(SC))=C9 THEN 13
10
800 RETURN
810 D=P-C1:IF D/C20=INT(D/C20) THEN D=
P+C1
820 SCR$(P,P)=TEMP$(X):SCR$(D,D)=CHR$(
153):FOR Y=C0 TO C7:RESTORE 910+Y:FOR
Z=C0 TO C7
830 READ A:POKE ADDR+C8*25+Z,A:SOUND C
0,C200,C0,Y*2:NEXT Z:NEXT Y
835 FOR Y=C7 TO C0 STEP -C1:RESTORE 91
0+Y:FOR Z=C0 TO C7
840 READ A:POKE ADDR+C8*25+Z,A:SOUND C
0,C100,C0,Y+C2:NEXT Z:NEXT Y:FOR Z=C2
TO C30 STEP C2
850 IF  NOT PR AND ASC(P1$(Z))=ASC(TEM
P$(X+C10)) THEN 880
860 IF PR AND ASC(P2$(Z))=ASC(TEMP$(X+
C10)) THEN 880
870 NEXT Z:GOSUB 1600+PR*C20+SCREEN(PR
)*C10:LT1=LT1+X*( NOT PR):LT2=LT2+X*PR
:GOTO 510+PR*C200
880 FOR Y=-C150 TO C150 STEP C10:SOUND
 C0,ABS(Y),C6,C10-ABS(Y)/C15:POKE 5376
8,129:NEXT Y:POKE 53768,C0
890 IF PR THEN SC2$(SC,SC)=C0$:P2$(Z,Z
)=TEMP$(X):SCR$(450,479)=P2$:GOTO 710
```

As Easy As Falling Off . . .

HIGH QUALITY PRODUCTS AT AFFORDABLE PRICES

ALOG COMPUTING

ADVANCED SOFTWARE DESIGNS FOR HOME COMPUTERS

Dealer and distributor inquiries invited.

We at **Alog Computing** wish to thank you for the response you have given us on our first product the **Alog Pagewriter**. (See Comments) Now we have two other utility programs available for Atari Home Computer users. We have tried to make them simple and fun to use. We hope you enjoy them.

Note: Although these programs were developed for the Atari 400/800 models, we have tested them on the 1200 and have no reason to believe they will not run on all Atari machines.

THE ALOG PAGEWRITER

Turns your ATARI computer and 80 column printer into a very easy to use electronic typewriter. Because the entire page layout is displayed while the user is typing and editing, the **ALOG PAGEWRITER** is ideal for simple word processing tasks such as letters, notes, memos, or the kid's book reports and term papers. The average learning time is about five minutes.

KEY FEATURES

• Uses standard Atari editing keys (e.g. INSERT, DELETE, TAB, etc.) • A HELP screen with command summary • Visible and fully adjustable margins • FILL command for right justification • Automatic return option (with word moving) • Line 'Split' and 'Splice' commands • Store 10 pages with no disk swapping • Not copy protected

REQUIRED EQUIPMENT

• ATARI 400 or 800 Computer • ATARI 850 Interface Box • At least 32K of RAM • One disk drive • 80 column printer

THE ALOG DISPLAYMAKER

A versatile color graphics program in Forth that you can use for making graphs, charts, displays or just have fun drawing pictures.

KEY FEATURES

Graphics Mode: • Circles, Ellipses, Rectangles • "Rubber Band" lines • Semi-automatic fill • Color pallette for easy variation • Zoom magnification

Text Mode: • Three sizes of characters • Three colors • Regular ATARI character set • Special character set • "Built-in" character editor (gives unlimited number of characters)

Disk Storage of 8 displays

"Slide Show" display features

Graphics dump to most dot graphics printers (2 sizes) — Epson*, Gemini 10, Prowriter and others.

*requires Graphtrax option

REQUIRED EQUIPMENT

• ATARI 400 or 800 Computer • At least 48K of RAM • One disk drive • Optional (joy-stick and graphics printer)

THE ALOG MAILLIST

A simple, easy to use data base program specifically designed for making and keeping mailing lists and printing out mailing labels.

KEY FEATURES

• Multilevel fast search and sort • Complete disk utility package • Split & merge files • Supports 1 or 2 disk drives • Prints continuous form labels 1 to 2 inches high • Uses standard ATARI Editing Keys • Holds 130 records in memory and 500 records per disk • Make your own backup copies

REQUIRED EQUIPMENT

• ATARI 400/800 Computer • At least 32K RAM (48K recommended) • One disk drive (works fine with two) of any manufacture that works with ATARI DOS 2.0S • ATARI BASIC Cartridge • ATARI DOS 2.0S • An ATARI 850 Interface Module • A printer with adjustable tractor feed

UNSOLICITED COMMENTS

from users of the Alog Pagewriter (used with written permission)

"Great product. I wrote two pages the first time I sat down with it. I wish I had it a couple of years ago."
James Tanaka, Monterey Park, CA

"Refreshingly simple but adequate word processor at an affordable price — good work."
Jay Carccarese, San Jose, CA

"Super, I am a Valforth programmer and I love it."
P.R.S., San Antonio, Texas

"Good, simple text processor for letters — great to be able to back up master disk."
Larry Cox, Floyds Knobs, IN

"Excellent, it makes letter writing a dream."
M.H., Oaktown, VA

"Great price and easily learned."
Noel Brooks, Great Falls, MT

*"As I have used **PAGEWRITER** I have grown to love it. After using WORDSTAR and LETTER PERFECT a truly user kind word processor is a wonderful experience."*

"One feature which I feel sets it apart from all others is the capability to visualize the page. This feature is indispensable when designing tables for a report."

"Thanks again for an excellent piece of software."
John C. Goodman, Marblehead, PA

"Excellent for one or two pages of text. It's simple."
Richard E. Lane, Vandenberg AFB, CA

"Very Easy to use. It's very useful for letters"
Florian C. Pulver, Riverside, California

"I have found your product to be quite easy to operate and understand, both in the written instructions and manual operation. I have the ATARIWRITER and BANK STREET WRITER and I find myself using your product more than either one of these two products."

"I have also found that for the price of your product versus that of the other companies that your product has paid for itself twice over."
Edward Locke, Mentor, OH

"Very easy to use and perfect for my needs."
J.B. Karluk, Throop, PA

All programs are only $39.95 each. To order direct send check or money order to **Alog Computing, 1040 Veronica Springs Road,** Santa Barbara, CA 93105. We pay shipping. For information, credit card orders or C.O.D., call our distributor: **COMSTAR (805) 964-4660, outside Calif. (800) 558-8803.**

```
900 SC1$(SC,SC)=C0$:P1$(Z,Z)=TEMP$(X):
SCR$(C10,39)=P1$:GOTO 510
910 DATA 0,0,0,0,0,0,0,0
911 DATA 0,0,0,2,2,0,0,0
912 DATA 0,0,4,6,6,0,0,0
913 DATA 0,0,12,14,6,8,0,0
914 DATA 0,16,12,30,6,24,0,0
915 DATA 0,48,12,62,6,24,32,0
916 DATA 64,48,12,126,6,88,32,64
917 DATA 192,48,12,254,6,88,160,64
1000 SCR$(P,P)=TEMP$(X):KN=C3:FOR I=C1
 TO C6:GOSUB 160:NEXT I:KN=C1+( NOT PR
)
1010 FOR I=110 TO 80 STEP -C1:SOUND C0
,I,C4,(110-I)/C2:POKE 53768,65:NEXT I
1015 FOR I=C0 TO 14:SOUND C0,125,C6,14
-I:NEXT I
1020 FOR Z=C2 TO C30 STEP C2:IF (PR AN
D ASC(P2$(Z))=144) OR ( NOT PR AND ASC
(P1$(Z))=208) THEN 1050
1030 NEXT Z:LT1=LT1+(PR=C0):LT2=LT2+PR
:GOSUB 1600+C20*PR+C10*SCREEN(PR):MOVE
=SCREEN(PR)*43+( NOT SCREEN(PR))*81
1040 GOTO 510+C200*PR
1050 IF PR THEN P2$(Z,Z)=TEMP$(X):SCR$
(450,479)=P2$:SC2$(SC,SC)=C0$:GOTO 710
1060 P1$(Z,Z)=TEMP$(X):SCR$(C10,39)=P1
$:SC1$(SC,SC)=C0$:GOTO 510
1070 SCR$(P,P)=TEMP$(X):FOR Z=C50 TO C
150:SOUND C0,Z+C15,12,C15-Z/C10:SOUND
C1,Z,C10,C15-Z/C10:NEXT Z
1080 LT1=LT1+C3*( NOT PR):LT2=LT2+C3*P
R:GOSUB 1600+C20*PR+C10*SCREEN(PR):RET
URN
1090 SCR$(P,P)=CHR$(C10):FOR Z=C1 TO C
20:RESTORE 1190
1095 FOR I=C0 TO C2:READ A:POKE ADDR+C
8*C10+I,A:SOUND C0,C50,14,C6:NEXT I
1100 FOR I=C0 TO C2:READ A:POKE ADDR+8
0+I,A:SOUND C0,C10,14,C8:NEXT I:NEXT Z
1110 FOR Z=C50 TO C150:SOUND C0,Z,14,C
8:NEXT Z:FOR Z=C10 TO C0 STEP -C1:SOUN
D C0,C200,C0,Z:NEXT Z
1120 IF PR AND  NOT SCREEN(PR) THEN SC
2A$(POS2,POS2)=C0$
1130 IF PR AND SCREEN(PR) THEN SC2B$(P
OS2,POS2)=C0$
1140 IF  NOT PR AND  NOT SCREEN(PR) TH
EN SC1A$(POS1,POS1)=C0$
1150 IF  NOT PR AND SCREEN(PR) THEN SC
1B$(POS1,POS1)=C0$
1160 P=POS+(PR*52)+(( NOT PR)*51)-C1:S
CR$(P,P)=C0$:GOSUB 1600+C20*PR+C10*SCR
EEN(PR)
1170 P=SCREEN(PR)*M2+( NOT SCREEN(PR))
*M1:IF PR THEN POS2=P:P=52+P-C1:SCR$(P
,P)=CHR$(129):MR2=C1:RETURN
1180 POS1=P:P=51+P-C1:SCR$(P,P)=CHR$(1
94):MR1=C1:RETURN
1190 DATA 160,78,170,64,174,74
1200 FOR Z=C1 TO C20:SOUND C0,C100,C10
,Z/C2:SCR$(P,P)=TEMP$(X):FOR I=C1 TO C
10:NEXT I:SOUND C0,C50,C10,C10-Z/C2
1210 SCR$(P,P)=C0$:FOR I=C1 TO C10:NEX
T I:NEXT Z:GOSUB 1540+C20*PR:GOTO 510+
C200*PR
1220 FOR Z=C1 TO C6:SCR$(P,P)="]":FOR
I=C200 TO C150 STEP -C1:SOUND C0,I,C8,
C8:NEXT I:SCR$(P,P)="["
1230 FOR I=C100 TO C150:SOUND C0,I,C8,
C6:NEXT I:NEXT Z:FOR Z=C2 TO C30 STEP
C2
1240 IF (PR AND ASC(P2$(Z))=C30) OR (
NOT PR AND ASC(P1$(Z))=C30) THEN 1260
1250 NEXT Z:SCR$(P,P)=C0$:LT1=LT1+C2*(
 NOT PR):LT2=LT2+C2*PR:RETURN
1260 FOR I=C1 TO C4:FOR A=C50 TO C20 S
TEP -C1:SOUND C0,A,14,C8:NEXT A:NEXT I
:SOUND C0,C0,C0,C0
1270 IF PR THEN SC2$(SC,SC)=C0$:P2$(Z,
Z)=TEMP$(X):SCR$(450,479)=P2$:GOTO 710
1280 SC1$(SC,SC)=C0$:P1$(Z,Z)=TEMP$(X)
:SCR$(C10,39)=P1$:GOTO 510
1290 FOR Z=C1 TO C15:FOR I=C10 TO C0 S
TEP -C2:SOUND C0,I*C10,14,I:NEXT I:SCR
$(P,P)=CHR$(C30)
1295 FOR I=C10 TO C0 STEP -C1
1300 SOUND C0,I*C20,C10,I:SCR$(P,P)=C0
$:NEXT I:NEXT Z:GOSUB 1540+C20*PR:GOTO
 510+C200*PR
1310 SCR$(P,P)=CHR$(25):FOR X=-C5 TO C
5:Y=ABS(X):A=C0
1320 RESTORE 1360+Y:FOR Z=-120 TO 160
STEP C40:I=ABS(Z):SOUND C0,I,14,Y*C2:S
OUND C1,I+1,14,C2*Y
1330 READ D:POKE ADDR+C8*25+A,D:A=A+C1
:NEXT Z:NEXT X:SOUND C0,C0,C0,C0:SOUND
 C1,C0,C0,C0:IF PR THEN CHAL2=C1
1340 RESTORE 2380:FOR I=C0 TO C7:READ
A:POKE ADDR+C8*(C1+(PR=C0))+I,A:NEXT I
1345 GOSUB 1540+C20*PR:IF  NOT PR THEN
 CHAL1=C1
1350 GOTO 510+C200*PR
1360 DATA 129,0,0,0,0,0,0,129
1361 DATA 129,66,0,0,0,0,66,129
1362 DATA 0,66,38,0,0,38,66,0
1363 DATA 0,0,38,24,24,38,0,0
1364 DATA 0,0,0,24,24,0,0,0
1365 DATA 0,0,0,0,0,0,0,0
1500 POS1=58:SC1A$(POS1,POS1)=CHR$(194
):SC1B$(POS,POS)=C0$:SCR$(51,51+C139)=
SC1A$:RETURN
1510 POS1=43:SC1B$(POS1,POS1)=CHR$(194
):SC1A$(POS,POS)=C0$:SCR$(51,51+C139)=
SC1B$:RETURN
1520 POS2=58:SC2A$(POS2,POS2)=CHR$(129
):SC2B$(POS,POS)=C0$:SCR$(52,52+C139)=
SC2A$:RETURN
1530 POS2=43:SC2B$(POS2,POS2)=CHR$(129
):SC2A$(POS,POS)=C0$:SCR$(52,52+C139)=
SC2B$:RETURN
1540 FOR Z=C2 TO C30 STEP C2:IF ASC(P1
$(Z)) THEN NEXT Z:RETURN
1550 P1$(Z,Z)=SC1$(SC):SCR$(C10,39)=P1
$:SC1$(SC,SC)=C0$:RETURN
1560 FOR Z=C2 TO C30 STEP C2:IF ASC(P2
$(Z)) THEN NEXT Z:RETURN
1570 P2$(Z,Z)=SC2$(SC):SCR$(450,479)=P
2$:SC2$(SC,SC)=C0$:RETURN
1600 SC1A$(MOVE,MOVE)=TEMP$(X):MOVE=PO
S1:RETURN
1610 SC1B$(MOVE,MOVE)=TEMP$(X):MOVE=PO
S1:RETURN
1620 SC2A$(MOVE,MOVE)=TEMP$(X):MOVE=PO
S2:RETURN
52:RETURN
1630 SC2B$(MOVE,MOVE)=TEMP$(X):MOVE=PO
S2:RETURN
2000 GRAPHICS 17+(PLR=C1):GOSUB 140:PO
SITION C3,C5:? #C6;"DON THY ARMOR"
2010 SETCOLOR C0,C0,C10:SETCOLOR C1,13
,C6:SETCOLOR C2,C4,C4:SETCOLOR C3,C8,C
4:GOTO 2030
2020 DIM A$(C40):RAMTOP=PEEK(106)-C8:P
OKE 106,RAMTOP:CHBAS=RAMTOP:ADDR=CHBAS
*C256:PAGE=C4
2025 RESTORE 2055:FOR X=C1 TO C40:READ
 A:A$(X)=CHR$(A):NEXT X
2030 A=USR(ADR(A$),ADDR,PAGE)
2040 RESTORE 2100:FOR X=C1 TO C30:IF X
=17 THEN X=19
2050 FOR Y=C0 TO C7:READ A:POKE ADDR+X
*C8+Y,A:NEXT Y:NEXT X
2055 DATA 104,104,133,207,104,133,206,
104,104,133,212,169,0,133,204,169,224,
133,205,162
2057 DATA 1,160,0,177,204,145,206,200,
208,249,230,205,230,207,232,228,212,20
8,240,96
2060 POKE C756,CHBAS
2070 DLL=PEEK(560)+C256*PEEK(561)+C4:D
LH=DLL+C1
2080 IF  NOT PLR THEN RETURN
2090 REM ALT CHAR SET
2100 DATA 0,18,20,155,125,125,124,136
2110 DATA 0,18,20,155,125,125,124,136
2120 DATA 44,68,140,252,76,44,18,99
2130 DATA 0,60,126,235,175,219,118,60
2140 DATA 122,205,206,15,23,5,1,62
2150 DATA 56,126,251,95,238,126,24,24
2160 DATA 60,126,247,255,106,60,24,24
2170 DATA 0,255,60,126,60,24,24,60
2180 DATA 127,85,127,62,28,8,8,62
2190 DATA 64,174,74,0,126,14,62,127
```

```
2200 DATA 146,243,243,97,97,115,127,12
7
2210 DATA 73,207,207,134,134,206,254,2
54
2220 DATA 127,126,124,116,116,84,84,12
7
2230 DATA 254,126,62,46,46,42,42,254
2240 DATA 0,96,191,149,208,96,0,0
2250 DATA 32,16,146,127,146,16,16,32
2260 DATA 0,64,32,255,32,64,0,0
2270 DATA 0,24,102,177,48,32,0,0
2280 DATA 0,0,32,48,177,102,24,0
2290 DATA 0,3,7,13,25,255,5,1
2300 DATA 0,255,66,66,66,255,66,66
2310 DATA 0,192,224,176,152,255,160,12
8
2320 DATA 0,0,0,0,0,0,0,0
2330 DATA 60,126,255,255,255,255,126,6
0
2340 DATA 195,153,126,60,24,24,36,66
2350 DATA 146,40,68,130,68,40,146,0
2360 DATA 24,60,126,153,219,36,66,129
2370 DATA 129,66,36,24,24,36,36,24
2380 DATA 0,72,40,217,190,190,62,17
2400 POSITION C1,C8:? #C6;"PREPARE FOR
 BATTLE"
2410 SCR$=C0$:SCR$(480)=C0$:SCR$(C2)=S
CR$:SC1A$=SCR$:SC2A$=SCR$:SC1B$=SCR$:S
C2B$=SCR$:S1=61:S2=C261
2420 SCR$(221)="+.)'(43":SCR$(229)="!.
$":SCR$(233)="#(!,)#%3":SCR$(41)="TUTU
TUTUTUTUTUTUTU":SCR$(201)=SCR$(41)
2430 SCR$(241)="UTUTUTUTUTUTUTUTUTUT":
SCR$(401)=SCR$(241):SCR$="knight":SCR$
(8)="@":SCR$(441)="+.)'(4"
2435 SCR$(448)=CHR$(146)
2440 FOR X=C1 TO C10
2450 Y=INT(C140*RND(C0)+C1):Z=INT(C2*R
ND(C0)+70):IF ASC(SC1A$(Y)) THEN 2450
2460 SC1A$(Y,Y)=CHR$(Z)
2470 Y=INT(C140*RND(C0)+C1):Z=INT(C2*R
ND(C0)+70):IF ASC(SC1B$(Y)) THEN 2470
2480 SC1B$(Y,Y)=CHR$(Z)
2490 Y=INT(C140*RND(C0)+C1):Z=INT(C2*R
ND(C0)+70):IF ASC(SC2A$(Y)) THEN 2490
2500 SC2A$(Y,Y)=CHR$(Z)
2510 Y=INT(C140*RND(C0)+C1):Z=INT(C2*R
ND(C0)+70):IF ASC(SC2B$(Y)) THEN 2510
2520 SC2B$(Y,Y)=CHR$(Z):NEXT X:SC1A$(2
1)="KL":SC1A$(41)="MN":SC2A$(21)=CHR$(
139):SC2A$(22)=CHR$(140)
2525 SC2A$(41)=CHR$(141):SC2A$(42)=CHR
$(142):SC1A$(59)=CHR$(22):SC1A$(60)=CH
R$(23)
2530 SC1B$(41)=CHR$(23):SC1B$(42)=CHR$
(24):SC1B$(43)=C0$:SC2A$(59,60)=SC1A$(
59):SC2B$(41,43)=SC1B$(41)
2540 M1=C140:M2=C20:SC1A$(M1,M1)=CHR$(
28):SC2A$(M1,M1)=CHR$(28):SC1B$(M2,M2)
=CHR$(28):SC1B$(M1)=C0$
2545 SC2B$(M2,M2)=CHR$(28):IF MR1 OR M
R2 THEN RETURN
2550 SC1$(1,C140)=SC1A$:SC1$(C141,C280
)=SC1B$:SC2$(1,C140)=SC2A$:SC2$(C141,C
280)=SC2B$:POS1=81:POS2=81
2560 SC1A$(POS1,POS1)=CHR$(194):SC2A$(
POS2,POS2)=CHR$(194):SC1$(POS1,POS1)=C
0$:SC2$(POS2,POS2)=C0$
2570 SCR$(S1,S1+C139)=SC1A$:SCR$(S2,S2
+C139)=SC2A$
2580 SC1$(58,58)="]":SC2$(58,58)="]":S
C1A$(58,58)=C0$:SC2A$(58,58)=C0$
2590 RESTORE 2850:FOR X=C1 TO 18:READ
Y:TEMP$(X)=CHR$(Y):NEXT X:FOR X=C1 TO
18:IF X=C5 THEN A=C0
2600 Y=INT(C139*RND(C0)+C1):IF Y=POS1
OR ASC(SC1$(Y)) THEN 2600
2610 SC1$(Y,Y)=TEMP$(X)
2620 Y=INT(C139*RND(C0)+C141):IF Y=C26
1 OR ASC(SC1$(Y)) THEN 2620
2630 SC1$(Y,Y)=TEMP$(X):IF  NOT A THEN
 A=C1:GOTO 2600
2640 IF X=C5 THEN X=C6:A=C0
2650 Y=INT(C139*RND(C0)+C1):IF Y=POS2
OR ASC(SC2$(Y)) THEN 2650
2660 SC2$(Y,Y)=TEMP$(X)
2670 Y=INT(C139*RND(C0)+C141):IF Y=C26
1 OR ASC(SC2$(Y)) THEN 2670
2680 SC2$(Y,Y)=TEMP$(X):IF  NOT A THEN
 A=C1:GOTO 2650
2690 NEXT X
2700 Y=INT(C279*RND(C0)+C1):IF Y=POS1
OR Y=C261 OR ASC(SC1$(Y)) THEN 2700
2710 SC1$(Y,Y)=CHR$(207)
2720 Y=INT(C279*RND(C0)+C1):IF Y=POS2
OR Y=C261 OR ASC(SC2$(Y)) THEN 2720
2730 SC2$(Y,Y)=CHR$(143)
2740 FOR X=C1 TO C4
2750 Y=INT(C279*RND(C0)+C1):IF Y=POS1
OR Y=C261 OR ASC(SC1$(Y)) THEN 2750
2760 SC1$(Y,Y)=CHR$(208)
2770 Y=INT(C279*RND(C0)+C1):IF Y=POS2
OR Y=C261 OR ASC(SC2$(Y)) THEN 2770
2780 SC2$(Y,Y)=CHR$(144):NEXT X
2790 Y=INT(C139*RND(C0)+C1):IF Y=POS1
OR ASC(SC1$(Y)) THEN 2790
2800 SC1$(Y,Y)=CHR$(C30)
2810 Y=INT(C139*RND(C0)+C1):IF Y=POS2
OR ASC(SC2$(Y)) THEN 2810
2820 SC2$(Y,Y)=CHR$(C30)
2830 SA=ADR(SCR$):SH=INT(SA/C256):SL=S
A-SH*C256:POKE DLL,SL:POKE DLH,SH
2840 TEMP$(C20-C1,C20-C1)="]":TEMP$(C2
0,C20)=CHR$(C30)
2850 DATA 133,197,69,5,131,195,132,132
,10,10,147,211,83,19,218,218,218,218
2860 X=INT(C141*RND(C0)+C140):IF ASC(S
C1$(X)) THEN 2860
2870 SC1$(X,X)=CHR$(C8):Y=INT(C4*RND(C
0)+C1):X=X-C1:IF X)C140 THEN IF  NOT A
SC(SC1$(X)) THEN SC1$(X,X)=TEMP$(Y)
2880 X=X+21:IF X<C281 THEN IF  NOT ASC
(SC1$(X)) THEN SC1$(X,X)=TEMP$(Y)
2890 X=X-C40:IF X)159 THEN IF  NOT ASC
(SC1$(X)) THEN SC1$(X,X)=TEMP$(Y)
2900 X=INT(C141*RND(C0)+C141):IF ASC(S
C2$(X)) THEN 2900
2910 SC2$(X,X)=CHR$(C9):Y=INT(C4*RND(C
0)+C1):X=X-C1:IF X)C140 THEN IF  NOT A
SC(SC2$(X)) THEN SC2$(X,X)=TEMP$(Y)
2920 X=X+21:IF X<C281 THEN IF  NOT ASC
(SC2$(X)) THEN SC2$(X,X)=TEMP$(Y)
2930 X=X-C40:IF X)159 THEN IF  NOT ASC
(SC2$(X)) THEN SC2$(X,X)=TEMP$(Y)
2940 GOTO 30
3000 CLR :C0=0:C1=1:C2=2:C3=3:C4=4:C5=
5:C6=6:C7=7:C8=8:C9=9:C10=10:C20=20:C3
0=30:C40=40:C140=140
3002 C200=200:C255=255:C256=256:C141=1
41:C280=280:C261=261
3003 C15=15:C50=50:C150=150:C756=756:C
139=139:C279=279:C281=281:C656=656:C65
7=657:C764=764:C90=98:C100=100
3005 GRAPHICS C2:GOSUB 140
3010 POKE 710,C0:POKE 708,148:POKE 752
,C1:POKE C656,C0:POKE C657,14:? "INITI
ALIZING":GOSUB 2020:PR=C0:KN=C1
3020 DIM F1$(1):F1A=ADR(F1$):F2L=(1-((
F1A/4096)-INT(F1A/4096)))*4096:IF F2L<
1800 THEN DIM F2$(F2L+2)
3022 DIM SCR$(480),SC2$(C280),SC1$(C28
0),SC1A$(C140),SC2A$(C140),SC1B$(C140)
,SC2B$(C140),TEMP$(C20),C0$(C1)
3025 SCR$=" knights ":SCR$(C10)=CHR$(C
1):C0$=CHR$(C0)
3030 DIM P1$(C30),P2$(C30),SCREEN(C1):
SCREEN(C0)=C0:SCREEN(C1)=C0:P1$=C0$:P1
$(C30)=C0$:P1$(C2)=P1$:P2$=P1$:? "K"
3040 DIM PL1$(C20),PL2$(C20):FOR X=C10
 TO C1 STEP -C1:POSITION C0,C3:? #C6;S
CR$(X,C10):GOSUB 160:NEXT X
3050 FOR X=C9 TO 18:POSITION X,C3:? #C
6;SCR$(C9,C10):GOSUB 160:NEXT X:POSITI
ON 19,C3:? #C6;" "
3060 SCR$="        and ":SCR$(12)=CHR$(
C1)
3070 FOR X=12 TO C1 STEP -C1:POSITION
C0,C5:? #C6;SCR$(X,12):GOSUB 160:NEXT
X
3080 FOR X=11 TO 18:POSITION X,C5:? #C
6;SCR$(11,12):GOSUB 160:NEXT X:POSITIO
N 19,C5:? #C6;" "
3090 SCR$="          CHALICES ":SCR$(19
)=CHR$(C1)
```

COMPUTER PALACE

MasterCard VISA AMERICAN EXPRESS

Over 1000 Items For Your Atari!

ATARI®

We Know ATARI Computers... Call Us For The Latest Info: (503)683-5361

the Koala Pad

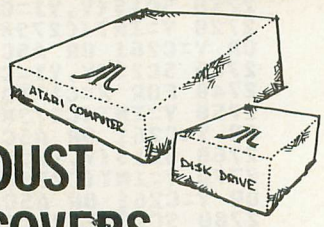
From Koala Tech.

The future is here... Input your commands with the touch of a finger on your Koala Pad.

- **Super Graphics & Drawing Program Included with Pad.**

Retail $100.00

$87.00 + FRT



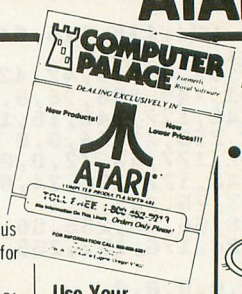
DUST COVERS

Made from durable brown vinyl & custom fitted for:

- ATARI 400, 800, 600XL, 1200XL Computers
- ATARI 410 Recorder, 810 Disk Drive
- EPSON MX-80 or FX-80 Printers

$8.95 EACH Please specify which model

FREE Catalog



With any order, or send $1 (refundable with purchase). Send us your name & address for FREE QUARTERLY FLYERS with SPECIALS!

Over 1000 items for your Atari...

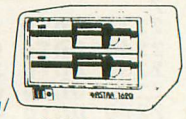
- Mosaic • Percom
- Broderbund • APX
- Roklan • Datasoft
- Synapse • A.I
- On-Line • Atari
- Hayes • Epson
- Visicalc • More!

Use Your Credit Cards!

We honor VISA, American Express, & MasterCard.

24 & 48 Hour Shipping Is Available... CALL!

Astra 1620



- Double Density/ Double Drives!

$488 + FRT

Ape-Face

Printer Interface

$89.50

Modem & Printer Interface

Auto Dial
Auto Answer

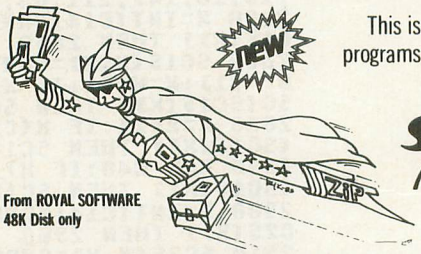
$149 + FRT

From Microperipheral Corp

PINBALL (Much more than a game!) CONSTRUCTION SET

- **A Fantastic Program!**
- **Design the Pinball Game of your dreams... or play one of five games included on the disk!**

New from Electronic Arts **48K Disk** **$35.10**



From ROYAL SOFTWARE
48K Disk only

This is one of the **Most Versatile** data-base programs available for business or home use!

Super Mailer PLUS

Much More Than A Mailing List!

Maintain Your Book Library
Organize Your Record Collection
Keep Track Of Your Recipies
Are You A Stamp Collector?
Let Super Mailer Plus keep track of all these and more!

Dealer Inquiries Invited

Our Price

$49.95

You can accomplish this **and more** because of the **"User Re-defineable fields"!** The program has many other features: Multiple Search—Special Code Field—Single/or/Double Density Operation—Create Sub-files—Auto-delete—Print Labels 1/2/or 3-up—Print File Copy—Merge/Search/Sort—Built-in Official Post Office State Abbreviations—More...

$54.00

The Monkey Wrench II

A cartridge for the right-hand Slot of your Atari 800!

This is a BASIC & Machine-language **Programmers Aid** for the 800. It plugs into the right slot, and works with Atari BASIC to add 16 new direct-mode commands (Incl: Auto line numbering, Delete lines, Renumber BASIC, Hex-Dec conv., Find string, Move lines, Exchange string, Disk directory from BASIC, Mem. test, More...). Also includes a monitor with 15 commands to interact with the 6502. This has been proven to be a very helpful tool when writing your own programs!

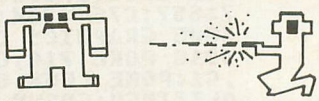
ALL-TIME FAVORITES

Star Raiders(C)	38.20
Eastern Front (D,T)	29.95
Pacman (C)	39.50
Temple of Apshi (D,T)	35.10
Zork I, II, or III (D)	each 35.10
Centipede (C)	39.50
Missle Command (C)	29.70
Shamus (D,T)	31.50
Ghost Hunter (D)	34.95
Ghost Hunter (T)	29.95

TOP-TEN PROGRAMS

Pole Position (C)	44.90
Donkey Kong (C)	44.90
Blue Max (D,T)	31.50
Baseball (D,T)	28.90
Pinball Construction Set (48K, Disk)	35.10
Football (D,T)	28.90
Monkey Wrench II (not a game) (C)	54.00
Ultima II (D)	52.50
Archon (D)	35.10
Dig Dug (C)	44.90

SUPER SPECIALS... Up To 70% Off!



K-Razy Shootout Cartridge **Special $17.95**

From Royal Software

Special $9.95 Reg. $29.95

Meteor Storm 24K Disk Only

Save the cities & the people!
100% Machine Language
Fast action game

	Regular		SPECIAL
Action Quest(D,T)	29.95		17.95
Alien Numbers(D,T)	29.95	Super!	9.95
Alien Swarm(D,T)	29.95	Special!	9.95
Baja Buggies(D,T)	31.95	Wow!	17.95
Bug-Off(D,T)	29.95	Look!	9.95
Frogger(D,T)	34.95		23.95
Genetic Drift(D,T)	29.95		17.95
Ghost Encounters(D,T)	29.95		17.95
Intruder[Inhome](D,T)	29.95	Great!	9.95
Labyrinth(D,T)	29.95		17.95
Magic Mail(D)	59.95	Deal!	29.90
Maurader(D)	34.95		19.90
Megalegs[CTIA Only](T)	34.95	Look!	9.95
Meteor Storm(D)	29.95	Gee!	9.95
River Rescue(C)	39.95		29.90
Slime[Synapse](D,T)	34.95		19.95
Stratos(D,T)	34.95		19.95
Survival of the Fittest(C)	39.95	Look!	17.95
Survivor(D,T)	34.95		19.90
Tutti-Frutti(D,T)	29.95	Wow!	9.95
Votrax Type-N-Talk	350.00	A Steal!	150.00
Wall War(D)	29.95	Super!	17.95

LIMITED QUANTITIES—WHILE SUPPLIES LAST!

HARDWARE

Astra Disk Drive	488.00
Micromainframe Disk Drive	499.00
Atari 1020 Printer	199.00
Atari 1010 Recorder	86.00
Super-5 Printer	329.00
Panasonic Monitor (Grn Scrn, w/sound)	199.00
Dust Covers (400/800/410/810)	each 8.95
Trak A2-D2 Disk Drive	398.00

SOFTWARE

Archon (D,T)	35.10
Pinball Construction (D)	35.10
Microsoft Basic II (C)	79.10
Diskey Utility (D)	44.90
Quasimodo (D,T)	31.50
Murder on the Zinderneuf (D)	35.10
Blue Max (D,T)	31.50
Odessa Chess (D)	59.50
Frogger (T)	23.95
Pharoah's Pyramid (D)	31.50
Rally Speedway (C)	44.90
Ultima II (D)	52.50

COMPUTER PALACE

MasterCard VISA AMERICAN EXPRESS

OPEN Monday - Saturday, 9 to 6
2160 W. 11th Avenue Eugene, Oregon 97402

USE YOUR CREDIT CARD & CALL

Toll Free 1-800-452-8013

(Orders Only, Please)

For Information Call (503) 683-5361

Shipping Note: UPS & Parcel Post-$2.00—UPS Air-$3.90
Shipping on hardware & paper supplis extra—Call for $.
24 & 48 hour shipping available... Call (503) 683-5361

We specialize in *ATARI Compatible Products*
So we know what works best!!!
Call for the latest info... (503) 683-5361

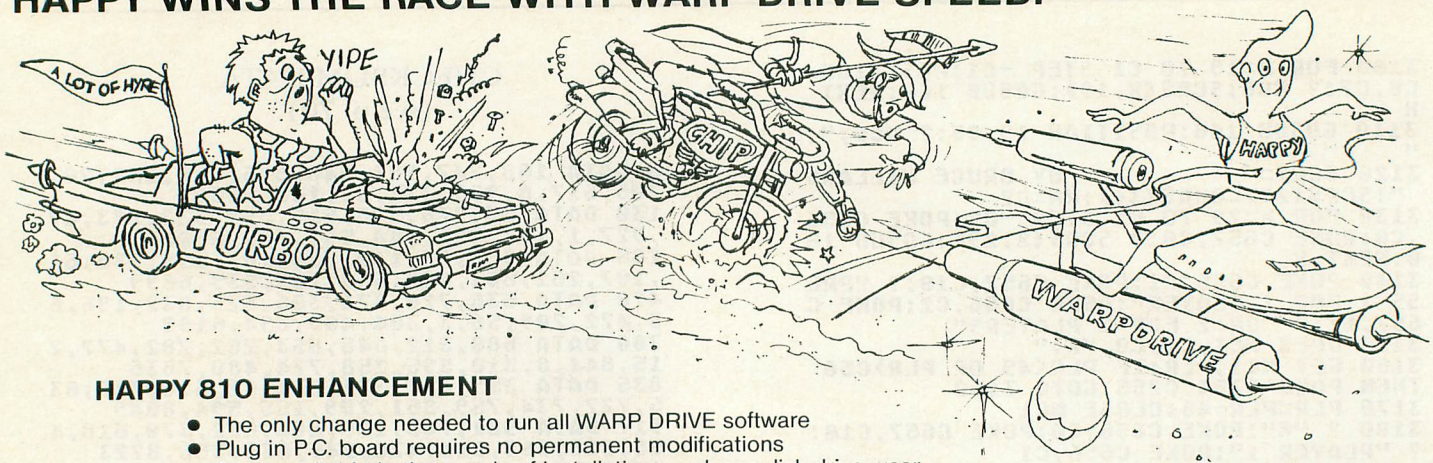
```
3100 FOR X=19 TO C1 STEP -C1:POSITION
C0,C7:? #C6;SCR$(X,19):GOSUB 160:NEXT
X
3110 GOSUB 160:POSITION 19,C5:? #C6;"
"
3120 SCR$="            BY BRUCE WILLARD
 ":SCR$(29)=CHR$(34):KN=C2
3130 FOR X=29 TO C1 STEP -C2:POKE C656
,C0:POKE C657,C0:? SCR$(X,29):GOSUB 16
0:NEXT X
3140 POKE C656,C1:POKE C657,C10:? "PRE
SS 1 FOR 1 PLAYER":POKE C656,C2:POKE C
657,11:? "OR 2 FOR 2 PLAYERS"
3150 OPEN #C1,C4,C0,"K:"
3160 GET #C1,PLR:IF PLR<49 OR PLR>C50
THEN POKE C764,C255:GOTO 3160
3170 PLR=PLR-48:CLOSE #C1
3180 ? "ß":POKE C656,C0:POKE C657,C10:
? "PLAYER 1":POKE C656,C1
3185 POKE C657,C10:? "LORD OR LADY":PO
KE C656,C2
3190 POKE C657,C10:INPUT PL1$:? "ß":PO
KE C656,C0
3195 POKE C657,C10:? "AND YOUR NAME":P
OKE C656,C1:POKE C657,C10
3200 INPUT TEMP$:PL1$(C6)=TEMP$:FOR I=
C1 TO LEN(PL1$):PL1$(I,I)=CHR$(ASC(PL1
$(I))+160):NEXT I:PL1$(C5,C5)=" "
3210 IF PLR=C1 THEN 3250
3220 ? "ß":POKE C656,C0:POKE C657,C10:
? "PLAYER 2":POKE C656,C1
3225 POKE C657,C10:? "LORD OR LADY":PO
KE C656,C2
3230 POKE C657,C10:INPUT PL2$:? "ß":PO
KE C656,C0
3235 POKE C657,C10:? "AND YOUR NAME":P
OKE C656,C1:POKE C657,C10
3240 INPUT TEMP$:PL2$(C6)=TEMP$:FOR I=
1 TO LEN(PL2$):PL2$(I,I)=CHR$(ASC(PL2$
(I))+128):NEXT I
3250 PL2$(C5,C5)=CHR$(128):GOTO 2000
3300 POKE 106,RAMTOP+C8:GRAPHICS 17:GO
SUB 140
3310 SETCOLOR C0,C0,C10:SETCOLOR C1,13
,C6:SETCOLOR C2,C4,C4:SETCOLOR C3,C8,C
4:POKE C756,224:POSITION C2,C5
3320 ? #C6;"CONGRATULATIONS":IF PR THE
N PL1$=PL2$
3330 A=19-LEN(PL1$):A=INT(A/C2):POSITI
ON A,C7:? #C6;PL1$:IF PLR=C1 THEN GOSU
B 3400:GOTO 3360
3340 POSITION C4,C10:? #C6;"you bested
":POSITION C4,11:? #C6;"your opponent"
:POSITION C4,14:? #C6;"the realm"
3350 POSITION C4,C15:? #C6;"salutes yo
u"
3360 POSITION C1,C20:? #C6;"ANOTHER CH
ALLENGE?":POSITION C7,22:? #C6;"(Y/N)"
:POKE C764,C255
3370 A=PEEK(C764):IF A=C255 THEN 3370
3380 POKE C764,C255:IF A=35 THEN END
3390 CLR :RUN
3400 A=C100-LT1:B=PEEK(1536):POSITION
C4,C15:? #C6;"HIGH SCORE:"
3410 IF A>B THEN POKE 1536,A:FOR X=C1
TO LEN(PL1$):POKE 1537+X,ASC(PL1$(X)):
NEXT X:POKE 1537,LEN(PL1$)
3420 POSITION C2,C10:? #C6;"thy score
was ";A:POSITION C2,12:? #C6;"thy coat
 of arms":POSITION C2,13
3430 ? #C6;"is a ";:IF LT1<C20 THEN ?
#C6;"lion":GOTO 3480
3440 IF LT1<C30 THEN ? #C6;"kangaroo":
GOTO 3480
3450 IF LT1<C40 THEN ? #C6;"goat":GOTO
 3480
3460 IF LT1<C50 THEN ? #C6;"duck":GOTO
 3480
3470 ? #C6;"snail"
3480 B=PEEK(1536):C=PEEK(1537):FOR X=C
1 TO C:PL1$(X,X)=CHR$(PEEK(1537+X)):NE
XT X
3490 L=17-C:L=INT(L/C2):POSITION L,17:
? #C6;PL1$(C1,C);" ";B:RETURN
```

CHECKSUM DATA

(see p. 70)

```
0 DATA 168,747,991,640,576,720,142,20,
705,617,8,969,327,82,148,6860
130 DATA 55,270,596,516,599,932,283,87
,927,1,153,591,188,593,789,6580
280 DATA 958,25,149,590,187,596,785,61
,107,262,884,848,669,239,539,6899
420 DATA 536,720,671,306,825,832,196,6
3,472,209,58,3,304,409,594,6198
700 DATA 680,312,846,853,202,782,477,2
15,844,8,310,595,258,774,480,7636
835 DATA 359,447,981,531,13,616,338,63
5,727,734,759,861,299,155,594,8049
917 DATA 981,783,185,666,672,370,618,4
55,806,563,214,478,246,630,556,8223
1120 DATA 52,420,797,79,16,232,129,386
,609,323,272,744,871,748,869,6547
1270 DATA 232,866,262,654,108,610,564,
206,782,839,628,98,583,160,95,6687
1364 DATA 990,939,262,258,279,275,84,7
07,89,877,674,676,681,683,714,8188
2010 DATA 110,537,83,681,526,687,938,6
87,604,743,754,930,196,197,817,8490
2130 DATA 191,863,212,192,865,689,719,
287,517,279,5,696,938,609,402,7464
2280 DATA 409,104,934,264,927,279,959,
740,222,880,757,103,120,371,312,7381
2435 DATA 882,428,835,773,844,776,854,
773,828,195,448,226,321,372,744,9299
2560 DATA 768,89,975,269,569,870,532,5
36,525,592,878,550,554,558,96,8361
2710 DATA 945,104,945,609,106,953,114,
163,610,949,586,954,373,234,632,8277
2860 DATA 773,746,761,778,771,751,762,
779,650,166,621,193,282,80,622,8735
3022 DATA 292,502,158,959,727,520,435,
508,381,470,799,573,105,657,784,7870
3160 DATA 642,475,456,46,956,812,140,2
7,455,43,955,809,180,284,966,7246
3310 DATA 624,472,403,788,72,478,782,7
43,855,102,599,107,621,267,789,7702
3460 DATA 767,658,831,310,2566
```

HAPPY WINS THE RACE WITH WARP DRIVE SPEED!



HAPPY 810 ENHANCEMENT

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

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

NEW HAPPY WARP DRIVE SOFTWARE

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

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

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

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

HAPPY WARP DRIVE SECTOR COPY PROGRAM
- Standard format whole disk read, write and verify in 105 seconds
- Use with sngle or dual drives, mix ENHANCED and NON-ENHANCED drives

HAPPY CUSTOMIZER PROGRAM (sold separately $99.95)
- Creates custom format disks of any specification
- Any type bad sector, duplicate sector numbers, or interleave
- Easy to use but requires an advanced level user to interpret the results

REVIEWED IN POPULAR MAGAZINES

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

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

SPECIAL SUGGESTED RETAIL PRICE BEFORE FEBRUARY 28, 1984: *Get the HAPPY 810 ENHANCEMENT with the single and multi drive HAPPY BACKUP PROGRAM, plus the HAPPY COMPACTOR PROGRAM, plus the HAPPY DRIVE DOS, plus the HAPPY SECTOR COPY, all with WARP DRIVE speed, including our diagnostic for $249.95. Existing registered ENHANCEMENT owners may upgrade to WARP DRIVE speed for $15.00 with no hardware changes.!*

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

HAPPY COMPUTERS, INC. ● P. O. Box 1268 ● Morgan Hill, California 95037 ● (408) 779-3830

MUSIC SYNTHESIZER

16K Cassette 24K Disk
Requires Music Composer cartridge

by Ken Collier

In the two years I've owned my 400 computer, I have enjoyed experimenting with Atari's **Music Composer** cartridge. Unfortunately, it only makes an organ-like tone that can't be changed. Think how much nicer it would be if you could control the actual sound of the notes as well as their pitch!

I wrote **Music Synthesizer** for this very purpose. Used in conjunction with the **Music Composer**, I can orchestrate my own four-piece band that doesn't take breaks. Sound changes are just a keystroke away. Anything from piano to organ to bongos to science fiction "weird" can be created, from a melodic ditty to a noisy dirge.

Atari's **Music Composer** cartridge must be used to compose the music. After a music file has been saved on disk or cassette, it can be entered into the **Synthesizer**. Files containing up to approximately 5000 notes are accepted. All ten phrases are independently controllable, so each phrase can sound different if desired. All voices and all "arrange music" commands are supported during playback. Also available are adjustable tempo and the ability to repeat a song up to 255 times automatically.

Music Synthesizer works by altering four sound parameters: attack, decay, volume drop and vibrato. "Attack" is the time required for a note to rise from no volume to maximum volume. "Decay" is the time required for the note to drop to a steady (sustain) volume. In **Music Synthesizer**, a note remains at sustain volume until its duration is over. Then it drops instantly to zero volume. Sustain and release (zero-drop) times cannot be varied; the difference in sound is small.

"Volume drop" determines how far a note drops from maximum volume before sustaining, from no drop to a complete drop. "Vibrato" is a repeated shift to a new pitch, followed by a return to the original pitch. Both the vibrato range and speed are variable. The range determines how far from the original pitch the note travels before returning. The same range will create an apparently larger sound difference for high-pitched notes than it will for lower notes. The reason can be deduced from the chart of notes on page 58 of the *Atari BASIC Reference Manual*: the values for the lower pitches are farther apart. If the range you specify is large enough, the pitch will "wrap around" the scale. The result can sound rather like a crazed marimba player. Vibrato speed determines how fast this pitch exchange takes place.

Typing the program.

The Atari BASIC program in **Listing 1** is the main data and data checking routine. This listing is used to create both disk and cassette versions of **Music Synthesizer**. The DATA statements are listed in hexadecimal (base 16) to make the program fit in a 16K cassette-based system. It makes typing the program a little more difficult, but it's a necessary evil.

Listing 2 must be added to **Listing 1** if you're using a 410 or 1010 cassette recorder.

Listing 3 must be added to **Listing 1** if you're using a disk drive.

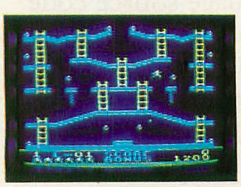
Listing 4 is the assembly-language source code for **Music Synthesizer**, written with the Atari Assembler/Editor cartridge. You do *not* have to type in **Listing 4** to use the **Synthesizer**! It's provided for those readers who are interested in seeing how the program works.

Cassette instructions.

1. **Type Listing 1** into your computer and verify your typing with **C:CHECK** (see page 70).

2. With **Listing 1** correctly entered, type in **Listing 2**. The program lines will automatically

JUMPMAN'S A GREAT GAME. BUT YOU'VE GOT TO WATCH YOUR STEP.



Meet the Alienators. A fiendish bunch who've planted bombs throughout your Jupiter Command Headquarters.

Your job? Use your lightning speed to scale ladders, scurry across girders, climb ropes and race through 30 levels to defuse the bombs before they go off.

That's the kind of hot, non-stop action we've packed into the award-winning,* best-selling Jumpman,™ and into Jumpman Jr.,™ our new cartridge version with 12 all-new, different and exciting screens.

Both games force you to make tough choices. Should you avoid that Alienator, climb to the top and try to work your way down, or try to hurdle him and defuse the bombs closest to you before they go off?

If you move fast you'll earn extra lives. But if you're not careful, it's a long way down.

So jump to it. And find out why Jumpman and Jumpman Jr. are on a level all their own.

One to four players; 8 speeds; joystick control. Jumpman has 30 screens. Jumpman Jr. has 12 screens.

*1983 C.E.S. award winner.

EPYX COMPUTER SOFTWARE

STRATEGY GAMES FOR THE ACTION-GAME PLAYER.

merge with **Listing 1**. Make sure these new lines were typed correctly! It's a good idea to CSAVE the entire program at this point.

3. Type RUN and press RETURN. The program will begin checking the DATA lines, printing each line number in turn. You will be alerted if there are any problems. Fix incorrect lines and re-RUN the program as necessary until all errors are eliminated.

4. When all DATA is correct, you will be prompted to "Ready cassette and press RETURN." Put a blank cassette into your recorder, press the RECORD and PLAY keys simultaneously and hit RETURN. The message "Writing file" will appear and the program will create a boot-tape version of **Music Synthesizer**, displaying each line number as it goes. When the READY prompt reappears, the **Synthesizer** is ready to use. Be sure you have CSAVEd the BASIC program on a separate tape.

5. Whenever you want to use **Music Synthesizer**, do the following: Rewind the tape created by the BASIC program to the beginning. Turn your computer OFF and remove all cartridges. Press the PLAY key on your recorder and turn your computer back ON while holding down the START key. The computer will "beep" once. Press RETURN and the **Synthesizer** will load and run automatically.

Disk instructions.

1. Type **Listing 1** into your computer and verify your typing with **D:CHECK2** (see page 70).

2. With **Listing 1** correctly entered, type in **Listing 3**. The program lines will automatically merge with **Listing 1**. Make sure these new lines were typed correctly! It's a good idea to SAVE the entire program at this point.

3. Type RUN and press RETURN. The program will begin checking the DATA lines, printing each line number in turn. You will be alerted if there are any problems. Fix incorrect lines and re-RUN the program as necessary until all errors are eliminated.

4. When all DATA is correct, you will be prompted to "Insert disk with DOS, press RETURN." Put a disk containing DOS 2.0S in drive #1 and hit RETURN. The message "Writing file" will appear and the program will create an AUTORUN.SYS version of **Music Synthesizer**, displaying each line number as it goes. When the READY prompt reappears, the **Synthesizer** is ready to use. Be sure you have SAVEd the BASIC program.

5. Whenever you want to use **Music Synthesizer**, do the following: Turn your computer OFF and remove all cartridges. Insert the disk containing the AUTORUN.SYS file into drive #1 and turn the computer back ON. **Music Synthesizer** will load and run automatically.

Using the Synthesizer.

If everything worked, **Music Synthesizer**'s main menu will be displayed. Type the letter of your choice. The options are:

Listen. Listen to a song. Push SELECT to stop a song before it ends. If no **Music Composer** files have been entered, the **Synthesizer** will print "No Data."

Retrieve. Enter a **Music Composer** file from disk or cassette. When prompted for a device name, type "C" for cassette or "D(number):filename" for disk.

Number. Number of times to play a song. The program counts down to zero; the default value is one. Pushing SELECT during playback resets **Number** to one.

Phrase Number. Which phrase to change (0-9). Hitting RETURN here will bring you back to the top of the main menu. If the selected phrase wasn't used in your file, the program will display a "No Data" message. Otherwise, it will prompt you for sound parameters:

Attack Rate. 1 is the fastest, 255 the slowest.

Decay Rate. Same as **Attack**.

Volume Drop. 0 is no drop, 7 is a complete drop (like a drum).

Vibrato Range. 0 is no vibrato, 1 is the smallest range, 255 the largest.

Vibrato Speed. 1 is the quickest, 255 the slowest.

Tempo. Speed to play the song. 1 is the quickest (useful for hyperkinetics), 255 is the slowest (for water torture).

Hitting RETURN over any parameter value will leave that value the same. Hitting SYSTEM RESET restores all default values.

The rest is up to you. Enjoy all the variations possible, but have some sympathy for your friends and neighbors, too. "99 Bottles of Beer on the Wall" might drive them crazy, no matter how nice it sounds. □

Listing 1.

```
1 REM *** MUSIC SYNTHESIZER ***
10 DATA 0,1,2,3,4,5,6,7,8,9,0,0,0,0,0,
0,0,10,11,12,13,14,15
20 DIM DAT$(91),HEX(22):FOR X=0 TO 22:
READ N:HEX(X)=N:NEXT X:LINE=990:RESTOR
E 1000:TRAP 60:? "CHECKING DATA"
25 LINE=LINE+10:? "LINE:";LINE:READ DA
T$:IF LEN(DAT$)<>90 THEN 110
28 DATLIN=PEEK(183)+PEEK(184)*256:IF D
ATLIN<>LINE THEN ? "LINE ";LINE;" MISS
ING!":END
30 FOR X=1 TO 89 STEP 2:D1=ASC(DAT$(X,
X))-48:D2=ASC(DAT$(X+1,X+1))-48:BYTE=H
EX(D1)*16+HEX(D2)
35 IF PASS=2 THEN PUT #1,BYTE:NEXT X:R
EAD CHKSUM:GOTO 25
```

```
40 TOTAL=TOTAL+BYTE:IF TOTAL>999 THEN
TOTAL=TOTAL-1000
45 NEXT X:READ CHKSUM:IF TOTAL=CHKSUM
THEN 25
50 GOTO 110
60 IF PEEK(195)<>6 THEN 110
100 ? "WRITING FILE":PAS5=2:LINE=990:R
ESTORE 1000:TRAP 60:GOTO 25
110 ? "BAD DATA: LINE ";LINE:END
1000 DATA D820E22020A32120A3264C3C25A2
30A97F9D4403A9289D450320B2263028A9079D
4203A9009D4403A92B9D4503,888
1010 DATA A9009D4803A92A9D49032056E4C0
88D008A90C9D42034C56E484D4208C26A2BAA0
27209624A230203E2068684C,359
1020 DATA C521A90085A0A02B84A1A8B1A0C9
AAD035C8D002E6A1B1A0300AAA9480A5A19581
4C8D20A90385A0B1A00A0A0A,11
1030 DATA 85C1A90085A0C8D002E6A1B1A0C9
FFF008C8D0F2E6A14C8D20C8D002E6A1B1A0C9
FFD0BAA206BDDA2085A6A92A,36
1040 DATA 85A7A9FFA00191A688B59485A0B5
9585A1F00EB1A091A6C8B1A091A6C8C9FFD0F2
CACA10D460002A402A802AC0,708
1050 DATA 2AA21BA9009580CA10FB60A203A9
0685A8B5B9F031C902D0034C3123202F21C907
B01FA8B9152185A6B91C2185,432
1060 DATA A7202F216CA600F245549D8EF297
202121212120210A90095B9CA8A0A85A810C460
FE1C06BD412185A0A92A85A1,136
1070 DATA BC1C06B1A060004080C038E9010A
DE2006F0A49D1C064CF220A8B934069D0806B9
3E069D0C06B948069D1006B9,317
1080 DATA 52069D1406B95C069D1806980AA8
A90195B99D2C06B9800095B0B981009SB4F002
F6B94CF2200A09A09D28064C,861
1090 DATA F2209D20064CF2209D24064CF220
A9FD8D3002A9268D3102A9628DC80260204925
20E22020D325200D20205C20,892
1100 DATA 4C242220C626482048246BC94CF0
16C952F0DEC94EF031C950F06BC954F049208C
264CC521A595059705990059B,4
1110 DATA D0034C5324A265A02720712620CD
2420EC2020B1242038254C0A25A28FA0272071
262066251006209D264C0A22,348
1120 DATA D006AAF0F5CA86B2048244CC521
A5C1209024A212A027202B231006209D264C2A
22D0E4AAF0F585C14C2422A2,368
1130 DATA 1FA027202B23D0D3C90A9006209D
264C4822AAF0F785A80AA8B98100D0034C5324
204824A6A8BD340620902402,848
1140 DATA 2CA027202B231006208C264C6A22
D008AAF0F5A6A89D3406204824A6A8BD3E0620
9024A23AA027202B23100620,595
1150 DATA 8C264C8E22D008AAF0F5A6A89D3E
0620482AA6A8A9AE38FD48064A209024A280A0
27202B233006D017C9089006,188
1160 DATA 208C264CB2220A85AFA9AE38E5AF
A6A89D4806204824A6A8BD5206209024A247A0
27202B231006208C264CE322,450
1170 DATA D005A6A89D5206204824A6A8BD5C
06209024A256A027202B231006208C264C0423
D008AAF0F5A6A89D5C064C24,691
1180 DATA 222071264C6625A4A8BD2C06D012
DE3006F0034C2721B59CD0069900D29901D2F6
B0D002F6B4B59CD011A90195,804
1190 DATA C6BD08069D0006A9A29901D295A9
A900A885AE95BD9D2C06BD18069D0406B5B085
A0B5B485A1B1A0C955D00B98,255
1200 DATA 95C6A4A89901D24C1A24C956D005
A9244C0F24C97FD009F6B0D0ABF6B44C4A23C9
FFD010A90095C6A4A89901D2,117
1210 DATA A90195B94CF2204A26AE4A26AEA0
00C9079006E907C84CC02348B9012485AD68A8
B9082465ADA4AE7905248SAD,985
1220 DATA BD2406C981B00B65ADC9259024E9
244CE723297F85AFA5ADE5AFC9259013692234C
F823000C182400FF01000204,488
1230 DATA 0507090BA8B9D82695C2A4A89900
D2F6B0D002F6B4B5B085A0B5B485A1A000949C
B1A00A369C4AA8B93B249D30,276
1240 DATA 064C27210406080C10182030406O
80C0A227A9009D5728CA10FA602048246A278A0
272071264CC521A203A006DE,941
1250 DATA 0406D021B5BDF00AB5C29900D2D6
BD4C8424B5C2F00F38FD14069900D2F6BDBD18
069D04068888CA10D56085D4,490
1260 DATA A2ABA0272068268SD520AAD920E6
D8D8A0FFC8B1F3291F996628B1F310F460A5C1
85AFA90485A6A03088D0FD20,808
1270 DATA 1826C6A6D0F4206024C6AFD0E960
2029258D08D22065E4A203A006A900959C95B0
95B495C69D24069D20069D1C,975
1280 DATA 06A9019D2C0695B9A9AA9D2806A9
A095A9B99500D00295B9CA8888810D160A5B905
BA05BB05BCD012A5B8F005C6,498
1290 DATA B84CE721204824202925 4CC5214C
FE21A206A9009D01D29D00D2CACA10F660A511
D029C611202925B5B8204824,988
1300 DATA 4CC521A209A9409D3E069D5C06A9
019D3406A9009D5206A9A29D4806CA10E660A9
0085AFCE7A2820C626C99BF0,796
1310 DATA 39C97ED013A5AFF0F1C6AFA6AFA9
FF9D7A28FE7B284C6D25A6AFE003F0DCC93090
D8C93AB0D4A6AF9D6606E91F,478
1320 DATA 9D7A28A9FF9D7B28E6AF4C6D25A6
AFD003E6AF60A96685F3A90685F4A9009D6606
85F22000D820D2D9D8A5D4A6,695
1330 DATA D5F002A28060A29EA02720682685
AFCE6B2820C626A6AFC97ED0108AF0F4C6AFA9
FF9D6A28FE6B284CDF259D7F,793
1340 DATA 28C99BF01538E9209D6B28A9FF9D
6C28E6AFE8E011D0CE4CE825FE6B2860A203A0
06B5C6F042DE0006D03DC901,653
1350 DATA F007C902F0204C6226F6A9B5A999
01D2DD2806F006BD08064C4726F6C6BD0C069D
00064C6226B5A9DD1006F00C,488
1360 DATA 900AD6A9B5A99901D24C4426F6C6
8888CA10B560A95785A4A9284C7726A96B85A4
A92885A586A284A3A000B1A2,297
1370 DATA F01938E92091A4C84C7F26A020A2
7F8E1FD08E0AD4CA10F78810F260208C264C48
24A220203E20A9C49D4403A9,290
1380 DATA 269D4503A9039D4203A9049D4A03
A9009D4B034C56E44B9BA220A9009D48039D49
03A9079D42034C56E4F3E6D9,696
1390 DATA CCC1B6ADA2999088807972 6C6660
5B555 14C4844403C3935322F2D2A282523211F
1D70707047CA270770700707,701
1400 DATA 0707077070070741FD2654454D50
4F20312D3235353A0050485241534520233 12D
393A004154544143 4B20312D,338
1410 DATA 3235353A00444543415920312D32
35353A00562E52414E47452030 2D3235353A00
562E535045454420312D3235,816
1420 DATA 353A005055534820425245414B20
544F2053544F50004E4F204441544100564F4C
2E2044524F5020302D373A00,526
1430 DATA 52455045415453203 12D32353 53A
0044455649435204E414D453F004355525245
4E542056414C55453A00492F,352
1440 DATA 4F204552524F5220434F44453A00
00006D75736963007379 6E7468657369 7A6572
0000000062790000 6B656E00,536
1450 DATA 636F6C6C6965720000000002C69
7374656E0000000000000000000000000032
65747269657665000000000,679
1460 DATA 000000000000002E756D62657200
0000000000000000000000000306872617365
006368616E67650000000000,457
1470 DATA 000034656D706F00000000000000
0000000000000000000000000000000000000
0000000000000000000000,942
```

•

CHECKSUM DATA

(see p. 70)

```
1 DATA 353,955,686,427,745,192,617,545
,276,445,496,549,150,536,720,7692
1020 DATA 29,320,981,723,704,675,689,6
23,563,432,681,654,517,704,729,9024
1170 DATA 745,704,992,56,935,803,904,4
41,866,967,735,932,625,970,467,11142
1320 DATA 153,284,95,914,846,793,840,5
19,158,167,295,209,207,938,802,7220
1470 DATA 471,471
```

•

```
2 REM *** CASSETTE VERSION ***
65 IF PASS=2 THEN FOR X=1 TO 109:PUT #
1,0:NEXT X:CLOSE #1:END
70 ? "READY CASSETTE AND PRESS RETURN"
;:OPEN #1,8,128,"C:":RESTORE 200:FOR X
=1 TO 35:READ N:PUT #1,N:NEXT X
200 DATA 0,18,221,31,255,31,169,60,141
,2,211,169,0,141,231,2,133,14,169,56,1
41,232,2
210 DATA 133,15,169,0,133,10,169,32,13
3,11,24,96
```

●

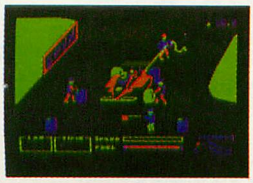
```
2 REM *** DISK VERSION ***
65 IF PASS=2 THEN PUT #1,224:PUT #1,2:
PUT #1,225:PUT #1,2:PUT #1,0:PUT #1,32
:CLOSE #1:END
70 ? "INSERT DISK WITH DOS, PRESS RETU
RN";:DIM IN$(1):INPUT IN$:OPEN #1,8,0,
"D:SYN.OBJ"
90 PUT #1,255:PUT #1,255:PUT #1,0:PUT
#1,32:PUT #1,111:PUT #1,40
```

●

```
; +--------------------+
; | MUSIC SYNTHESIZER  |
; |  by  Ken Collier   |
; +--------------------+
;
; Page Six variables.
;
          *=   $0600
;
SYNCNT    *=   *+4          syn counter
VIBCNT    *=   *+4          vibrato counter
ATTACK    *=   *+4          attack counter
DECAY     *=   *+4          decay counter
DROP      *=   *+4          minimum volume
VRANGE    *=   *+4          vibrato range
VSPEED    *=   *+4          vibrato speed
VCOUNT    *=   *+4          voice counter
LINECNT   *=   *+4          line counter
TRANSPOS  *=   *+4          transpose value
HIGHVOL   *=   *+4          maximum volume
GETNOTE   *=   *+4          need new note
DURCNT    *=   *+4          duration count
PATTACK   *=   *+10         phrase attack
PDECAY    *=   *+10         phrase decay
PDROP     *=   *+10         phrase min vol
PVRANG    *=   *+10         phrase v.range
PVSPEED   *=   *+10         phrase v.speed
NMBUF     *=   *+4          numeric buffer
;
; Page Zero variables.
;
          *=   $80
;
TABLE     *=   *+28         voices/phrases
TIED      *=   *+4          tied note flags
PTR       *=   *+2          temp index
PINDX     *=   *+2          print index
P2NDX     *=   *+2          print index 2
TMP2      *=   *+2          temporary 2
XSTORE    *=   *+1          voice index
VOLNOW    *=   *+4          current volume
NOTE      *=   *+1          note value
ACCID     *=   *+1          accidental
TMP       *=   *+1          temporary
PTRL      *=   *+4          phrase addr low
PTRH      *=   *+4          phrase addr hi
REPEAT    *=   *+1          repeat music
STATUS    *=   *+4          voice status
VSTAT     *=   *+4          vibrato status
TEMPO     *=   *+1          music speed
VNOTE     *=   *+4          vibrato sound
SYNSTAT   *=   *+4          syn status
;
FR0       =    $D4          F.P. register 0
CIX       =    $F2          F.P. index
INBUFF    =    $F3          F.P. buffer
;
; Miscellaneous labels.
;
CIOV      =    $E456        CIO vector
SIOINV    =    $E465        SIO init vector
AFP       =    $D800        ASCII to F.P.
FASC      =    $D8E6        F.P. to ASCII
IFP       =    $D9AA        INTEGER to F.P.
FPI       =    $D9D2        F.P. to INTEGER
ICCOM     =    $0342        command byte
ICBAL     =    $0344        buffer addr lo
ICBAH     =    $0345        buffer addr hi
ICBLL     =    $0348        buffer len lo
ICBLH     =    $0349        buffer len hi
ICAX1     =    $034A        aux info 1
ICAX2     =    $034B        aux info 2
OPEN      =    $03          open command
CLOSE     =    $0C          close command
READ      =    $07          get characters
EOF       =    136          end file flag
IOCB2     =    $20          IOCB #2 offset
IOCB3     =    $30          IOCB #3 offset
WSPACE    =    $2A00        workspace addr
BUFS      =    $2A00        buffer size
AUDF1     =    $D200        audio freq 1
AUDC1     =    $D201        audio volume 1
SDLSTL    =    $0230        display list
COLBK     =    $02C8        backgnd color
BRKKEY    =    $11          break key flag
CONSOL    =    $D01F        console speaker
SKCTL     =    $D20F        serial I/O ctrl
AUDCTL    =    $D208        audio control
WSYNC     =    $D40A        wait for sync
RETURN    =    $9B          return key
BACKS     =    $7E          backspace key
VLEN      =    $40          voice buff len
;
          *=   $2000        loads here!
;
; Setup the program.
;
START     CLD                clear decimal
          JSR  CLRTBL        clear tables
          JSR  SETDL         set display
          JSR  OPENKBD       open keyboard
          JMP  STOPIT        continue
;
; Input MUSIC COMPOSER files.
;
DATAIN    LDX  #IOCB3        use IOCB #3
          LDA  #NAME%*$FF    filename lo
          STA  ICBAL,X       buffer addr lo
          LDA  #NAME/256     filename hi
          STA  ICBAH,X       buffer addr hi
          JSR  OPENIT        open file
          BMI  ERROR         error?
          LDA  #READ         get characters
          STA  ICCOM,X       command byte
          LDA  #BUFF&*$FF    read buffer lo
          STA  ICBAL,X       buffer addr lo
          LDA  #BUFF/256     read buffer hi
          STA  ICBAH,X       buffer addr hi
          LDA  #BUFS&*$FF    read len lo
          STA  ICBLL,X       buffer len lo
          LDA  #BUFS/256     read len hi
          STA  ICBLH,X       buffer len hi
          JSR  CIOV          read music data
          CPY  #EOF          end of file?
          BNE  ERROR         yes.
CLOSEIT   LDA  #CLOSE        close cmd
          STA  ICCOM,X       command byte
          JMP  CIOV          close file
;
; Input/output error handler.
;
ERROR     STY  FR0           save error code
          JSR  BELL          ring bell
          LDX  #ETX%*$FF     I/O err msg lo
          LDY  #ETX/256      I/O err msg hi
          JSR  CURNT2        print msg+code
          LDX  #IOCB3        use IOCB 3
          JSR  CLOSEIT       close file
          PLA                pull two byte
          PLA                return address
          JMP  CHOICE        choose option
;
; Read through data and find
; PHRASE data,VOICE data, and
; MISCELLANEOUS data and store
; location in table.
;
POINTERS  LDA  #0            get zero
          STA  PTR           index lo
          LDY  #BUFF/256     buff addr hi
          STY  PTR+1         index hi
          TAY                set Y to zero
SET1      LDA  (PTR),Y       get byte
          CMP  #170          header byte?
          BNE  SET2          no.
          INY                increment index
          BNE  WHICH1        overflow?
          INC  PTR+1         yes, inc high
WHICH1    LDA  (PTR),Y       get byte
          BMI  MISC          misc. record?
          TAX                use as index
          STY  TABLE,X       save address
          LDA  PTR+1         of record in
          STA  TABLE+1,X     table
          JMP  ENDMARK       continue
MISC      LDA  #3            offset by 3
          STA  PTR           bytes in index
          LDA  (PTR),Y       get tempo
          ASL  A             times 2
          ASL  A             times 4
          ASL  A             times 8
          STA  TEMPO         save tempo
          LDA  #0            get zero to
          STA  PTR           reset index
ENDMARK   INY                inc index
          BNE  END1          overflow?
          INC  PTR+1         yes. inc hi
END1      LDA  (PTR),Y       get byte
          CMP  #255          end of record?
          BEQ  SET2          yes.
          INY                inc index
          BNE  ENDMARK       overflow?
          INC  PTR+1         yes. inc hi
          JMP  ENDMARK       continue
SET2      INY                inc index
          BNE  SET3          overflow?
          INC  PTR+1         yes. inc hi
SET3      LDA  (PTR),Y       get byte
          CMP  #255          end of file?
          BNE  SET1          no.
;
; Transfer the VOICE records
; to storage areas set aside.
;
          LDX  #6            set counter
TR1       LDA  TRVTBL,X      voice addr lo
          STA  TMP2          index lo
          LDA  #VC1/256      voice addr hi
          STA  TMP2+1        index hi
          LDA  #255          end of rec flag
          LDY  #1            record offset
          STA  (TMP2),Y      init record
          DEY                zero Y register
          LDA  TABLE+20,X    voice rec lo
          STA  PTR           index lo
```

```
        LDA TABLE+21,X  voice rec hi
        STA PTR+1       index hi
        BEQ TRN         empty voice rec
TR2     LDA (PTR),Y     put voice byte
        STA (TMP2),Y    in voice buf
        INY             inc index
        LDA (PTR),Y     put 2nd byte
        STA (TMP2),Y    in voice buf
        INY             inc index
        CMP #255        end of voice?
        BNE TR2         no.
TRN     DEX             yes. dec count
        DEX             index by 2
        BPL TR1         all voices?
        RTS             yes. continue
;
TRVTBL  .WORD VC1,VC2   voice addr
        .WORD VC3,VC4   table
;
; Erase old table before input.
;
CLRTBL  LDX #27         init index
        LDA #0          get zero
CT1     STA TABLE,X     store table
        DEX             dec index
        BPL CT1         done?
        RTS             yes. continue
;
; Check to see if a VOICE
; needs a new record.
;
NEEDREC LDX #3          init index
        LDA #6          init voice index
        STA XSTORE      save voice index
NEXTREC LDA STATUS,X    active?
        BEQ INCX        no.
        CMP #2          need voice rec?
        BNE GETREC      yes.
        JMP PLAYIT      play note
;
; Get a new VOICE record
; and process the info.
;
GETREC  JSR FINDREC     record byte
        CMP #7          valid commands?
        BCS STATUS1     no. end voice
        TAY             make index
        LDA JMPTBL,Y    jmp addr lo
        STA TMP2        jmp pointer lo
        LDA JMPTBH,Y    jmp addr hi
        STA TMP2+1      jmp pointer hi
        JSR FINDREC     operand byte
        JMP (TMP2)      go to it
;
JMPTBL  .BYTE NEXTREC&$FF,GOTO1&$FF
        .BYTE PLAY1&$FF,TRANS1&$FF
        .BYTE VOLUME1&$FF,NEXTREC&$FF
        .BYTE COUNT1&$FF
JMPTBH  .BYTE NEXTREC/256,GOTO1/256
        .BYTE PLAY1/256,TRANS1/256
        .BYTE VOLUME1/256,NEXTREC/256
        .BYTE COUNT1/256
;
; Clear status.
;
STATUS1 LDA #0          get zero
        STA STATUS,X    stop voice
;
; Increment to next VOICE.
;
INCX    DEX             dec index
        TXA             put in Acc
        ASL A           times 2
        STA XSTORE      voice index
        BPL NEXTREC     done yet?
        RTS             yes. continue
;
; Lookup next VOICE command
; in VOICE storage area.
;
FINDREC INC VCOUNT,X    voice cnt
        LDA FTBL,X      voice addr lo
        STA PTR         index lo
        LDA #VC1/256    voice addr hi
        STA PTR+1       index hi
        LDY VCOUNT,X    voice count
        LDA (PTR),Y     get byte
        RTS             continue
;
FTBL    .BYTE VC1&$FF   voice table
        .BYTE VC2&$FF
        .BYTE VC3&$FF
        .BYTE VC4&$FF
;
; If VOICE record is a GOTO
; then process the jump.
;
GOTO1   SEC             set carry
        SBC #1          subtract 1
        ASL A           multiply by 2
        DEC LINECNT,X   dec linecount
        BEQ NEXTREC     done? yes.
        STA VCOUNT,X    point to line
        JMP NEXTREC     get next record
;
; Set sound parameters for
; PHRASE number and set index to
; PHRASE's location in memory.
;
PLAY1   TAY             use as index
        LDA PATTACK,Y   attack shadow
        STA ATTACK,X    attack count
        LDA PDECAY,Y    decay shadow
        STA DECAY,X     decay count
        LDA PDROP,Y     min vol shadow
        STA DROP,X      min vol value
        LDA PVRANG,Y    f-range shadow
        STA VRANGE,X    freq. range
        LDA PVSPEED,Y   speed shadow
        STA VSPEED,X    vibrato speed
        TYA             index to Acc
        ASL A           multiply by 2
        TAY             to index again
        LDA #1          get one
        STA STATUS,X    get rec status
        STA GETNOTE,X   need a note
        LDA TABLE,Y     phrase addr lo
        STA PTRL,X      phrase tbl lo
        LDA TABLE+1,Y   phrase addr hi
        STA PTRH,X      phrase tbl hi
        BEQ PLAY2       data in table?
        INC STATUS,X    yes. play it
PLAY2   JMP NEXTREC     get next rec
;
; Set volume.
;
VOLUME1 ASL A           times 2
        ORA #$A0        pure tone
        STA HIGHVOL,X   save it
        JMP NEXTREC     get next rec
;
; Store the PHRASE count.
;
COUNT1  STA LINECNT,X   save count
        JMP NEXTREC     get next rec
;
; Do transposition.
;
TRANS1  STA TRANSPOS,X  transpose
        JMP NEXTREC     get next rec
;
; Iinitialize screen.
;
SETDL   LDA #DLIST&$FF  display lo
        STA SDLSTL      DL pointer lo
        LDA #DLIST/256  display hi
        STA SDLSTL+1    DL pointer hi
        LDA #$62        dark blue
        STA COLBK       background
        RTS             continue
;
; Get new file.
;
RETRIEVE JSR DEFAULT    do defaults
        JSR CLRTBL      clear table
        JSR NAMEIT      get filename
        JSR DATAIN      read data
        JSR POINTERS    set pointers
        JMP NMBR1       back to menu
;
; Get user input from keyboard.
;
CHOICE  JSR GETKEY      get character
        PHA             push value
        JSR ERASE       erase window
        PLA             pull value
        CMP #'L         play the music?
        BEQ LISTEN      yes.
        CMP #'R         load file?
        BEQ RETRIEVE    yes.
        CMP #'N         repeat count?
        BEQ NUMBER      yes.
        CMP #'P         phrase change?
        BEQ CPHRASE     yes.
        CMP #'T         tempo change?
        BEQ CTEMPO      yes.
        JSR BELL        ring bell
        JMP CHOICE      try again
;
; LISTEN to the music.
;
LISTEN  LDA TABLE+21    OR together
        ORA TABLE+23    high bytes of
        ORA TABLE+25    voice pointers
        ORA TABLE+27    to see if any
        BNE LIS1        are active?
        JMP NODATA      no.
LIS1    LDX #QTX&$FF    play msg lo
        LDY #QTX/256    play msg hi
        JSR PRINTD      print msg
        JSR PDEFAULT    set defaults
PLAYING JSR NEEDREC     voice rec
        JSR DELAY       synthesizer
        JSR BRKCHK      break key?
        JMP FINISHED    all done?
;
; Store # times to repeat music.
;
NUMBER  LDX #NMTX&$FF   number msg lo
        LDY #NMTX/256   number msg hi
        JSR PRINTD      print msg
        JSR GETNUM      get number
        BPL NMBR0       range error?
NMBRE   JSR RINGER      yes. ring bell
        JMP NUMBER      try again
NMBR0   BNE NMBR1       return?
        TAX             no. put in X
        BEQ NMBRE       value zero?
        DEX             no. 1-255
        STX REPEAT      store 0-254
NMBR1   JSR ERASE       erase window
        JMP CHOICE      goto menu
;
; Change TEMPO of playback.
;
CTEMPO  LDA TEMPO       get tempo
        JSR CURRENT     print it
        LDX #TTX&$FF    tempo msg lo
        LDY #TTX/256    tempo msg hi
        JSR TXNUM       ? msg / get num
        BPL TEM1        numeric error?
TEME    JSR RINGER      yes. ring bell
        JMP CTEMPO      try again
TEM1    BNE NMBR1       return?
        TAX             no.
        BEQ TEME        zero value?
        STA TEMPO       no. 1-255
        JMP NMBR1       continue
;
; Change parameters of
; sound for one phrase.
;
CPHRASE LDX #PTX&$FF    phrase msg lo
        LDY #PTX/256    phrase msg hi
        JSR TXNUM       print/get num
        BNE NMBR1       return?
        CMP #10         no. >9?
        BCC C1C         no.
C1E     JSR RINGER      ring bell
        JMP CPHRASE     try again
C1C     TAX             test zero flag
        BEQ C1E         num = zero?
        STA XSTORE      no. save number
        ASL A           times 2
        TAY             use as index
        LDA TABLE+1,Y   phrase addr hi
        BNE C3          active phrase?
        JMP NODATA      no.
C3      JSR ERASE       erase window
        LDX XSTORE      get phrase #
        LDA PATTACK,X   attack value
        JSR CURRENT     print it
        LDX #ATX&$FF    attack msg lo
        LDY #ATX/256    attack msg hi
        JSR TXNUM       print/get num
        BPL C3A         numeric error?
C3E     JSR BELL        no. ring bell
        JMP C3          try again
C3A     BNE C5A         return?
        TAX             no. test zero
        BEQ C3E         zero value?
        LDX XSTORE      no. get index
        STA PATTACK,X   save attack
C5A     JSR ERASE       erase window
        LDX XSTORE      get index
        LDA PDECAY,X    decay value
        JSR CURRENT     print it
        LDX #DTX&$FF    decay msg lo
        LDY #DTX/256    decay msg hi
        JSR TXNUM       print/get num
        BPL C5B         numeric error?
C5E     JSR BELL        yes. ring bell
        JMP C5A         try again
C5B     BNE C6A         return?
        TAX             no. test zero
        BEQ C5E         zero?
        LDX XSTORE      no. get index
        STA PDECAY,X    new decay
C6A     JSR ERASE       erase window
        LDX XSTORE      get index
        LDA #$AE        maximum volume
        SEC             set carry flag
        SBC PDROP,X     sub min volume
        LSR A           0..14 => 0..7
        JSR CURRENT     print it
        LDX #VTX&$FF    v.drop msg lo
        LDY #VTX/256    v.drop msg hi
        JSR TXNUM       print/get num
        BMI C6E         numeric error?
        BNE C8A         no. return?
        CMP #8          no. >7?
        BCC C8C         no. continue
C6E     JSR BELL        yes. ring bell
        JMP C6A         try again
C8C     ASL A           multiply by 2
        STA TMP         use later
        LDA #$AE        maximum volume
        SEC             set carry
        SBC TMP         subtract drop
        LDX XSTORE      get index
        STA PDROP,X     minimum volume
```

PITSTOP.™
WHERE WINNING IS THE PITS.



You'll never make Grand Prix champion just driving in circles.

You've got to stop sometime. The question is when. Right now you're in the lead. But the faster you go, the more gas you consume. And the quicker your tires wear down.

If you do pull into the pits, though, you lose precious seconds. So it's up to you to make sure the pit crew is quick with those tires. And careful with that gas. Otherwise, poof! you're out of the race.

See your retailer for available computer formats.

So what'll it be, Mario? Think your tires will hold up for another lap? Or should you play it safe and go get some new ones?

Think it over. Because Pitstop™ is the one and only road race game where winning is more than just driving. It's the pits.

Goggles not included.

One or two players; 6 racecourses, joystick control.

EPYX COMPUTER SOFTWARE

STRATEGY GAMES FOR THE ACTION-GAME PLAYER.

Vietnam, 5/75. Crashing through the dense jungle foliage, you hear the distant fire of the NVA camp guards. You've escaped, but you have a long way to go before you reach Saigon — if you ever do. Uncle Sam is pulling out, and Saigon holds your only hope of ever going home.

SAIGON: THE FINAL DAYS

By Jyym Pearson, Robyn Pearson & Norman Sailer

Gritty realism and historic fact blend to form a unique adventuring experience that plunges you into a controversial chapter of recent history.

Visit your local dealer *today*, or call us for your copy!

800-327-7172

Available for your 16K TRS-80 Model 1 or 3, in either tape or disk format, for only $24.95.
The Atari tape version (needs 16K) is $24.95.
The Apple version has great full-color hi-res graphics for $34.95.

Adventure® INTERNATIONAL
A DIVISION OF SCOTT ADAMS, INC.

BOX 3435 • LONGWOOD, FL 32750
(305) 862-6917

149

Photo by Paul Grupp

```
C8A     JSR ERASE       erase window
        LDX XSTORE      get index
        LDA PVRANG,X    vibrato range
        JSR CURRENT     print it
        LDX #RTX&$FF    v.range msg lo
        LDY #RTX/256    v.range msg hi
        JSR TXNUM       print/get num
        BPL C8B         numeric error?
        JSR BELL        yes. ring bell
        JMP C8A         try again
C8B     BNE C10A        return?
        LDX XSTORE      no. get index
        STA PVRANG,X    vibrato range
C10A    JSR ERASE       erase window
        LDX XSTORE      get index
        LDA PVSPEED,X   vibrato speed
        JSR CURRENT     print it
        LDX #STX&$FF    v.speed msg lo
        LDY #STX/256    v.speed msg hi
        JSR TXNUM       print/get num
        BPL C10B        numeric error?
C10E    JSR BELL        yes. ring bell
        JMP C10A        try again
C10B    BNE C11A        return?
        TAX             no. test zero
        BEQ C10E        yes. error
        LDX XSTORE      get index
        STA PVSPEED,X   vibrato speed
C11A    JMP NMBR1       continue
;
TXNUM   JSR PRINTD      print string
        JMP GETNUM      get number
;
; Play the notes.
;
PLAYIT  LDY XSTORE      get index
        LDA GETNOTE,X   need note?
        BNE NEWNOTE     yes. get a note
        DEC DURCNT,X    dec note length
        BEQ NOTEDONE    note done? yes.
        JMP INCX        another voice
NOTEDONE LDA TIED,X     note tied?
        BNE NEWNOTE     yes. skip
        STA AUDF1,Y     poke ultrasonic
        STA AUDC1,Y     zero volume
NEWNOTE INC PTRL,X      note pntr lo
        BNE NW2         overflow? no.
        INC PTRH,X      note pntr hi
NW2     LDA TIED,X      note tied?
        BNE NW3A        yes. skip
        LDA #1          attack token
        STA SYNSTAT,X   set status
        LDA ATTACK,X    attack count
        STA SYNCNT,X    save count
        LDA #$A2        starting volume
        STA AUDC1,Y     poke hardware
        STA VOLNOW,X    save volume
NW3A    LDA #0          get zero
        TAY             set Y to zero
        STA ACCID       init accidental
        STA VSTAT,X     vibrato status
        STA GETNOTE,X   not any more
        LDA VSPEED,X    vibrato speed
        STA VIBCNT,X    speed count
        LDA PTRL,X      phrase ptr lo
        STA PTR         save pointer lo
        LDA PTRH,X      phrase ptr hi
        STA PTR+1       save pointer hi
        LDA (PTR),Y     get note
        CMP #85         rest?
        BNE NW3         no. skip
        TYA             Acc = 0
        STA SYNSTAT,X   synthesizer off
        LDY XSTORE      get index
        STA AUDC1,Y     zero volume
        JMP GETDURA     get duration
NW3     CMP #86         C#6?
        BNE NW4         no. skip
        LDA #36         C6 index
        JMP NW40        get frequency
NW4     CMP #127        measure?
        BNE NW5         no. skip
        INC PTRL,X      note pointer lo
        BNE NEWNOTE     overflow? no.
        INC PTRH,X      note pointer hi
        JMP NEWNOTE     get a new note
NW5     CMP #255        end of phrase?
        BNE NW6         no. skip
        LDA #0          get a zero
        STA SYNSTAT,X   synthesizer off
        LDY XSTORE      get index
        STA AUDC1,Y     zero volume
        LDA #1          get voice token
        STA STATUS,X    set status
        JMP NEXTREC     next voice rec
NW6     LSR A           divide by 2
        ROL ACCID       carry into ACCID
        LSR A           divide by 4
        ROL ACCID       carry into ACCID
        LDY #0          init octave #
NW7     CMP #7          compare 0..7
        BCC NW8         in range? yes.
        SBC #7          subtract 7
        INY             inc octave #
        JMP NW7         try again
NW8     PHA             save note #
        LDA OCTBL,Y     octave offset
        STA NOTE        save octave
        PLA             get note #
        TAY             use as index
        LDA NOTBL,Y     note offset
        ADC NOTE        add octave
        LDY ACCID       accidental
        ADC ACTBL,Y     accid offset
        STA NOTE        save note
        LDA TRANSPOS,X  transpose
        CMP #129        up/down point
        BCS TRANSDN     down?
        ADC NOTE        add note
NW36    CMP #37         in range?
        BCC NW40        no. skip
        SBC #36         make in range
        JMP NW36        try again
TRANSDN AND #$7F        transpose
        STA TMP         save it
        LDA NOTE        get note
        SBC TMP         transpose it
NW37    CMP #37         in range?
        BCC NW40        yes. skip
        ADC #36-1       make in range
        JMP NW37        try again
;
OCTBL   .BYTE 0,12,24,36 octave table
ACTBL   .BYTE 0,$FF,1   accidental table
NOTBL   .BYTE 0,2,4,5   note table
        .BYTE 7,9,11
;
NW40    TAY             use as index
        LDA NOTES,Y     get actual note
        STA VNOTE,X     save it
        LDY XSTORE      get index
        STA AUDF1,Y     poke hardware
;
; Compute note duration.
;
GETDURA INC PTRL,X      pointer tbl lo
        BNE GDUR1       overflow?
        INC PTRH,X      yes.pointer hi
GDUR1   LDA PTRL,X      pointer tbl lo
        STA PTR         pointer lo
        LDA PTRH,X      pointer tbl hi
        STA PTR+1       pointer hi
        LDY #0          get zero
        STY TIED,X      zero tied flag
        LDA (PTR),Y     duration byte
        ASL A           minus => carry
        ROL TIED,X      carry => 1=tied
        LSR A           rotate back
        TAY             use as index
        LDA GDTBL-1,Y   actual duration
        STA DURCNT,X    save it
        JMP INCX        next voice
;
GDTBL   .BYTE 4,6,8     duration table
        .BYTE 12,16,24
        .BYTE 32,48,64
        .BYTE 96,128,192
;
; Erase text at screen bottom.
;
ERASE   LDX #39         40 bytes 0..39
        LDA #0          display blank
ERSC    STA TX1,X       store in window
        DEX             decrement index
        BPL ERSC        done?
        RTS             yes. continue
;
; If no data in memory.
;
NODATA  JSR ERASE       erase window
        LDX #NTX&$FF    no data msg lo
        LDY #NTX/256    no data msg hi
        JSR PRINTD      print it
        JMP CHOICE      goto menu
;
; Add VIBRATO.
;
VIBRATO LDX #3          init index
        LDY #6          hardware index
VBRT    DEC VIBCNT,X    v.speed count
        BNE VNXTX       done? yes.
        LDA VSTAT,X     vibrato status
        BEQ RAISE       0=raise pitch
        LDA VNOTE,X     get frequency
        STA AUDF1,Y     poke hardware
        DEC VSTAT,X     vstat=0
        JMP PUTVIB      continue
RAISE   LDA VNOTE,X     get frequency
        BEQ VNXTX       0=no sound
        SEC             set carry
        SBC VRANGE,X    sub vib range
        STA AUDF1,Y     poke hardware
        INC VSTAT,X     vstat=1
PUTVIB  LDA VSPEED,X    vibrato speed
        STA VIBCNT,X    v.speed count
VNXTX   DEY             hardware index
        DEY             do it twice
        DEX             dec index
        BPL VBRT        done? no.
        RTS             continue
;
; Print current sound value.
;
CURRENT STA FR0         save value
        LDX #CTX&$FF    current msg lo
        LDY #CTX/256    current msg hi
CURNT2  JSR PRINTU      print in top
        STA FR0+1       zero in F.P. hi
        JSR IFP         INTEGER to F.P.
        JSR FASC        F.P. to ASCII
        CLD             clear decimal
        LDY #$FF        init index
PNUM    INY             inc index
        LDA (INBUFF),Y  get ASCII
        AND #$1F        to display code
        STA TX1+15,Y    put msg window
        LDA (INBUFF),Y  get ASCII
        BPL PNUM        end of string
        RTS             yes. continue
;
; Add delay to create TEMPO.
;
DELAY   LDA TEMPO       music tempo
        STA TMP         save it
DEL     LDA #4          do SYN 4 times
        STA TMP2        save count
DEL1    LDY #48         delay loop cnt
DEL2    DEY             dec count
        BNE DEL2        done? no.
        JSR SYN         synthesizer
        DEC TMP2        dec 4 times cnt
        BNE DEL1        done? no.
        JSR VIBRATO     vibrato effect
        DEC TMP         dec tempo value
        BNE DEL         done? no.
        RTS             continue
;
; Plug in default values.
;
PDEFAULT JSR SHUTOFF    zero sounds
        STA AUDCTL      zero audio ctrl
        JSR SIOINV      init hardware
        LDX #3          init index
        LDY #6          init tbl index
DEF1    LDA #0          get zero
        STA TIED,X      zero tied
        STA PTRL,X      phrase tbl lo
        STA PTRH,X      phrase tbl hi
        STA SYNSTAT,X   syn status
        STA TRANSPOS,X  transpose
        STA LINECNT,X   voice line
        STA VCOUNT,X    voice rpt cnt
        LDA #1          get one
        STA GETNOTE,X   need new note
        STA STATUS,X    need new record
        LDA #$AA        define high vol
        STA HIGHVOL,X   new high vol
        LDA #$A0        current volume
        STA VOLNOW,X    new volnow
        LDA TABLE+21,Y  voice tbl hi
        BNE DEF2        active? yes.
        STA STATUS,X    shut off voice
DEF2    DEX             dec index
        DEY             dec table index
        DEY             do it twice
        BPL DEF1        done? no.
        RTS             continue
;
; Check if all VOICEs are done.
;
FINISHED LDA STATUS     OR status
        ORA STATUS+1    of all voices
        ORA STATUS+2    together. if
        ORA STATUS+3    result <> 0,
        BNE ALLP        keep playing
        LDA REPEAT      play again?
        BEQ ALL3        no. end it
        DEC REPEAT      dec repeat val
        JMP LISTEN      play it again
ALL3    JSR ERASE       erase window
        JSR SHUTOFF     stop all sound
        JMP CHOICE      goto menu
ALLP    JMP PLAYING     cont playing
;
; Stop all sound.
;
SHUTOFF LDX #6          init index
        LDA #0          zero sound val
SHUT    STA AUDC1,X     zero volume
        STA AUDF1,X     zero frequency
        DEX             dec index
        DEX             do it again
        BPL SHUT        done? no.
        RTS             continue
;
; Check if the BREAK key
; has been pushed.
;
```

```
BRKCHK  LDA BRKKEY      get break key
        BNE BRKX        key hit? no.
STOPIT  DEC BRKKEY      reset break
        JSR SHUTOFF     turn off sound
        STA REPEAT      reset repeat
        JSR ERASE       erase window
        JMP CHOICE      goto menu
;
; Set sound values to default.
;
DEFAULT LDX #9          do ten phrases
SYND    LDA #64         default
        STA PDECAY,X    decay
        STA PVSPEED,X   vibrato speed
        LDA #1          default
        STA PATTACK,X   attack
        LDA #0          default
        STA PVRANG,X    vibrato range
        LDA #$A2        default
        STA PDROP,X     minimum volume
        DEX             dec index
        BPL SYND        done? no.
BRKX    RTS             continue
;
; Input and print a number.
;
GETNUM  LDA #0          get zero
        STA TMP         save as pointer
        DEC TX2+15      cursor $FF
GN1     JSR GETKEY      get a letter
        CMP #RETURN     is it a return
        BEQ GN3         yes. goto exit
        CMP #BACKS      backspace?
        BNE GN2         no. goto GN2
        LDA TMP         get pointer
        BEQ GN1         left margin=0?
        DEC TMP         dec pointer
        LDX TMP         use as index
        LDA #$FF        cursor char
        STA TX2+15,X    put cursor
        INC TX2+16,X    make space
        JMP GN1         get a char
GN2     LDX TMP         get pointer
        CPX #3          three digits?
        BEQ GN1         yes. try again
        CMP #'0         .LT. ASCII 0?
        BCC GN1         yes. try again
        CMP #'9+1       .GT. ASCII 9?
        BCS GN1         yes. try again
        LDX TMP         get index
        STA NMBUF,X     put in buffer
        SBC #32-1       display code
        STA TX2+15,X    put in window
        LDA #$FF        cursor char
        STA TX2+16,X    put cursor
        INC TMP         inc position
        JMP GN1         get a char
GN3     LDX TMP         get pointer
        BNE GN4         any characters?
        INC TMP         no. set <>zero
        RTS             continue
GN4     LDA #NMBUF&$FF  buffer adr lo
        STA INBUFF      inbuff pntr lo
        LDA #NMBUF/256  buffer adr hi
        STA INBUFF+1    inbuff pntr hi
        LDA #0          non numeric
        STA NMBUF,X     number +1
        STA CIX         FP offset
        JSR AFP         ASCII to F.P.
        JSR FPI         F.P. to INTEGER
        CLD             clear decimal
        LDA FR0         get value
        LDX FR0+1       test >255
        BEQ GN5         no. ok.
        LDX #$80        set minus flag
GN5     RTS             continue
;
; Name the INPUT device.
;
NAMEIT  LDX #ITX&$FF    device msg lo
        LDY #ITX/256    device msg hi
        JSR PRINTU      print it
        STA TMP         init index
        DEC TX2         init cursor $FF
NMT1    JSR GETKEY      get character
        LDX TMP         get index
        CMP #BACKS      backspace?
        BNE NMT2        no. on screen
NMTD    TXA             yes. test index
        BEQ NMT1        left margin=0?
        DEC TMP         position left
        LDA #$FF        get cursor
        STA TX2-1,X     put screen
        INC TX2,X       turn off cursor
        JMP NMT1        try again
NMT2    STA NAME,X      ascii name buf
        CMP #RETURN     return key?
        BEQ NMT5        yes. goto NMT5
        SEC             set carry
        SBC #32         display code
        STA TX2,X       put screen
        LDA #$FF        get cursor
        STA TX2+1,X     put screen
        INC TMP         inc index sav
        INX             inc index
        CPX #17         18 char yet?
        BNE NMT1        no. get more
        JMP NMTD        delete last
NMT5    INC TX2,X       delete cursor
        RTS             continue
;
; Controls operation of
; the SYNTHESIZER section,
; attack and decay.
;
SYN     LDX #3          init index
        LDY #6          hardware index
SYN2    LDA SYNSTAT,X   voice active?
        BEQ NXTX        no. do another
        DEC SYNCNT,X    syn dur count
        BNE NXTX        count done? no.
        CMP #1          do attack?
        BEQ ATTCK       yes.
        CMP #2          do decay?
        BEQ DECY        yes.
        JMP NXTX        next voice
ATTCK   INC VOLNOW,X    inc volume
        LDA VOLNOW,X    current volume
        STA AUDC1,Y     poke hardware
        CMP HIGHVOL,X   at maximum?
        BEQ GODECAY     yes. now decay
        LDA ATTACK,X    attack duration
        JMP DEC3        continue
GODECAY INC SYNSTAT,X   make decay
DEC2    LDA DECAY,X     decay duration
DEC3    STA SYNCNT,X    syn dur count
        JMP NXTX        next voice
DECY    LDA VOLNOW,X    current volume
        CMP DROP,X      compare low vol
        BEQ GOSUSTN     =low volume?
        BCC GOSUSTN     <low volume?
        DEC VOLNOW,X    dec volume
        LDA VOLNOW,X    current volume
        STA AUDC1,Y     poke hardware
        JMP DEC2        continue
GOSUSTN INC SYNSTAT,X   stop voice
NXTX    DEY             hardware index
        DEY             do it twice
        DEX             index
        BPL SYN2        all done? no.
        RTS             continue
;
; Print text to screen.
;
PRINTU  LDA #TX1&$FF    upper line lo
        STA P2NDX       ? to pointer lo
        LDA #TX1/256    upper line hi
        JMP PR0         skip
PRINTD  LDA #TX2&$FF    lower line lo
        STA P2NDX       ? to pointer lo
        LDA #TX2/256    lower line hi
PR0     STA P2NDX+1     ? to pointer hi
        STX PINDX       X=text addr lo
        STY PINDX+1     Y=text addr hi
        LDY #0          init index
PR1     LDA (PINDX),Y   get ascii char
        BEQ PRX         end of string?
        SEC             no. set carry
        SBC #32         display code
        STA (P2NDX),Y   put screen
        INY             inc index
        JMP PR1         again
;
; Ring bell (CTRL/2).
;
BELL    LDY #$20        duration count
BELL1   LDX #$7F        inner loop
BELL2   STX CONSOL      poke speaker
        STX WSYNC       wait for sync
        DEX             dec inner loop
        BPL BELL2       done? no.
        DEY             dec duration
        BPL BELL1       done? no.
PRX     RTS             continue
;
; Ring bell and erase text.
;
RINGER  JSR BELL        ring bell
        JMP ERASE       erase text
;
; Open keyboard for INPUT.
;
OPENKBD LDX #IOCB2      use IOCB #2
        JSR CLOSEIT     close #2
        LDA #KBD&$FF    file name lo
        STA ICBAL,X     buffer addr lo
        LDA #KBD/256    file name hi
        STA ICBAH,X     buffer addr hi
OPENIT  LDA #OPEN       open command
        STA ICCOM,X     command byte
        LDA #4          input option
        STA ICAX1,X     aux1 byte
        LDA #0          zero
        STA ICAX2,X     aux2 byte
        JMP CIOV        open file
;
;Keyboard file name.
;
KBD     .BYTE "K",RETURN
;
; Get character from keyboard.
;
GETKEY  LDX #IOCB2      use IOCB #2
        LDA #0          buffer length
        STA ICBLL,X     buf length lo
        STA ICBLH,X     buf length hi
        LDA #READ       get character
        STA ICCOM,X     command byte
        JMP CIOV        get a character
;
; Values for NOTE pitches
; from low to high.
;
NOTES   .BYTE 243,230,217,204
        .BYTE 193,182,173,162
        .BYTE 153,144,136,128
        .BYTE 121,114,108,102
        .BYTE 96,91,85,81
        .BYTE 76,72,68,64
        .BYTE 60,57,53,50
        .BYTE 47,45,42,40
        .BYTE 37,35,33,31
        .BYTE 29
;
; Display list.
;
DLIST   .BYTE $70,$70,$70,$47
        .WORD SCREEN
        .BYTE 7,$70,$70,7
        .BYTE 7,7,7,7
        .BYTE $70,$70,7,7
        .BYTE $41
        .WORD DLIST
;
; Words at screen bottom.
;
TTX     .BYTE "TEMPO 1-255:",0
PTX     .BYTE "PHRASE #1-9:",0
ATX     .BYTE "ATTACK 1-255:",0
DTX     .BYTE "DECAY 1-255:",0
RTX     .BYTE "V.RANGE 0-255!",0
STX     .BYTE "V.SPEED 1-255:",0
QTX     .BYTE "PUSH BREAK TO STOP",0
NTX     .BYTE "NO DATA",0
VTX     .BYTE "VOL. DROP 0-7:",0
NMTX    .BYTE "REPEATS 1-255:",0
ITX     .BYTE "DEVICE NAME?",0
CTX     .BYTE "CURRENT VALUE:",0
ETX     .BYTE "I/O ERROR CODE:",0
;
; Data for screen menu.
;
SCREEN  .BYTE 0,0,"music",0
        .BYTE "synthesizer",0
        .BYTE 0,0,0,"by",0,0
        .BYTE "ken",0,"collier",0,0
        .BYTE 0,0,0,"L-32,"isten"
        .BYTE 0,0,0,0,0,0,0,0,0,0,0
        .BYTE 0,0,0,"R-32,"etrieve"
        .BYTE 0,0,0,0,0,0,0,0,0,0
        .BYTE 0,0,0,"N-32,"umber"
        .BYTE 0,0,0,0,0,0,0,0,0,0,0
        .BYTE 0,0,0,"P-32,"hrase",0
        .BYTE "change",0,0,0,0
        .BYTE 0,0,0,"T-32,"empo"
        .BYTE 0,0,0,0,0,0,0
        .BYTE 0,0,0,0,0,0,0
        .BYTE 0                 next line offset
;
TX1     *=  *+20        text window 1
TX2     *=  *+20        text window 2
NAME    *=  *+20        file name buf
;
        *=  WSPACE
;
VC1     *=  *+VLEN      voice space 1
VC2     *=  *+VLEN      voice space 2
VC3     *=  *+VLEN      voice space 3
VC4     *=  *+VLEN      voice space 4
;
BUFF    *=  *+BUFS      file space
;
; RUN address
;
        *=  $02E0       file run addr
        .WORD START     addr to goto
;
        .END
```

EASTCOAST SOFTWARE

ecs...inflation-fighting prices.

No hidden charges! No credit card fees!

ATARI

SOFTWARE	LIST	SALE
747 Landing Simul C/D	$22.95	$18.25
AE D	**34.95**	**24.15**
Ali Baba/40 Thieves D	32.95	23.25
Alien Ambush D	29.95	20.70
Alien Swarm C	29.95	21.20
Alpha Shield R	39.95	28.30
Anti-Sub Patrol C	19.95	13.75
Anti-Sub Patrol D	29.95	20.70
Apple Cider Spider R	**39.95**	**28.30**
Apple Panic C/D	29.95	20.70
Assembler Editor R	59.95	46.40
Atari Basic R	59.95	46.40
Atari Writer R	**99.95**	**77.50**
Attack at EP-CYG-4 C	29.95	21.20
Attack at EP-CYG-4 D	32.95	23.35
Avalanche C/D	22.95	18.25
Bandits D	34.95	24.75
Bank Street Writer D	**69.95**	**48.35**
Baseball R	44.95	31.85
Basic A+ With OS/A+	80.00	56.65
Basic Compiler D	79.95	55.25
Battle For Normandy C/D	39.95	28.90
Battle of Shiloh C/D	39.95	28.90
Battling Bugs/Concent C	29.95	22.60
Blue Max C/D	**34.95**	**24.75**
Bookkeeper D	149.95	116.25
Canyon Climber C/D	24.95	17.25
Castle Wolfenstein D	29.95	20.95
Chess D	69.95	48.35
Choplifter R	44.95	31.05
Claim Jumper C/D	34.95	24.75
Clowns & Balloons C/D	24.95	17.25
Compu-Math/Decimals C	29.95	20.70
Compu-Math/Fractions C	29.95	20.70
Compu-Read C	19.95	14.15
Controller D	30.00	20.70
Conversational French C	59.95	46.40
Cosmic Balance D	39.95	28.90
Crossfire C/D	29.95	20.70
Crush Crumble Chomp C/D	29.95	20.70
Dark Crystal D	39.95	27.60
Data Perfect D	**129.95**	**99.70**
David's Midnight Magic D	34.95	24.15
Deadline D	**49.95**	**34.50**
Defender R	44.95	34.75
Delta Drawing D	59.95	43.05
Deluxe Invaders R	39.95	27.60
Dig Dug R	44.95	34.75
Dimension X C/D	**34.95**	**24.75**
Diskey D	49.95	35.35
Dneiper River Line D	30.00	21.25
Dog Daze C/D	22.95	18.25
Donkey Kong R	49.95	38.75
Drelbs C/D	34.95	24.20
Eastern Front C/D	29.95	23.85
Edit 6502 R	179.95	138.00
Empire of Overmind C	30.00	21.25
ET Phone Home R	49.95	38.75
Facemaker D	34.95	24.75
Family Finance D	49.95	38.75
Fantastic Voyage R	34.95	25.70
Figure Fun C	29.95	23.00
File It 2 System D	49.95	35.35
File Manager 800+	**99.95**	**69.00**
Flip Out D	29.95	21.20
Floyd of the Jungle C/D	**29.95**	**23.05**
Fort Apocalypse C/D	34.95	24.75
Fort Apocalypse R	34.95	26.35
Frogger C/D	34.95	24.10
Galaxian R	44.95	34.75
GFS Sorceress D	35.00	24.20
Gorf D	39.95	27.60
Gorf R	44.95	31.05
Hellcat Ace C/D	29.95	23.05
Hey Diddle Diddle D	29.95	20.70
Home Accountant D	**74.95**	**48.05**
Home Filing Mgr D	49.95	38.75
Home Financial MGMT C	29.95	23.00
Home Inventory D	19.95	15.05
Home Manager Kit D	79.95	62.00
Household Finance D	34.95	26.35
In Search of/Amazing D	39.95	27.60
Insta Edit C/D	19.95	14.15
Invitat.-Programming 1 C	24.95	19.25
Invitat.-Programming 2 C	29.95	23.15
Invitat.-Programming 3 C	29.95	23.15
It-Is-Balloon C/D	34.95	24.75
J. White Music Lessons C/D	29.95	21.20
Jawbreaker C/D	29.95	20.70
Jumbo Jet Pilot R	49.95	38.30
K-DOS D	89.95	62.75
K-razy Antiks R	39.95	28.30
K-razy Kritters R	39.95	28.30
K-razy Shootout R	39.95	28.30
K-Star Patrol R	39.95	28.30
Kindercomp D	29.95	20.70
Knights of the Desert C/D	39.95	28.30
Letter Perfect D	**149.95**	**103.60**
Letter Perfect-80 COL R	199.95	153.30
Letterman C/D	22.95	18.25
Lunar Leeper D	29.95	20.70
MAC/65 D	80.00	56.65
Macro Assemb/Text Edit D	89.95	70.00
Master Type D	**39.95**	**27.60**
Microsoft Basic II D	89.95	70.00
Miner 2049er R	**49.95**	**34.50**
My First Alphabet D	34.85	27.15
Nato Commander C/D	34.95	26.90
Number Blast C/D	15.95	12.70
Outlaw/Howitzer C/D	22.95	18.25
Pilot (Home Pkg) R	79.95	61.85
Pinball C/D	29.95	21.20
Poker Sam C/D	24.95	17.65
Pool 1.5 D	34.95	24.15
Pooyam D	**29.95**	**20.70**
Prepare for Sat D	119.95	84.90
Preppie C/D	29.95	20.70
Preppie II C/D	34.95	24.15
Preschool IQ Builder 1 C	18.95	13.45
Preschool IQ Builder 2 C	18.95	13.45
Preschool Library C	59.95	42.45
Programmer Kit R	59.95	51.35
Protector 2 D	34.95	24.20
Protector 2 R	44.95	31.15
Qix R	44.95	34.75
QS Reversi D	29.95	22.55
Rhymes & Riddles D	29.95	20.70
Sea Fox D	**29.95**	**20.70**
Sea Fox R	39.95	27.60
Serpentine C/D	34.95	24.15
Shadow World C/D	34.95	25.30
Shamus Case 2 C/D	34.95	24.75
Snake Byte D	29.95	21.20
Snooper Troops 1 D	44.95	32.50
Snooper Troops 2 D	44.95	32.50
Software Automouth D	59.95	41.45
Space Invaders R	34.95	28.45
Spelling Bee Games D	39.95	28.30
Squish 'Em R	39.95	28.30
Star Raiders R	44.95	34.75
Starcross D	**39.95**	**27.60**
Story Machine D	34.95	25.45
Strip Poker D	34.95	26.90
Submarine Commander R	49.95	38.30
Superman III R	49.95	38.75
Syn Assembler D	49.95	35.35
Teletalk D	**49.95**	**34.50**
Teletari D	39.95	28.30
Temple of Apshi C/D	39.95	27.60
Text Wizard D	49.95	34.50
Tigers in the Snow C/D	39.95	28.90
Tricky Tutorials #1-#6 C/D	119.95	84.90
Ultima II D	59.95	41.40
Ulysses & Gold Fleece D	39.95	27.60
Upper Reaches Apshai C/D	19.95	14.15
Visicalc (Special) D	**199.95**	**162.50**
Way Out D	39.95	28.30
Wingman C/D	34.95	26.90
Wizard of Wor D	39.95	27.60
Wizard of Wor R	44.95	31.05
Zaxxon C/D	**39.95**	**27.60**
Zork I, II, or III D	39.95	27.60

HARDWARE	LIST	SALE
Amdek 300A 12 In. Amber Mon	$199.00	$161.50
Amdek 300G 12 In. Green	179.00	143.40
Amdek Color I Monitor	**399.00**	**320.60**
Atari Joystick (Single)	**9.95**	**6.45**
Atari Paddles	14.95	11.50
Banana Printer	249.95	221.55
Gorilla 12-in Green Scrn	99.00	91.50
Intec Ram Card-16K	60.00	43.40
Intec Ram Card-32K	90.00	62.70
Intec Ram Card-48K	**115.00**	**83.15**
Intec Ram Card-64K	150.00	108.50
Koala Graphic Tablet-Atari	99.95	75.35
Kraft Joystick-Atari	16.95	13.00
Microbits Printer Interface	**99.95**	**88.15**
Microtek Ram Board-16K	89.00	50.30
Microtek Ram Board-32K	119.00	62.85
Mosaic 64K RAM (400/800)	199.00	152.60
MPC 32K Memory Board	139.50	107.00
NEC 1201 Green Monitor	285.00	166.15
NEC 1212 Color Monitor	399.00	310.00
NEC 1260 Green Monitor	149.95	113.50
NEC PC8023 Printer	599.00	441.85
NEC PC8025 Printer	975.00	825.00
Parallel Printer Cable	34.50	24.45
Percom ATA88-S1 1st Drive	488.00	461.80
PI-2 12 in Green Monitor	210.00	154.55
PI-3 12 in Amber Monitor	249.00	162.00
PI-4 9 in Amber Monitor	199.00	144.50
Rampower 128K Memory Board	379.95	319.15
Rana 1000 Disk Drive	**449.00**	**318.50**
Sanyo 8112 12 in Green Mon	260.00	204.60
Serial Modem Cable	39.50	28.50
Serial Printer Cable	34.50	28.50
Signalman Modem MK 11	99.00	81.40
Smartmodem - Stack	289.00	215.00
Starwriter F10-40PU Print	1,895.00	1,562.50
Taxan 12 in Amber Monitor	189.00	138.65
Taxan 12 in Green Monitor	179.00	131.60
Taxan Color Mntr W/Audio	399.00	305.50
TG Joystick - Atari	34.95	25.75
TG Track Ball	64.95	46.00
Wico 12-ft Joystick Ext	7.95	6.00
Wico Joystick	29.95	20.55
Wico Red Ball Joystick	34.95	21.80
Wico Trackball	49.95	38.40
Wico's "The Boss" Joystick	**19.95**	**14.10**

BOOKS	LIST	SALE
Atari Assembler Book	$12.95	$9.95
Atari Basic Book	10.95	8.40
Atari Basic Reference Man	10.95	8.50
Atari Games & Recreation	14.95	11.50
Atari Pilot For Beginners	14.95	11.50
Atari Sound & Graphics	10.95	8.40
Atari Tech Reference Notes	**29.95**	**23.75**
Book of Atari Software '83	**19.95**	**12.80**
Kids and the Atari	**19.95**	**14.45**
Your Atari Computer Book	**16.95**	**13.00**

SUPPLIES	LIST	SALE
Dust Cover - Atari 800	**10.98**	**8.50**
Elephant Disks SS/DD (10)	37.00	24.35
Elephant Disks SS/SD (10)	**34.00**	**21.15**
Epson MX-80 Ribbon 2-Pac	28.00	12.55
Flip-N-File (Atari Cart)	29.95	18.85
Flip-N-File (Disks)	**29.95**	**18.85**
Head Cleaning Kit	29.95	17.70
Maxell MD1 Disks (10 Pack)	51.90	29.40
NEC 8023 Ribbon 2-Pac	19.95	17.10
Starwriter Film Ribbon	5.50	4.50
Starwriter Nylon Ribbon	5.75	4.65
Trunk Floppy Storage	29.95	21.50
Verbatim MD 525 Softpack	**34.00**	**26.95**
Verbatim MD 525 Hard Pack	35.00	27.60

800-233-3237

For Orders Only

Atari® is a registered trademark of ATARI COMPUTER INC.

CHECK ONE: ☐ VISA ☐ MASTERCARD

Card#_____ Exp. _____

Telephone _____

Name _____

Address _____

City _____

State _____ Zip _____

Please enter my order for:_____

_____

_____

Please specify machine:_____

☐ *Please send free catalog.*

EASTCOAST SOFTWARE

40 Hockersville Rd.
Hershey, PA 17033

MAIL YOUR ORDER OR CALL

(717) 533-8125 or 533-8480

We Ship UPS — Shipping 1% ($2.50 min.)
C.O.D. Add an Additional $2.50
Hardware (Printers/Monitors)
Shipping is $2.50 + 6% (U.P.S.)
For Air Mail, Canadian, APO/FPO, or Overseas Orders,
Shipping $2.50 + 9% of Total Order (US Funds)

No Minimum Order
Visa-Mastercard-C.O.D.-Check

Prices Subject To Change

800-233-3237

VISA® master charge THE INTERBANK CARD

This ad prepared August, 1983.

DISK USERS:

DON'T LET THIS HAPPEN TO YOU!

by Jerry White

While working on a BASIC program very late one night, I had saved quite a few versions of my program using filenames such as "D:TEST1" and "D:TEST-2." Shortly before sunrise, I decided to call it a day. I saved the most current version of my still unfinished program as "D:TEST8." I then realized that I had previously saved a version called "D:TEST9," and that I had better clean up my work disk before shutting my bloodshot eyes.

I called DOS and took a quick look at the directory file. With my fingers running much faster than my brain, I renamed TEST8, TEST9. After pressing RETURN, I realized that I now had two programs with the filename TEST9. Since I wasn't sure which of the two TEST9 programs was the most current, I decided to rename one of them as "TEMP." Much to my surprise, DOS renamed both "TEST9" programs as "TEMP." Now what?

I went back to BASIC and loaded "TEMP." Knowing that the program in RAM was the first version of the "TEMP" found in the directory file, I listed it onto the screen to see if it was the most current version. No such luck. It was the second version of "TEMP" that I needed. Bat's breath!

I then tried to solve my problem by going back to DOS and deleting the first "TEMP." This was just not my night. DOS deleted both my "TEMP" files before I could do anything about it.

Having just received Adventure International's **Diskey** utility program, I used it to remove the DELETE flag from the second "TEMP" in the directory file. I had solved my problem but I was more than a bit cranky. It just seemed to be absurd that DOS should rename more than one filename at a time. Before getting to sleep that morning, I vowed to correct the situation so that it couldn't happen again.

After a quick look at COMPUTE's *Inside ATARI DOS*, I found that I could correct this problem by simply eliminating one BCC instruction. This can be done using two NOPs. If you'd like to change DOS.SYS version 2.OS so that RENAME effects only one file, just follow these instructions:

With your ATARI BASIC cartridge in place, boot up using a disk that contains DOS 20S. At the BASIC ready prompt, in immediate mode, enter the following commands then press RETURN.

```
POKE 3117,234:POKE 3118,234:X=USR(8309
)
```

Note that the DOS utilities did not have to reload from disk. Since no deferred mode lines of BASIC were entered and no program was loaded, DUP.SYS was unaltered and still in RAM. We simply jumped back into DUP.SYS with that immediate mode USR command.

In order to make our patch to DOS permanent, use the "H" function to *write DOS files.* □

ATARI®

600XL	CALL
1200XL	CALL *

*Reflects $100 Atari Rebate

1050 DRIVE	$335	MICROBITS INFC	$78
1025 PRINTER	$399	80 COLUMN BD	$249
1020 COLOR PTR	$219	TECHNICAL NOTES	$25
1027 PRINTER	$279	REAL TIME CLK	$38
1010 RECORDER	$72	AXIOM PRINTER INFC	$80
MICRO PERIF. MODEM/INFC	$129		

DRIVES

ASTRA 1620	$469
RANA 1000	$295
PERCOM 88-S1	$298
PERCOM 40-S1	$418
PERCOM 40-S2	$718
PERCOM 44-S1	$510
PERCOM 44-S2	$929

MEMORIES

48K RAM (INTEC)	$95
64K RAM (INTEC)	$119
48K RAM (MOSAIC)	$109
64K RAM (MOSAIC)	$145
128K RAM DISK	$299
32K RAM (MOSAIC)	$68

ATARI SOFTWARE

ADVENTURE INT'L
Adv. 1-12 each (C) $18
Preppie (C/D) $20
Preppie II (C/D) $23
Diskey (D) $33
Sea Dragon (C/D) $23
Bug Off! (C/D) $20
Tutti-Frutti (C/D) $17
Saga 1-3 (ea.) (D) $27
Lunar Lndr (C) $11 (D) $15
Galactic Empire (C) $14
Galactic Trader (C) $14
Stratos (C/D) $23
Rear Guard (C) $14 (D) $17

APX
Eastern Front (C/D) $23
747 Land Sim. (C/D) $17
Fig-Forth (C) $30
Family Cash Flow (D) $17
Downhill (C/D) $17
Avalanche (C/D) $17
Outlaw/Howitzer (C/D) $17
Salmon Run (C/D) $17
Data Management (D) $17
Galahad/Holy Grail (D) $21
Adv. Music System (D) $21

ATARI INC.
Microsoft Basic II (R) $62
Mickey in Great Outdoors (C/D) $36
Paint (D) $30
Speed Reading (C) $54
Qix (R) $30
Dig Dug (R) $30
Atari Writer (R) $68
Time Wise (D) $23
Visicalc (D) $139
Juggles House (C/D) $22
Juggles Rnbw (C/D) $22
Pilot (Home) (R) $55
Galaxian (R) $30
Defender (R) $30
ET $34
Microsoft Basic (D) $62
Macro Ass. & Edit (D) $62
Assembler Editor (R) $42
Basic Cartridge (R) $45
Pac Man (R) $30
Centipede (R) $30
Caverns of Mars (D) $28
Star Raiders (R) $30
Conv. Lang. Ea. (C) $42
Music Composer (R) $31
Super Breakout (R) $26
My First Alphabet (D) $26
Prog.2 & E (ea.)(C) $21
Word Processor (D) $102
Pilot (Educ.) (R) $92
Touch Typing (C) $19
Home File Mngr (D) $36
Bookkeeper (D) $102
Family Finance (D) $35
Prog. 1 (C) $18
Scram (C) $18
Asteroids (R) $27
Space Invaders (R) $27
Missile Command (R) $27
Telelink (R) $21
Superman III (R) $34
Basketball (R) $24
Donkey Kong (R) $34

AVALON HILL
VC (D) $17
B-1 Nuc. Bomber (C) $12
Legionnaire (C) $23
Empire of Overmind (D) $23 (C) $20
Tanktics (D) $20 (C) $17
Comptr Stock & Bonds (D) $17 (C) $14
Dnieper River Line (D) $20
Voyager (D) $17
Controller (D) $20
GFS Sorceress (D) $23 (C) $20
Telegard (D) $19 (C) $16
Vorrak (D) $18
Lord of Karma (C) $14

AVALON HILL(Cont'd)
Moon Patrol (C) $17
Planet Miners (C) $12
Conflict 2500 (C) $12
Space Station Zulu (D) $17 (C) $14
Flying Ace (D) $21 (C) $18
Gypsy (D) $18 (C) $15
Galaxy (D) $17 (C) $14
Guns of Fort Def. (C) $14
Nuke War (C) $12
Andromeda Conquest (D) $16 (C) $13
Close Assit (D) $23 (C) $20

BRODERBUND
Sky Blazer (D) $22
Bank St. Writer (D) $46
A.E. (D) $23
Arcade Machine (D) $39
Choplifter (D) $23 (R) $29
Labyrinth (C/D) $20
Serpentine (D) $23 (R) $27
Steller Shuttle (C/D) $17
Apple Panic (C/D) $20
Genetic Drift (C/D) $20
David's Midnight (D) $23
Sea Fox (D) $20
Track Attack (D) $20
Operatn Whrlwnd (D) $23
Match Boxes (C/D) $20

CDY CONSULTING
Pogoman (C/D) $27
Mad-Netter (C/D) $23
It-Is-Balloon (C/D) $23

CBS
Mountain King (R) $27
Boulders & Bombs (R) $27
Krazy (each) (R) $27

CONTINENTAL SOFT.
Home Accountant (D) $48
Tax Advantage (D) $39

DATA MOST
Pig Pen (D) $20
Mating Zone (D) $20
Roundabout (D) $20
Bilestoad (D) $20
Night Raiders (D) $20
Monster Smash (D) $20

DATASOFT
Text Wizard (D) $65
Graphic Master (D) $27
Micro Painter (D) $23
Lisp Interpreter (D) $79
Graphics Gen.(D) $17
Basic Compiler (D) $65
Zaxxon (C/D) $27
Teletalk (D) $34
Pac. Coast Hwy (C/D) $20
Clowns/Balloons (C/D) $20
Spell Wizard (D) $53
Canyon Climber (C/D) $20
Sands of Egypt (D) $27
O'Riley's Mine (C/D) $23
Rosen's Brigade (C/D) $23
Moon Shuttle (D) $27

DON'T ASK
Sam (D) $39
P.M. Animator (D) $23
Teletari (D) $27
Poker Sam (D) $17
Word Race (D) $17 (C) $14

EDU-WARE
Prisoner II (D) $27
Spelling Bee (D) $27
Rendevous (D) $27
Compu/Read (D) $21 (C) $15
Compu/Math-FR (D) $27 (C) $21
Compu/Math-Dec (D) $27 (C) $21

EDUCATIONAL SOFT.
Tricky Tutorial 1,2,3 or 4 (C/D) $15
Tricky Tutorial 5,6 or 7 (C/D) $22

Printers/Etc.

GEMINI 10X	$279	PROWRITER	$345
GORILLA	$199	SMITH TPI	$488

CITOH

Prowriter	$345
Prowriter II	$629
Starwriter	$1149
Printmaster	$1448

NEC

8023 A-C	$409
3510	$1375
3530	$1579
3550	$1779
7710/7730	$1998

SILVER REED P	$669
QUME 11/40 +	$1299

OKI-DATA

Microline 82A	CALL
Microline 83A	CALL
Microline 84P	CALL
Microline 92	CALL
Microline 93	CALL

DIABLO

620R	$939
630R	$1719

MONITORS

AMDEK

Color I	$289
V300	$139
V300A	$149
Color II	$449

NEC

GRN (JB1260)	$115
GRN (JB1201)	$155
Color Composite	$298
RGB Color	$598

MODEMS

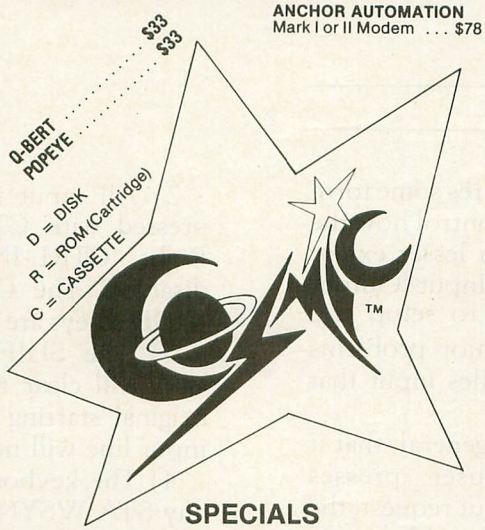
HAYES

Smartmodem	$209
Smartmodem 1200	$498
Micromodem II	$259

NOVATION

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

ANCHOR AUTOMATION
Mark I or II Modem $78



SPECIALS

Gemini 15X Printer	$399		
Axiom AT-100 Printer (with interface	$229		
Astra Double Density Dual Drive	$469		
Rana 1000 Drive	$319		
Bit-3 80 Column Board	$245		
Mannesmann Talley 160L Printer	$589		
Atari 400 Keyboard (In Home)	$35		
Programmer Kit	$48	Entertainer Kit	$64
Wico Joystick	$23	Wico Trackball	$49

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

(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.

*ASTRA 1620
DISK DRIVE SYSTEM

MORE FOR YOUR MONEY
DOUBLE OR SINGLE DENSITY
TWO DRIVES

SPECIAL $469

EPYX-AUTO. SIMULATION
Rescue at Rigel (C/D) $20
Datestones/Ryn (C/D) $14
Upr Rchs Apshai (C/D) $14
Ricochet (C/D) $14
Keys/Acheron (C/D) $14
Curse of Ra (C/D) $14
Danger/Drindisti (C/D) $14
Jumpman (C/D) $27
Escp Vulcans Isle (D) $20
Crypt of Undead (D) $20
Nightmare (D) $20
Armor Assault (D) $27
Monster Maze (R) $27
Alien Garden (R) $27
Plattermarna (R) $27
Morloc's Tower (C) $14
King Arthur's Heir (D) $20 (C) $14
Hellfire Warrior (C/D) $27
Invasion Orion (C/D) $17
Temple of Aps. (C/D) $27
Star Warrior (C/D) $27
Dragon's Eye (D) $20
Crush Crumble (C/D) $20

GEBELLI
Firebird (R) $27
Embargo (R) $30
Candy Factory (D) $20
Match Racers (C/D) $20
Dr. Goodcode (D) $20

HAYDEN
Go (D) $23 (C) $20
Sargon II (D) $23 (C) $20
Bulldog Pinball (C) $20

HES
Coco (C/D) $34
Coco II (C/D) $27

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

INHOME
Baseball (R) $23 (D) $23
Crypts of Terror (D) $23 (C) $20

INNOVATIVE DESIGN
Pool 1.5 (D) $23
Juggler (D) $20
Speedway Blast (D) $20 (R) $27
Pool 400 (R) $27

JV SOFTWARE
Jrny to Plnts (C/D) $20
Action Quest (C/D) $20
Ghost Encount. (C/D) $20

LJK
Letter Perfect (D) $104
Data Perfect (D) $74
Letter Perfect (R) $137
Edit 6502 (R) $137

MONARCH DATA SYS.
ABC Compiler (D) $48

ODESTA
Chess (D) $45
Checkers (D) $34
Odin (D) $34

OPTIMIZED SYSTEMS
C-65 (D) $58
Bug-65 (D) $23
Max-65 (D) $58
Basic A + (D) $58
Action (D) $65
Speed Read Plus (D) $40

PHOENIX
Birth of Phoenix (D) $16
Adv. In Time (D) $20

QUALITY
Fastgammon (C) $14
Name That Song (D) $13 (C) $11
Starbase Hyperion (D) $17 (C) $14
Ali Baba/40 Thvs (D) $20
Jeeper Creepers (D) $20

ROKLAN
Gorf (D) $27 (R) $30
Wizard/Wor (D) $27 (R) $30
Dlx Invaders (D) $23 (R) $27
Anti Sub Ptrl (D) $20 (C) $14
Telecom (D) $49
Space Journey (R) $30

SENTIENT
Gold Rush (D) $23
Cyborg (D) $23

SIERRA ON-LINE
Mission Asteroids (D) $17
Ulys.& Gldn Fleece (D) $27
Crossfire (D) $20 (R) $23
Mouseattack (D) $23
Jawbreaker (D) $20 (R) $23
Threshold (D) $27
Softporn (D) $20
Ultima II (D) $39

SIERRA ON-LINE
Marauder (D) $23
Lunar Leeper (D) $20
Wiz & Princess (D) $22
Frogger (C/D) $23

SIRIUS
Alpha Shield (R) $27
Wavy Navy (D) $23
Bandits (D) $23
Space Eggs (D) $20
Sneakers (D) $20
Way Out (D) $27
Type Attack (D) $27
Repton (D) $27
Critical Mass (D) $27
Fast Eddy (R) $23
Worm War (R) $23

SPINNAKER
Snooper Troop 1,2 (D) $30
Kindercomp (D) $20
Rhymes & Riddles (D) $20
Hey Diddle Diddle (D) $20
Srch Amzng Thngs (D) $27
Story Machine (D) $23
Face Maker (D) $23

STRATEGIC SIM.
Cosmic Balance (D) $27
Cosmic Balance II (D) $27
Tigers In Snow (C/D) $27
Battle of Shiloh (C/D) $27
Battle of Norm. (C/D) $27
Cytron Masters (D) $27
Knights/Desert (C/D) $27
Combat Leader (C/D) $27

SUBLOGIC
Pinball (C/D) $20

SWIFTY
J. White Music Lessons (C/D) $20
Space Shuttle (D) $20

SYNAPSE SOFTWARE
File Mngr 800 + $65
Protector II (D) $23 (R) $29
Shamus (D) $23 (R) $29
Fort Apocalypse (C/D) $23
Shamus II (C/D) $23
Necromancer (C/D) $23
Pharoh's Curse (C/D) $23
Slime (C/D) $23
Page 6 (D) $23
Reptilian (C/D) $23
Picnic Paranoia (C/D) $23
Claim Jumper (C/D) $23
Drelbs (C/D) $23
Shadow World (C/D) $23
Survivor (C/D) $23
T.N.T. (C/D) $23
N.Y.C. (C/D) $23
Slamball (C/D) $23
Blue Max (C/D) $23
Nautilus (C/D) $23
Chicken (R/D) $23
Quasimodo (C/D) $23
River Quest (C/D) $23

SYNERGISTIC
Programr Wkshp (D) $23
Graph Workshop (D) $27
Disk Workshop (D) $23
Worlock's Revenge (D) $23

THORN EMI
Soccer (R) $34
Jumbo Jet (R) $34
Submarine Comm. (R) $34
Kick Back (R) $34
Pool (C) $22
Humpty Dumpty (C) $22
Hckry Dckry Dck (C) $22
Darts (C) $22

USA
Atari World (D) $39
3-D Sprgrphcs (C/D) $27
Survival Adv. (C/D) $17

VERSA
Mind Bgglrs I (D) $14 (C) $12
Globe Master (D) $20

MISCELLANEOUS
Financial Wizard (D) $41
Castle Wolfenstein (D) $20
Master Type (D) $27
Astro Chase (D) $22
Miner 2049er (R) $34
Cypher Bowel (C) $33
Bug Attack (C/D) $20
Raster Blaster (D) $20
Air Strike (C/D) $27
Prism (D) $18
Megalegs (C) $23
Cap'n Cosmo (D) $19
Spy's Demise (D) $14
Galac. Chase (D) $20 (C) $18
Kid Grid (C/D) $20
Battle Trek (D) $20
Snapper (D) $19
T&F-PMP Property Management (D) $149
Millionaire (D) $39

AN ALTERNATIVE KEYBOARD HANDLER

16K Cassette or Disk

by Stephen Howard

Almost every type of software requires some form of keyboard input from the user to control how the program functions. BASIC, and to a lesser extent machine language (via the Central Input/Output routines) make this relatively easy to setup and process. However, there are two major problems with how the operating system handles input that can wreak havoc with the display.

First of all, the input routine is so general, that it will accept anything. If the user presses SHIFT/CLEAR in response to an input request, the entire screen will be cleared. Similar problems can arise with the use of the SHIFT/INSERT and SHIFT/DELETE keys. There is no simple way to discriminate against all such destructive key presses, and still retain the powerful features of the screen editor.

Secondly, there is the problem with display list interrupts. Whenever a key is pressed in response to an active request for input, the operating system's keyboard handler generates a click on the console speaker. The routine used to generate this click (located at $FCD8) uses a series of STA WSYNC instructions for accurate timing of the click frequency. This throws off the careful timing of a display list interrupt. The result: the scan line on which the display list interrupt is supposed to occur, jiggles back and forth. Unfortunatelyt, there is no memory location you can POKE or any other easy way to disable the click.

The alternative.

The only effective way to overcome both of these problems is to use a different keyboard handler. The one presented here incorporates the following features:

1) A maximum of one physical line of input, starting from the cursor location at the time this routine is called, up to screen column 38 or the right margin setting, whichever is less.

2) All input is treated as string data. All keys pressed with CTRL, except Cursor Left, Cursor Right, CTRL-INSERT and CTRL-DELETE are disabled. The CAPS, TAB, ESC, and INVERSE VIDEO keys are also disabled. Shift lock is in effect.

3) The SHIFT/CLEAR and SHIFT/DELETE keys will clear the input line, but only up to the original starting column. Any text preceeding the input line will not be erased.

4) The keyboard click routine does not employ any STA WSYNC instructions, and therefore does not interfere with display list interrupts. The routine may be omitted if no click is desired.

5) The handler routine is written in relocatable machine language, and may be easily called from BASIC using the USR function.

6) This handler does not replace the operating system's keyboard handler, and no entries are made in the Handler Address Table (HATABS), so the normal keyboard handler is always available if desired.

Using the handler from machine language.

Type in **Listing 1** and assemble it with your main-line program. Subroutines to disable and re-enable the BREAK key are included at the end of the program, and should only be typed in if needed. If the keyboard click is not desired, simply omit lines 1500-1580. If you want the original operating system's click, replace lines 1500 to 1580 with JSR $FCD8. To call the handler for input, do the following:

1. Ensure that ROWCRS ($54) and COLCRS ($55) are set to the appropriate row and column you want the input to start from. If you used the Central Input/Output (CIO) utility to print to the screen, these will already be set.

2. Ensure that RMARGN ($53) is set to the desired right margin value.

3. Do a JSR to the keyboard handler's starting

address. The routine will return with the results of the input, followed by an End-of-Line ($9B), stored in LBUFF (starting at $580). The accumulator, X and Y registers will be altered by the handler during this process.

Using the handler from BASIC.

The handler used for BASIC is a modified version of the one used in machine language. The only difference is, the BASIC version returns the input result to a string variable. The Assembly language listing of this version appears in **Listing 2**.

To use the handler from BASIC, type in **Listing 3**, lines 100 to 390. This sets up the handler for use via the USR function call. The rest of **Listing 3** contains a test program and demonstrates how to use the handler, with its own click, no click, and the operating system's click routine by replacing a small portion of KEYBD$. Extra measures must be taken to use the handler in split screen modes, and an example of this is also included in the test program (see **Precautions**).

DISBRK$ is a short machine language routine that disables the BREAK key. This is necessary because if the user presses the BREAK key during the input request, strange things start happening. ENBRK$ re-enables the BREAK key.

To call the handler for a string input, the format is:

```
X=USR(ADR(KEYBD$),ADR(svar))
```

where svar is any legal string variable name which will contian the results of the input. Be sure that svar is dimensioned before calling the handler. Control will be returned to the program after an input has been received. To disable the BREAK key, use:

```
X=USR(ADR(DISBRK$))
```

This statement should appear before the first call to the keyboard handler and is only necessary once, unless a new GRAPHICS statement is executed. See **Precautions** for more detail. To re-enable the BREAK key, use:

```
X=USR(ADR(ENBRK$))
```

Try running the full test program, which contains a simple display list interrupt. Note how the border between colors in the center of the screen flutters when the operating system's click routine is used.

How it works.

In discussing how any keyboard handler operates, it is necessary to be familiar with the three different types of code used to represent characters:

Keycode — Used to represent which key has been pressed. In this code, if Bit 7 = 1, then the CTRL key is being pressed simultaneously with the key identified by the lower order bits. Similarly, if Bit 6 = 1, then the SHIFT key is being pressed.

Internal — Used to display a character on the screen. This is also the sequential number of the character in its character set.

ATASCII — The ATARI version of the ASCII standard code representation for characters.

The Alternative Keyboard Handler routine starts with some initial setup and then branches to a Get Character subroutine, where it waits for a key press by monitoring the Keyboard Code Operating System shadow — CH ($2FC). This register will contain the value 255 ($FF) if no key has been pressed. Once a key press is detected, a click is generated, and the key press is checked to see if the CTRL key is also being pressed. If this is the case, then it is checked to see if it is one of the allowed control character exceptions — CURSOR RIGHT, CURSOR LEFT, CTRL-INSERT, or CTRL-DELETE. If it is not an exception, no character is printed and the system goes back and waits for another key press.

Next, the key press is checked against a table of undesirable key presses — CAPS, ESC, TAB, etc. If it is not one of these, it is converted to ATASCII, using a ROM-based table — ATASCI (located at $FEFE). If the key is an undesirable key press, then again, no character is printed and the system goes back and waits for another. The ATASCII letter is now checked to see if it is a letter between A and Z and, if so, it is changed to upper case. Before finally being printed, the character is tested one last time to

(continued on page 99)

DISKEY

by Sparky Starks

Potential into practical reality. This is the core of DISKEY — a remarkable utility program that gives ATARI disk drive owners the flexibility to accomplish tasks that other utilities either ignored or only hinted at. With DISKEY, the user will be able to actually examine a disk and its directory, and repair some files that might otherwise have been lost. DISKEY also performs a multitude of other practical functions, including the following:

- ★ Automatically lists any unreadable or destroyed sectors
- ★ Sends contents of disk to printer selectively
- ★ Up to four separate drives may be addressed
- ★ Can be used to backup some of those "unbackupable" disks
- ★ Allows direct examination of any file
- ★ Over 50 separate key functions available

As an ATARI 400 or 800 owner, you have extraordinary power and versatility at your disposal. Tap into those resources effectively with DISKEY — new from Adventure International.

ATARI 400/800 32K DISK
052-0158 $49.95
(Note: Some features will require 40K)

SOME OF DISKEY KEY FUNCTIONS

A	Clear screen and filename
B	Byte compare, D1 to D2, OS to DS
C	Copy sectors, OD to DD, OS to DS
D	Toggle destination drive
E	Erase disk (format)
F	Select file sub-menu
L	Set automatic function lower limit (OS)
M	Modify Sector Map
N	New destination sector
O	Toggle originate drive
P	Print screen to printer
Q	Query (search for hex key, drive OD, sector OS to DS)
R	Read new OS, set DS to match
S	Search for ASCII key, drive OD, sector OS to DS
T	Tape to disk
U	Upper case conversion of printer lower case
V	Toggle write verify
W	Write memory buffer to sector DS, drive DD
X	Select EOR Sector Map screen print mask
Z	Zero memory buffer
+	Read upward, next sector on disk
–	Read downward
?	Directory information
!	Select directory sub-menu
cB	Byte compare, D1 to D2, whole disk
cC	Copy D1 to D2, whole disk
cD	Decimal to hex, ASCII conversion
cE	Erase disk (without new format)
cF	Modify sector forward sector chain reference
cH	Hex to decimal, ASCII conversion
cL	Locate bad sector on drive OD
cN	Modify sector file number reference
cO	Select one-drive functions sub-program
cP	Print current Disk Map
cR	RPM test drive OD
cS	Special file copy, no directory reference from source
cV	VTOC update and repair, drive OD
cY	Toggle Sure Response prompt enable
FA	File binary load address headers to printer
FD	Delete file
FF	Select filename for all file functions
FL	Lock file
FM	Show memory address load position in file
FQ	Relative Query
FR	Rename file
FS	Relative Search

BASIC ROUTINES FOR THE ATARI

by Jerry White

Finally, a self-help system that cuts through the stuffy technical jargon and allows the user to learn effective programming techniques fast! BASIC ROUTINES FOR THE ATARI has been written especially for ATARI 400 or 800 users who wish to write programs in BASIC. This package comes complete with an extensive manual and your choice of a tape or disk which contains all of the routines from the manual — which means you'll be able to actually see each of them in action on your ATARI. Some of the BASIC routines include joystick, sound, player missile strings and much more — and presented in a way that you'll quickly be able to learn and apply to your own programs.

If your programming ability lies somewhere between beginning and advanced, then look no further — BASIC ROUTINES FOR THE ATARI is the system for you.

BASIC ROUTINES book with 16K TAPE
051-0154 $24.95
BASIC ROUTINES book with 24K DISK
052-0154 $24.95

Published by Adventure INTERNATIONAL
a subsidiary of Scott Adams, Inc.
BOX 3435 • LONGWOOD, FL 32750 • (305) 830-8194

To order, see your local dealer. If he does not have the program, then call 1 (800) 327-7172 (orders only please) or write for our free catalog.

see if it is a special case — SHIFT/DELETE, SHIFT/CLEAR, CTRL-INSERT, CURSOR RIGHT, or RETURN. If the character is one of these, the handler executes a specialized routine. If not, it passes the character to CIO, which converts it to internal code and displays it in the proper place on the screen. Unless the key press was a RETURN, the routine goes back and waits for another key press.

When RETURN is pressed, the contents of the screen input are copied to the input buffer and execution returns to where the handler was called from. The BASIC version returns the input to a string variable and updates the length of that string variable before returning.

Precautions.

Care should be taken not to use memory locations $74-$77 in either non-maskable interrupt (NMI) routines (display list interrupts and vertical blank interrupts) or interrupt requests (IRQ), or else unusual events may occur. The handler routine saves the contents of these memory locations before using them, and restores them after it's done, so they can be used at any other time. However, since NMIs and IRQs will interrupt the handler during its execution, those routines should not use these memory locations.

On occasion, when a key is pressed and display list interrupts are being used, the screen may flash momentarily. This will happen when a keyboard IRQ is still being processed at the time a vertical blank interrupt (VBI) occurs. The 1st stage VBI does not include copying the playfield color shadow registers to their respective hardware registers. If an IRQ was in progress (as indicated by the I bit of the Processor Status register being set), the VBI will assume it interrupted "time critical" code, and will only execute stage 1 before returning. Thus, when the next screen is drawn, the colors before the DLI occurs will be wrong, and the screen will appear to flash. To correct this, write an immediate VBI routine to copy the appropriate shadow registers. Since an immediate VBI is executed before stage 1 of the operating system's VBI, this will eliminate any screen flashing problems.

If CTRL-1 is pressed, subsequent key presses will not appear until CTRL-1 is pressed again. This will not disrupt the display or input in any other way and it is relatively unlikely that this key press would ever be entered anyway.

More precautions.

The following notes refer to the BASIC version of the handler only.

SAVE the program before running it. One typographical error in the machine language portions can result in a system crash and complete loss of everything you typed in.

Memory locations $6EF-$6F2 (1775-1778) are used for temporary storage and may be used when the handler is not being used. The values stored there, however, are not saved and restored when the handler is called. Memory locations $6F3-$6FF (1779-1791) contain the tables of allowed control characters and undesirable key presses, and must not be tampered with at any time.

Ensure that all string variables used for input with the handler are dimensioned as early in the program as possible. The handler routine will only search the Variable Value Table for the first 32 variables (string or numerical) defined in a program before giving up and assuming no match will be found. This will result in not updating the length of the string.

Disabling the BREAK key is required every time a new GRAPHICS statement is executed.

Using the alternative keyboard handler in split screen modes poses a particular problem, because the operating system uses a different set of memory locations for cursor and margin control. Lines 760 to 980 of **Listing 3** demonstrate a solution to this problem. A subroutine is used to swap the two sets of values. The memory location used by the operating system to determine if split screen is in effect (BOTSCR — $2BF) is then changed to fool the system into thinking that split screen mode is not being used. A value of 24 in this register means 24 lines per screen (a full screen), and a value of 4 means 4 lines per screen, or split screen mode. □

Listing 1.

```
0100 ; ***********************************
0110 ;  *  ALTERNATIVE KEYBOARD HANDLER  *
0120 ; ***********************************
0130 LMARGN = $52      ; Text column left margin
0140 RMARGN = $53      ; Text column right margin value
0150 COLCRS = $55      ; Column position of cursor
0160 OLDCHR = $5D      ; Character under cursor restore
0170 OLDADR = $5E      ; Memory address of cursor
0180 MEMST = $74       ; Memory address where input starts
0190 MAXLEN = $76      ; Maximum line length
0200 CLKCTR = $77      ; Click routine counter
0210 CRSINH = $02F0    ; Cursor display inhibit
0220 CH  =    $02FC    ; Keyboard character code
0230 PUTCH = $0B       ; CIO command for PUT CHARACTERS
0240 ICCOM = $0342     ; CIO command byte
0250 ICBLL = $0348     ; CIO buffer length (low)
0260 ICBLH = $0349     ; CIO buffer length (high)
0270 CIOV =  $E456     ; CIO entry point
0280 LBUFF = $0580     ; Input result buffer
0290 CONSOL = $D01F    ; Console switch port
0300 INTATA = $FEFA    ; Internal code to ATASCII table
0310 ATASCI = $FEFE    ; Keycode to ATASCII table
0320       *=  $5000
0330 ;
0340 KEYBD LDX #3
0350 KYB0 LDA MEMST,X ; Save $74 - $77
0360       PHA
0370       DEX
0380       BPL KYB0
0390       LDA RMARGN  ; Save RMARGN
0400       PHA
0410       CMP #$27    ; If RMARGN = 39, don't increment it
0420       BEQ NOINC
0430       INC RMARGN  ; Add 1 to RMARGN so cursor can
0440 ;                   temporarily go there
0450 NOINC LDA LMARGN ; Save LMARGN
0460       PHA
0470       LDA CRSINH  ; Save CRSINH
0480       PHA
0490       LDA COLCRS  ; Set LMARGN to current
0500       STA LMARGN  ; cursor location
0510       LDA RMARGN
0520       CLC         ; Subtract (LMARGN+1) to
0530       SBC LMARGN  ; compute and save maximum line
0540       STA MAXLEN  ; length for use as an index
0550       LDA OLDADR+1 ; Save starting memory
0560       STA MEMST+1 ; address of input
0570       LDA OLDADR
0580       STA MEMST
0590       LDA #0      ; Ensure cursor on
0600       STA CRSINH
0610       STA ICBLL   ; Set up CIO to PUT CHARACTERS
0620       STA ICBLH   ; using single byte passed
0630       LDA #PUTCH  ; through the accumulator mode
0640       STA ICCOM
0650       BPL GTCHXF  ; Get an ATASCII character input
0660 KYB1  CMP #$9C    ; Shift/delete?
0670       BEQ CLRLIN
0680       CMP #$7D    ; Shift/clear?
0690       BEQ CLRLIN
0700       CMP #$FF    ; Insert character?
0710       BEQ CTLINS
0720       CMP #$1F    ; Cursor right?
0730       BEQ CRSRT
0740       CMP #$9B    ; If EOL, exit
```

```
0750      BEQ EXIT
0760 KYB2 LDX #0      ; IOCB index = 0 (Screen editor)
0770      JSR CIOV    ; Put character on screen
0780      LDA RMARGN  ; If cursor position is beyond right
0790      CMP COLCRS  ; margin, move cursor left
0800      BNE GETCH   ; Otherwise, get another character
0810      LDA #$1E    ; ATASCII cursor left
0820      BPL KYB2    ; JMP KYB2
0830 CLRLIN LDY MAXLEN
0840      LDA #0
0850 CL1  STA (MEMST),Y ; Clear line on screen
0860      DEY
0870      BPL CL1
0880      STA OLDCHR  ; Clear OLDCHR, or CIO will restore
0890      LDA RMARGN  ; it when cursor is moved
0900      STA COLCRS  ; Reset cursor to beginning by
0910      LDA #$1F    ; setting cursor at right edge, then
0920      BPL KYB2    ; executing a cursor right
0930 CTLINS LDY MAXLEN ; Is there a character
0940      LDA (MEMST),Y ; at the right margin?
0950      BNE GETCH   ; Yes, then don't do this
0960      LDA #$FF    ; No, do CTRL-INSERT
0970      BMI KYB2
0980 CRSRT LDX #0     ; Do cursor right
0990      JSR CIOV
1000      LDA COLCRS  ; Did cursor end up
1010      CMP RMARGN  ; past right margin?
1020      BNE GETCH   ; No, so get another character
1030      LDA #$1F    ; Yes, then reset cursor by doing
1040      BPL CRSRT   ; an additional cursor right
1050 GTCHXF BPL GETCH ; JMP GETCH (transfer)
1060 EXIT LDY MAXLEN  ; Get characters from screen
1070 EX1  LDA (MEMST),Y
1080      PHA         ; Save it temporarily
1090      ROL A       ; Convert from internal to ATASCII
1100      ROL A
1110      ROL A
1120      ROL A
1130      AND #3
1140      TAX
1150      PLA
1160      AND #$9F
1170      ORA INTATA,X
1180      AND #$7F    ; Mask off bit 7 in case cursor was
1190      STA LBUFF,Y ; on top and store in buffer
1200      DEY
1210      BPL EX1
1220      LDY MAXLEN
1230 EX2  LDA LBUFF,Y ; Put carriage return
1240      CMP #$20    ; after first non-blank
1250      BEQ EX3     ; character encountered from
1260      LDA #$9B    ; the right
1270      STA LBUFF+1,Y
1280      BMI RETURN  ; JMP RETURN
1290 KYB1XF BCS KYB1  ; JMP KYB1 (transfer)
1300 EX3  DEY
1310      BPL EX2
1320 RETURN PLA       ; Restore CRSINH
1330      STA CRSINH
1340      PLA         ; Restore LMARGN
1350      STA LMARGN
1360      PLA         ; Restore RMARGN
1370      STA RMARGN
1380      LDX #0
1390 RT1  PLA         ; Restore $74 - $77
1400      STA MEMST,X
1410      INX
1420      CPX #4
1430      BNE RT1
1440      RTS         ; All done, return
1450 GETCH LDA CH
1460      CMP #$FF    ; Key pressed?
1470      BEQ GETCH   ; No, keep trying
1480      LDX #$FF    ; Yes, clear CH
1490      STX CH
1500 CLICK LDY #$7F   ; Keyboard click routine
1510      STY CLKCTR
1520 CK1  LDY CLKCTR
1530      STY CONSOL  ; Turn speaker on/off
1540      LDX #8      ; This number directly controls
1550 DELAY DEX        ; the click frequency
1560      BPL DELAY   ; Wait a while
1570      DEC CLKCTR  ; Then do it again
1580      BPL CK1     ; Only do it 128 times
1590      CMP #$80    ; CTRL pressed?
1600      BCC B2      ; No, then branch
1610      LDX #3      ; Check for exceptions to 'No
1620 B1   CMP TABLE,X ; control characters'
1630      BEQ B2      ; Branch if allowed exception
1640      DEX
1650      BPL B1
1660      BMI GETCH   ; Not allowed, so get another
1670 B2   LDX #8      ; Check for other characters
1680 B3   CMP TABLE1,X ; that are not allowed
1690      BEQ GETCH   ; No good, go get another
1700      DEX
1710      BPL B3
1720      TAX         ; Allowed, so get ATASCII equivalent
1730      LDA ATASCI,X
1740      CMP #$61    ; If its a lower case letter
1750      BCC B4      ; make it an upper case
1760      CMP #$7B
1770      BCS B4
1780      SEC
1790      SBC #$20
1800 B4   SEC
1810      BCS KYB1XF  ; All done, return
1820 ;
1830 ;Control character exceptions: CURSOR LEFT,
1840 ;CURSOR RIGHT, CTRL-INSERT, CTRL-DELETE
1850 TABLE .BYTE $86,$87,$B4,$B7
1860 ;Non-control character undesirables:
1870 ;TAB,SHF/TAB,ESC,SHF/ESC,ATARI,SHF/ATARI
1880 ;SHF/INS,CAPS,SHF/CAPS
1890 TABLE1 .BYTE $2C,$6C,$1C,$5C,$27,$67,$77,$3C,$7C
1900 ;
1910 ; DISABLE BREAK KEY Subroutine
1920 ;
1930 POKMSK = $10     ; Pokey Mask (IRQEN shadow)
1940 IRQEN = $D20E    ; IRQ Enable
1950 DISBRK SEI       ; Disable IRQs while messing with IRQEN
1960      LDA POKMSK
1970      AND #$7F    ; Mask off BREAK key IRQ bit
1980      STA POKMSK  ; Store back in shadow
1990      STA IRQEN   ; Store in hardware register
2000      CLI         ; Re-enable IRQs
2010      RTS
2020 ;
2030 ; ENABLE BREAK KEY Subroutine
2040 ;
2050 ENBRK SEI
2060      LDA POKMSK
2070      ORA #$80
2080      STA POKMSK
2090      STA IRQEN
2100      CLI
2110      RTS
```

Listing 2.

```
0100 ; ***********************************
0110 ; *  ALTERNATIVE KEYBOARD HANDLER   *
0120 ; *        (Basic Version)          *
0130 ; ***********************************
0140 OFFST = $06EF    ; Offset from  String/Array area[2]
0150 TEMP  =  $06F1   ; Temporary storage[2]
0160 TABLE = $06F3    ; Table of allowed CTRL characters[4]
0170 TABLE1 = $06F7   ; Table of ignored key presses[8]
0180 LMARGN = $52     ; Text column left margin
0190 RMARGN = $53     ; Text column right margin value
0200 COLCRS = $55     ; Column position of cursor
0210 OLDCHR = $5D     ; Character under cursor restore
0220 OLDADR = $5E     ; Memory address of cursor[2]
0230 MEMST = $74      ; Memory address where input starts[2]
0240 STOVAR = MEMST   ; String variable address[2]
0250 MAXLEN = $76     ; Maximum line length
0260 CLKCTR = $77     ; Click routine counter
0270 VVTP =  $86      ; Variable Value Table Pointer
0280 STARP = $8C      ; String/Array area Pointer
0290 CRSINH = $02F0   ; Cursor display inhibit flag
0300 CH   =  $02FC    ; Keyboard char code
0310 LBUFF = $0580    ; Input return buffer
0320 PUTCH = $0B      ; CIO command for PUT CHARACTERS
0330 ICCOM = $0342    ; CIO command byte
0340 ICBLL = $0348    ; CIO buffer length (low)
0350 ICBLH = $0349    ; CIO buffer length (high)
0360 CIOV =  $E456    ; CIO entry point
0370 CONSOL = $D01F   ; Console switch port
0380 INTATA = $FEFA   ; Internal code to ATASCII table
0390 ATASCI = $FEFE   ; Keycode to ATASCII table
0400      *=  $5000
0410      PLA         ; Discard
0420      PLA         ; String variable address (high byte)
0430      STA TEMP+1  ; Save it
0440      PLA         ; String variable address (low byte)
0450      STA TEMP    ; Save it
0460      SEC
0470      SBC STARP   ; Compute address's offset
0480      STA OFFST   ; from the string/array pointer
0490      LDA TEMP+1  ; and save it for later
0500      SBC STARP+1
0510      STA OFFST+1
0520 KEYBD LDX #3
0530 KYB0 LDA MEMST,X ; Save $74 - $77
0540      PHA
0550      DEX
0560      BPL KYB0
0570      LDA RMARGN  ; Save RMARGN
0580      PHA
0590      CMP #39     ; If RMARGN = 39, don't increment it
0600      BEQ NOINC
0610      INC RMARGN  ; Add 1 to RMARGN so cursor can
0620 ;                temporarily go there
0630 NOINC LDA LMARGN ; Save LMARGN
0640      PHA
0650      LDA CRSINH  ; Save CRSINH
0660      PHA
0670      LDA COLCRS  ; Set LMARGN to current
0680      STA LMARGN  ; cursor location
0690      LDA RMARGN
0700      CLC         ; Subtract (LMARGN+1) to
0710      SBC LMARGN  ; compute and save maximum line
0720      STA MAXLEN  ; length for use as an index
0730      LDA OLDADR+1 ; Save starting memory
0740      STA MEMST+1 ; address of input
0750      LDA OLDADR
0760      STA MEMST
0770      LDA #0      ; Ensure cursor on
0780      STA CRSINH
0790      STA ICBLL   ; Set up CIO to PUT CHARACTERS
0800      STA ICBLH   ; using single byte passed
0810      LDA #PUTCH  ; through the accumulator mode
0820      STA ICCOM
0830      BNE GTCHXF  ; Get an ATASCII character input
0840 KYB1 CMP #$9C    ; Shift/delete?
0850      BEQ CLRLIN
0860      CMP #$7D    ; Shift/clear?
0870      BEQ CLRLIN
0880      CMP #$FF    ; Insert character?
0890      BEQ CTLINS
0900      CMP #$1F    ; Cursor right?
0910      BEQ CRSRT
0920      CMP #$9B    ; If EOL, exit
0930      BEQ EXIT
0940 KYB2 LDX #0      ; IOCB index = 0 (Screen editor)
0950      JSR CIOV    ; Put character on screen
0960      LDA RMARGN  ; If cursor position is beyond right
0970      CMP COLCRS  ; margin, move cursor left
0980      BNE GTCHXF  ; Otherwise, get another character
0990      LDA #$1E    ; ATASCII cursor left
1000      BPL KYB2    ; JMP KYB2
1010 CLRLIN LDY MAXLEN
1020      LDA #0
1030 CL1  STA (MEMST),Y ; Clear line on screen
1040      DEY
1050      BPL CL1
1060      STA OLDCHR  ; Clear OLDCHR, or CIO will restore
1070      LDA RMARGN  ; it when cursor is moved
1080      STA COLCRS  ; Reset cursor to beginning by
1090      LDA #$1F    ; setting cursor at right edge, then
1100      BPL KYB2    ; executing a cursor right
1110 CTLINS LDY MAXLEN ; Is there a character
1120      LDA (MEMST),Y ; at the right margin?
1130      BNE GTCHXF  ; Yes, then don't do this
1140      LDA #$FF    ; No, do CTRL-INSERT
1150      BMI KYB2
1160 CRSRT LDX #0     ; Do cursor right
1170      JSR CIOV
1180      LDA COLCRS  ; Did cursor end up
1190      CMP RMARGN  ; past right margin?
1200      BNE GTCHXF  ; No, so get another character
1210      LDA #$1F    ; Yes, then reset cursor by doing
1220      BPL CRSRT   ; an additional cursor right
1230 EXIT LDY MAXLEN  ; Get characters from screen
1240 EX1  LDA (MEMST),Y
1250      PHA         ; Save it temporarily
1260      ROL A       ; Convert from internal to ATASCII
1270      ROL A
1280      ROL A
1290      ROL A
1300      AND #3
1310      TAX
1320      PLA
1330      AND #$9F
1340      ORA INTATA,X
1350      AND #$7F    ; Mask off bit 7 in case cursor was
1360      STA LBUFF,Y ; on top and store in buffer
1370      DEY
1380      BPL EX1
1390      BMI COPY
1400 KYB1T1 BCS KYB1 ; JMP KYB1
1410 GTCHXF BNE GETCH ; JMP GETCH
1420 COPY LDA TEMP    ; Move string variable address
```

```
1430      STA STBVAR  ; from temporary storage
1440      LDA TEMP+1  ; to page zero for copy
1450      STA STBVAR+1
1460      LDY MAXLEN
1470 EX2  LDA LBUFF,Y ; Copy LBUFF to string variable
1480      STA (STBVAR),Y
1490      DEY
1500      BPL EX2
1510      LDY MAXLEN  ; Set length of string input
1520 EX3  LDA LBUFF,Y
1530      CMP #$20
1540      BNE EX4
1550      DEY
1560      BPL EX3
1570 EX4  INY
1580      TYA           ; Save it
1590      PHA
1600      LDX #0        ; Find location where BASIC keeps
1610      LDY #0        ; track of this string's current length
1620 EX5  LDA (VVTP),Y  ; Table entry 1 = $81?
1630      BPL EX6       ; No, branch - not a string variable
1640      INY
1650      INY
1660      LDA (VVTP),Y  ; Table entry 3 = low byte of
1670      CMP OFFST     ; string's address(by STARR offset)?
1680      BNE EX6       ; No, branch - this isn't it
1690      INY           ; Yes, this is probably it, but check
1700      LDA (VVTP),Y  ; high byte to be sure
1710      CMP OFFST+1
1720      BNE EX6       ; No match, lucky we checked - branch
1730      INY           ; A match! Just as we suspected
1740      PLA           ; Retrieve string input length
1750      STA (VVTP),Y  ; and store it
1760      BPL RETURN    ; Continue with exit routine
1770 EX6  INX           ; Set up to check next variable
1780      TXA           ; in Variable Value Table
1790      ASL A         ; Multiply variable number by 8
1800      ASL A         ; to get index (8 entries per variable)
1810      ASL A
1820      TAY           ; Set up as index and try again
1830      BNE EX5       ; Unconditional(essentially)
1840      PLA           ; Never found a match, so discard length
1850 ;       value and exit - this should never happen
1860 RETURN PLA         ; Restore CRSINH
1870      STA CRSINH
1880      PLA           ; Restore LMARGN
1890      STA LMARGN
1900      PLA           ; Restore RMARGN
1910      STA RMARGN
1920      LDX #0
1930 RT1  PLA           ; Restore $74 - $77
1940      STA MEMST,X
1950      INX
1960      CPX #4
1970      BNE RT1
1980      RTS           ; All done, return
1990 KYB1T2 BCS KYB1T1 ; JMP KYB1
2000 GETCH LDA CH
2010      CMP #$FF      ; Key pressed?
2020      BEQ GETCH     ; No, keep trying
2030      LDX #$FF      ; Yes, clear CH
2040      STX CH
2050 CLICK LDY #$7F     ; Keyboard click routine
2060      STY CLKCTR
2070 CK1  LDY CLKCTR
2080      STY CONSOL    ; Turn speaker on/off
2090      LDX #8        ; This number directly controls
2100 DELAY DEX          ; the click frequency
2110      BPL DELAY     ; Wait a while
2120      DEC CLKCTR    ; Then do it again
2130      BPL CK1       ; Only do it 128 times
2140      CMP #$80      ; CTRL pressed?
2150      BCC 82        ; No, then branch
2160      LDX #3        ; Check for exceptions to 'No
2170 81   CMP TABLE,X   ; control characters'
2180      BEQ 82        ; Branch if allowed exception
2190      DEX
2200      BPL 81
2210      BMI GETCH     ; Not allowed, so get another
2220 82   LDX #8        ; Check for other characters
2230 83   CMP TABLE1,X  ; that are not allowed
2240      BEQ GETCH     ; No good, go get another
2250      DEX
2260      BPL 83
2270      TAX           ; Allowed, so get ATASCII equivalent
2280      LDA ATASCI,X
2290      CMP #$61      ; If its a lower case letter
2300      BCC 84        ; make it an upper case
2310      CMP #$7B
2320      BCS 84
2330      SEC
2340      SBC #$20
2350 84   SEC
2360      BCS KYB1T2    ; All done, return
2370 ;
2380 ;Control character exceptions: CURSOR LEFT,
2390 ;CURSOR RIGHT, CTRL-INSERT, CTRL-DELETE
2400 ;
2410 ;Non-control character undesirables:
2420 ;TAB,SHF/TAB,ESC,SHF/ESC,ATARI,SHF/ATARI
2430 ;SHF/INS,CAPS,SHF/CAPS
```

Listing 3.

```
100 DIM KEYBD$(367),DISBRK$(13),ENBRK$
(13),A$(35)
110 FOR I=0 TO 12:READ BYTE:POKE 1779+
I,BYTE:NEXT I
120 FOR I=1 TO 13:READ BYTE:DISBRK$(I)
=CHR$(BYTE):NEXT I
130 FOR I=1 TO 13:READ BYTE:ENBRK$(I)=
CHR$(BYTE):NEXT I
140 FOR I=1 TO 367:READ BYTE:KEYBD$(I)
=CHR$(BYTE):NEXT I
150 REM * Legal control characters
160 DATA 134,135,180,183
170 REM * Illegal keycodes
180 DATA 44,108,28,92,39,103,119,60,12
4
190 REM * DISBRK$ DATA
200 DATA 104,120,165,16,41,127,133,16,
141,14,210,88,96
210 REM * ENBRK$ DATA
220 DATA 104,120,165,16,9,128,133,16,1
41,14,210,88,96
230 REM * KEYBD$ DATA
240 DATA 104,104,141,242,6,104,141,241
,6,56,229,140,141,239,6,173,242,6,229,
141,141,240,6,162
250 DATA 3,181,116,72,202,16,250,165,8
3,72,201,39,240,2,230,83,165,82,72,173
,240,2,72,165
260 DATA 85,133,82,165,83,24,229,82,13
3,118,165,95,133,117,165,94,133,116,16
9,0,141,240,2,141
270 DATA 72,3,141,73,3,169,11,141,66,3
,208,109,201,156,240,31,201,125,240,27
,201,255,240,42
280 DATA 201,31,240,48,201,155,240,59,
162,8,32,86,228,165,83,197,85,208,78,1
69,30,16,241,164
290 DATA 118,169,0,145,116,136,16,251,
133,93,165,83,133,85,169,31,16,222,164
,118,177,116,208,49
300 DATA 169,255,48,212,162,0,32,86,22
8,165,85,197,83,208,34,169,31,16,241,1
64,118,177,116,72
310 DATA 42,42,42,42,41,3,170,104,41,1
59,29,250,254,41,127,153,128,5,136,16,
232,48,4,176
320 DATA 147,208,98,173,241,6,133,116,
173,242,6,133,117,164,118,185,128,5,14
5,116,136,16,248,164
330 DATA 118,185,128,5,201,32,208,3,13
6,16,246,200,152,72,162,0,160,0,177,13
4,16,23,200,200
340 DATA 177,134,205,239,6,208,14,200,
177,134,205,240,6,208,6,200,104,145,13
4,16,9,232,138,10
350 DATA 10,10,168,208,221,104,104,141
,240,2,104,133,82,104,133,83,162,0,104
,149,116,232,224,4
360 DATA 208,248,96,176,154,173,252,2,
201,255,240,249,162,255,142,252,2,160,
127,132,119,164,119,140
370 DATA 31,208,162,8,202,16,253,198,1
19,16,242,201,128,144,12,162,3,221,243
,6,240,5,202,16
380 DATA 248,48,210,162,8,221,247,6,24
0,203,202,16,248,170,189,254,254,201,9
7,144,7,201,123,176
390 DATA 3,56,233,32,56,176,180
400 REM
410 REM * START OF DEMO PROGRAM
420 REM
430 GRAPHICS 0:SETCOLOR 1,12,0:SETCOLO
R 2,12,10:SETCOLOR 4,7,4
440 DLIST=PEEK(560)+256*PEEK(561)
450 REM Find the Display List
460 POKE DLIST+16,130
470 REM Set Display List Interrupt bit
 on mode line halfway down the screen
480 FOR I=0 TO 10:READ BYTE:POKE 1536+
I,BYTE:NEXT I
490 REM Poke in DLI routine
500 DATA 72,169,244,141,10,212,141,24,
208,104,64
510 REM Assembly Language equivalent:
520 REM PHA ; Save
530 REM LDA #$F4 ; Brown
540 REM STA WSYNC ; Wait for synch
550 REM STA COLPF2 ; Then store it
560 REM PLA ; Restore
570 REM RTI ; Return
580 POKE 512,0:POKE 513,6
590 REM Display List Interrupt routine
 starts at $600
600 POKE 54286,192
610 REM Enable DLI
620 PRINT "↓↓TEST INPUT? ";
630 X=USR(ADR(DISBRK$))
640 REM Disable BREAK key before first
 handler call
650 X=USR(ADR(KEYBD$),ADR(A$))
660 PRINT :PRINT "YOU TYPED : ";A$
670 PRINT "↓Now, no click!":PRINT "TES
T INPUT? ";
```

```
680 KEYBD$(306,323)="■■■■■■■■■■■■■■■■■■
■"
690 X=USR(ADR(KEYBD$),ADR(A$))
700 PRINT :PRINT "YOU TYPED : ";A$
710 PRINT "↓Now with the Operating Sys
tem click:":PRINT "TEST INPUT? ";
720 KEYBD$(306,323)=" ■■■■■■■■■■■■■■■■■
■"
730 X=USR(ADR(KEYBD$),ADR(A$))
740 PRINT :PRINT "YOU TYPED : ";A$
750 FOR I=1 TO 500:NEXT I
760 DIM T(12)
770 ROWCRS=84:TXTROW=656:BOTSCR=703
780 GRAPHICS 8:COLOR 1:PLOT 30,100:DRA
WTO 300,30
790 PRINT "TEST INPUT? ";:GOSUB 870
800 X=USR(ADR(DISBRK$))
810 REM Disable BREAK key after every
GRAPHICS statement
820 X=USR(ADR(KEYBD$),ADR(A$))
830 GOSUB 940
840 PRINT :PRINT "YOU TYPED :";A$
850 X=USR(ADR(ENBRK$)):END
860 REM Re-enable BREAK key before end
ing
870 FOR I=0 TO 11
880 T(I+1)=PEEK(ROWCRS+I):POKE ROWCRS+
I,PEEK(TXTROW+I):NEXT I
890 REM Copy split screen values to co
rresponding full screen value location
s
900 POKE BOTSCR,24
910 REM This fools the Operating Syste
m into thinking this is a normal
920 REM Graphics 0 display vice split
screen display
930 RETURN
940 FOR I=0 TO 11
950 REM Restore things back to normal
960 POKE ROWCRS+I,T(I+1):NEXT I:POKE B
OTSCR,4
970 RETURN
980 END
```

●

CHECKSUM DATA

(see p. 70)

```
100 DATA 937,617,176,415,244,275,839,5
16,330,75,770,761,588,767,611,7921
250 DATA 111,713,525,436,800,677,258,9
77,503,603,790,245,515,777,702,8632
400 DATA 80,515,86,934,217,957,897,834
,622,505,167,905,640,257,508,8124
550 DATA 351,338,125,314,477,331,908,6
60,140,974,909,702,984,230,921,8364
700 DATA 686,415,774,905,698,529,819,9
30,217,564,135,328,904,4,514,8422
850 DATA 667,921,146,645,521,580,115,7
17,606,139,902,245,618,66,6888
```

●

(see chart on opposite page.)

ALTERNATIVE KEYBOARD HANDLER

Table 1

LETTER	KEYCODE HEX	KEYCODE DEC	INTERNAL HEX	INTERNAL DEC	ATASCII HEX	ATASCII DEC
SPACE	21	33	00	0	20	32
!	5F	95	01	1	21	33
"	5E	94	02	2	22	34
#	5A	90	03	3	23	35
$	58	88	04	4	24	36
%	5D	93	05	5	25	37
&	5B	91	06	6	26	38
'	73	115	07	7	27	39
(	70	112	08	8	28	40
)	72	114	09	9	29	41
*	07	7	0A	10	2A	42
+	06	6	0B	11	2B	43
,	20	32	0C	12	2C	44
-	0E	14	0D	13	2D	45
.	22	34	0E	14	2E	46
/	26	38	0F	15	2F	47
0	32	50	10	16	30	48
1	1F	31	11	17	31	49
2	1E	30	12	18	32	50
3	1A	26	13	19	33	51
4	18	24	14	20	34	52
5	1D	29	15	21	35	53
6	1B	27	16	22	36	54
7	33	51	17	23	37	55
8	35	53	18	24	38	56
9	30	48	19	25	39	57
:	42	66	1A	26	3A	58
;	02	2	1B	27	3B	59
<	36	54	1C	28	3C	60
=	0F	15	1D	29	3D	61
>	37	55	1E	30	3E	62
?	66	102	1F	31	3F	63
@	75	117	20	32	40	64
A	7F	127	21	33	41	65
B	55	85	22	34	42	66
C	52	82	23	35	43	67
D	7A	122	24	36	44	68
E	6A	106	25	37	45	69
F	78	120	26	38	46	70
G	7D	125	27	39	47	71
H	79	121	28	40	48	72
I	4D	77	29	41	49	73
J	41	65	2A	42	4A	74
K	45	69	2B	43	4B	75
L	40	64	2C	44	4C	76
M	65	101	2D	45	4D	77
N	63	99	2E	46	4E	78
O	48	72	2F	47	4F	79
P	4A	74	30	48	50	80
Q	6F	111	31	49	51	81
R	68	104	32	50	52	82
S	7E	126	33	51	53	83
T	6D	109	34	52	54	84
U	4B	75	35	53	55	85
V	50	80	36	54	56	86
W	6E	110	37	55	57	87
X	56	86	38	56	58	88
Y	6B	107	39	57	59	89
Z	57	87	3A	58	5A	90
[	60	96	3B	59	5B	91
/	46	70	3C	60	5C	92
]	62	98	3D	61	5D	93
▷	47	71	3E	62	5E	94
_	4E	78	3F	63	5F	95
a	3F	63	61	97	61	97
b	15	21	62	98	62	98
c	12	18	63	99	63	99
d	3A	58	64	100	64	100
e	2A	42	65	101	65	101
f	38	56	66	102	66	102
g	3D	61	67	103	67	103
h	39	57	68	104	68	104
i	0D	13	69	105	69	105
j	01	1	6A	106	6A	106
k	05	5	6B	107	6B	107
l	00	0	6C	108	6C	108
m	25	37	6D	109	6D	109
n	23	35	6E	110	6E	110
o	08	8	6F	111	6F	111
p	0A	10	70	112	70	112
q	2F	47	71	113	71	113
r	28	40	72	114	72	114
s	3E	62	73	115	73	115
t	2D	45	74	116	74	116
u	0B	11	75	117	75	117
v	10	16	76	118	76	118
w	2E	46	77	119	77	119
x	16	22	78	120	78	120
y	2B	43	79	121	79	121
z	17	23	7A	122	7A	122
RET	0C	12	—	—	9B	155
CAPS	3C	60	—	—	—	—
BACKSP	34	52	7E	126	7E	126
ESC	1C	28	5B	91	1B	27
TAB	2C	44	7F	127	7F	127
ATARI	27	39	—	—	—	—

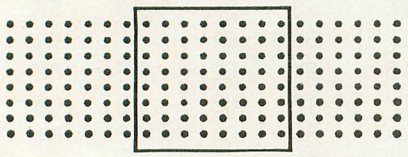
FINE SCROLLING PART III

16K Cassette or Disk

by Kyle Peacock

Hello, and welcome to the third of a four-part series on fine scrolling. Hopefully you've been keeping up with my previous installments and are well on your way to mastering the art of horizontal fine scrolling. Now we'll turn the tables slightly, and focus our attention on vertical scrolling.

Last issue, we arranged our screen RAM to look something like this:



Screen arrangement.

The dotted lines represent pages of screen memory. The box represents the television screen. Each dotted line is being displayed through the use of a Load Memory Scan (LMS) instruction. By incrementing or decrementing the horizontal scroll register HSCROL (location: 54276, $D404 hex) and incrementing or decrementing the address operands on the LMS instructions, we got horizontal scrolling. Partial examination of a display list incorporating horizontal scrolling reveals the following:

ANTIC Instruction		Explanation
Decimal	Hex	
112	$70	Blank 8 Scan Lines
112	$70	Blank 8 Scan Lines
112	$70	Blank 8 Scan Lines
82	$52	LMS Gr.0 w/horizontal scroll
Lo-Ptr 1	$Lo-Ptr 1	Lo-byte of Screen RAM
Hi-Ptr 1	$Hi-Ptr 1	Hi-byte of Screen RAM
82	$52	LMS Gr.0 w/horizontal scroll
Lo-Ptr 2	$Lo-Ptr 2	Lo-byte of Screen RAM
Hi-Ptr 2	$Hi-Ptr 2	Hi-byte of Screen RAM
82	$52	LMS Gr.0 w/horizontal scroll
Lo-Ptr 3	$Lo-Ptr 3	Lo-byte of screen RAM
Hi-Ptr 3	$Hi-Ptr 3	Hi-byte of screen RAM

(remainder of display list follows)

Pay particular attention to the three lo/hi byte address operands on the LMS instructions. I'll explain why they're so valuable a little later.

Vertical scroll bits (and pieces).

As you now know, the hardware register HSCROL controls horizontal scrolling. So it only makes sense that Atari would install another register to control vertical scrolling. This register is entitled VSCROL (location 54277, $D405 hex). Upon changing this hardware register, any line of text (or bit mapped graphics) with its vertical scroll bit set will smooth scroll up or down. To enable (turn on) the vertical scroll bit of a display list instruction, all you do is add 32 ($20 hex) to that instruction. Let's take a look at what I mean.

ANTIC Instruction		Explanation
Decimal	Hex	
02	$02	Display 1 Gr.0 line of text
+32	+$20	Add on to set vertical scroll bit
34	$22	New ANTIC Instruction
66	$42	Gr.0 LMS Instruction
xx	$XX	Lo-byte of screen RAM address
yy	$YY	Hi-byte of screen RAM address
+32	+$20	Add on to set vertical scroll bit
98	$62	New ANTIC Instruction
xx	$XX	Lo-byte of screen RAM address
yy	$YY	Hi-byte of screen RAM address

Note that only the opcode was changed, not the lo and hi bytes pointing to the screen RAM.

Here's a short BASIC program that sets the vertical scroll bit on the first three lines of a graphics mode zero screen, and smoothly scrolls the text on these lines.

Listing 1.

```
100 REM ********************************
110 REM *  VERTICAL SCROLLING DEMO     *
120 REM *        LISTING #1            *
130 REM *     BY KYLE S. PEACOCK       *
```

```
140 REM *     ANALOG COMPUTING #15    *
150 REM ********************************
160 REM
170 GRAPHICS 0
180 FOR X=1 TO 3
190 ? "THIS IS A TEST OF VERTICAL SCRO
LL"
200 NEXT X
210 REM
220 REM --- FIND THE DISPLAY LIST
230 REM
240 DLIST=PEEK(561)*256+PEEK(560)
250 BIT1=PEEK(DLIST+3)
260 BIT2=PEEK(DLIST+6)
270 BIT3=PEEK(DLIST+7)
280 REM
290 REM --- NOW ADD 32 (HEX: $20) TO
300 REM --- DISPLAY LIST INSTRUCTIONS
310 REM
320 POKE DLIST+3,BIT1+32
330 POKE DLIST+6,BIT2+32
340 POKE DLIST+7,BIT3+32
350 REM
360 REM --- VERTICAL SCROLL
370 REM
380 FOR X=0 TO 7
390 POKE 54277,X
400 FOR T=0 TO 50:NEXT T
410 NEXT X
420 GOTO 380
430 REM
440 REM --- TO REVERSE DIRECTION OF
450 REM --- SCROLLING CHANGE LINE 290
460 REM --- TO: FOR X=7 TO 0 STEP -1
```

CHECKSUM DATA

(see p. 70)

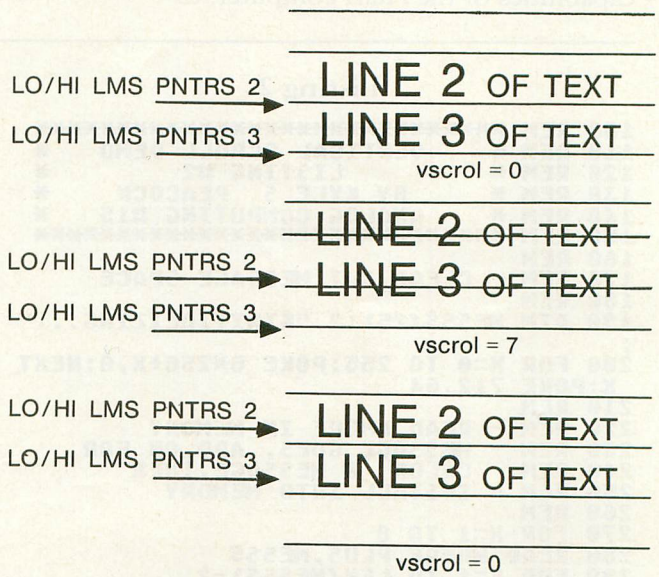
```
100 DATA 778,989,860,486,620,793,92,90
5,320,551,760,79,395,85,209,7922
250 DATA 141,150,157,100,253,353,81,31
6,326,332,93,769,99,329,365,3864
400 DATA 145,767,723,89,769,967,453,39
13
```

Several things become apparent when looking at the display generated by **Listing 1**. First, the graphics mode 0 screen has only 23 rows instead of the normal 24. This is due to the hardware arrangement of the computer. If, for example, a display list is designed to display 70 lines of bit-mapped graphics with vertical scroll bits enabled, ANTIC will only display 69. The 70th line is used as a vertical scroll 'buffer.' This one-line buffer will be scrolled into the screen area as the first line with its vertical scroll bit enabled is scrolled off. Notice how the cursor gradually scrolls into view as the program cycles through VSCROL. What is happening is that the cursor resides on that one line buffer, thus making it 'glide' into view.

You may also notice how the text on the screen smooth scrolls for a while, then 'jumps' back to its original position and repeats again. Let's think about what ANTIC is doing. By changing the value of VSCROL from zero all the way to seven, we shift the text upward. When VSCROL has a value of 0, the character in row one is positioned directly over the character in row two, the character in row two is positioned directly over the character in row three, and so on. When VSCROL finally gets a value of 7, the characters in row two are *very nearly* where the characters in row one *used* to be, the characters in row three are close to where the characters in row 2 used to be, etc. When VSCROL goes back to 0 again, the characters resume their normal positions.

To achieve true vertical scrolling, we must reset VSCROL back to zero AND adjust the LMS address operands in the display list.

Gee Kyle, I've been keeping up with you since Issue 13, but that last paragraph looked like TRS-80 control characters. How about an illustration? Sure!



Note how the LMS operands were shifted upward when 'vscrol' changes from 7 back to 0.

You can see from the above illustration that the more lines with vertical scroll bits enabled, the more operands we have to shift and the more we bog down the processor. You may also notice some occasional screen flicker while the program smooth-scrolls the text. This is due to the way ANTIC generates its display and the way BASIC executes a program. If we try to change the operands on our LMS instructions while ANTIC is refreshing the screen, the screen flickers. This problem is a little more difficult to solve and requires some assembly language programming. Let's save this problem for next issue's final installment, "Fine Scrolling Part IV: Taking the Plunge."

Wondering why **Listing 1** only used a *scrolling* range of 0 to 7? The value stored in VSCROL tells ANTIC how many TV scan lines to scroll upward. Graphics mode 0 characters are only 8 scan lines high. That's why I restricted the scroll to no more than seven units. The following table lists the most common ANTIC graphics modes and their associated VSCROL ranges.

Graphics Mode	VSCROL Range
0	0-7
1	0-7
2	0-15
3	0-7
4	0-3
5	0-3
6	0-1
6+	*
7	0-1
7+	*
8	*

*Do not change VSCROL, simply update all LMS operands.

Listing 2 demonstrates the fine vertical scrolling capabilities of the Atari computer. □

Listing 2.

```
100 REM ********************************
110 REM *    VERTICAL SCROLL DEMO      *
120 REM *          LISTING #2          *
130 REM *      BY KYLE S. PEACOCK      *
140 REM *    ANALOG COMPUTING #15      *
150 REM ********************************
160 REM
170 REM - CLEAR OUT MESSAGE SPACE
180 REM
190 DIM MES$(25):? "{INITIALIZING..."
;
200 FOR X=0 TO 255:POKE 6*256+X,0:NEXT
 X:POKE 712,64
210 REM
220 REM - READ WHERE IN MEMORY
230 REM - MESSAGE GOES, ADD ON FOR
240 REM - COLOR, & MESSAGE,THEN
250 REM - INSTALL INTO MEMORY
260 REM
270 FOR X=1 TO 8
280 READ WHERE,PLUS,MES$
290 FOR Y=1 TO LEN(MES$)-2
300 POKE WHERE+Y-1,ASC(MES$(Y+1,Y+1))
-32+PLUS
310 NEXT Y:NEXT X
320 REM
330 REM - NOW READ & INSTALL
340 REM - CUSTOMIZED DISPLAY LIST
350 REM
360 TRAP 400
370 READ DLIST
380 POKE 6*256+185+ADD,DLIST
390 ADD=ADD+1:GOTO 370
400 REM
410 REM - NOW TELL ANTIC WHERE TO
420 REM - FIND OUR NEW DISPLAY LIST
430 REM
440 POKE 560,185:POKE 561,6
450 REM
460 REM - CHANGE VALUE OF 'VSCROL'
470 REM
480 FOR X=0 TO 7:POKE 54277,X
490 REM
500 REM - LOOP VALUE OF 'Y' DICTATES
510 REM - SPEED OF VERTICAL SCROLL
520 REM
530 FOR Y=0 TO 50:NEXT Y:NEXT X
540 REM
550 REM - NOW SHIFT LMS OPERANDS
560 REM
570 REM - NOTE HOW ONLY THE LOW BYTE
580 REM - OPERANDS WERE CHANGED.  IN
590 REM - THIS PARTICULAR CASE, WE
600 REM - DON'T NEED TO CHANGE THE
610 REM - HIGH BYTE OPERANDS (SINCE
620 REM - ALL THE MESSAGES RESIDE ON
630 REM - PAGE SIX...)
640 REM
650 REM - NO! WE CAN'T GET RID OF
660 REM - THAT NASTY GLITCH!!!!!!
670 REM
680 POKE 54277,0
690 HOLD=PEEK(6*256+185+17)
700 FOR SHIFT=17 TO 29 STEP 3
710 POKE 6*256+185+SHIFT,PEEK(6*256+18
5+SHIFT+3)
720 NEXT SHIFT
730 POKE 6*256+185+32,HOLD
740 GOTO 460
750 REM
760 REM - MESSAGES
770 REM
780 DATA 1536,64
790 DATA "    A.N.A.L.O.G.    "
800 DATA 1556,64
810 DATA "       MAGAZINE     "
820 DATA 1576,192
830 DATA "WE ARE GOING MONTHLY"
840 DATA 1596,192
850 DATA "      THIS ISSUE    "
860 DATA 1616,0
870 DATA " SEE YOU IN 30 DAYS "
880 DATA 1636,0
890 DATA "       WHEN  WE     "
900 DATA 1656,128
910 DATA "  TAKE THE PLUNGE!  "
920 DATA 1676,64
930 DATA "      ISSUE  #15    "
940 REM
950 REM - DISPLAY LIST
960 REM
970 DATA 112,112,112,112
980 DATA 70,0,6,112,112
990 DATA 70,20,6,112,112,112,112
1000 REM
1010 REM - VERTICAL SCROLL BIT IS NOW
1020 REM - SET. CHANGING OF VSCROL
1030 REM - WILL NOW SMOOTH SCROLL THE
1040 REM - FOLLOWING LINES OF TEXT.
1050 REM
1060 DATA 102,40,6
1070 DATA 102,60,6
1080 DATA 102,80,6
1090 DATA 102,100,6
1100 DATA 102,120,6
1110 REM
1120 REM - VERTICAL SCROLL BIT IS NO
1130 REM - LONGER SET. THE NEXT LINE
1140 REM - DISPLAYED WILL ACT AS OUR
1150 REM - ONE LINE SCROLL 'BUFFER.'
1160 REM
1170 DATA 70,160,6
1180 DATA 112,112,112,112,112,112,112
1190 DATA 70,140,6
1200 DATA 65,185,6
```

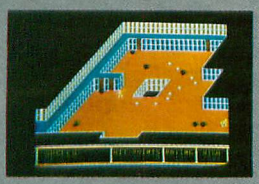
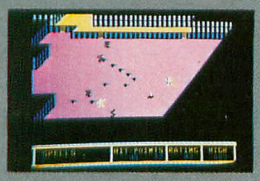
•

CHECKSUM DATA
(see p. 70)

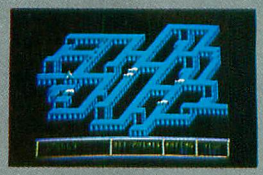
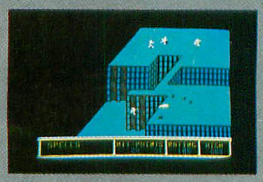
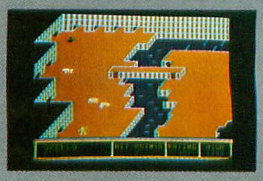
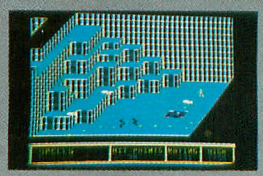
```
100 DATA 778,803,896,486,620,793,92,45
3,98,55,709,79,232,741,326,7161
250 DATA 66,94,329,790,604,31,523,84,8
36,694,93,718,70,776,80,5788
400 DATA 80,492,764,89,370,95,619,101,
674,107,969,911,88,376,94,5829
550 DATA 369,100,44,896,377,700,658,88
7,29,96,96,173,105,288,364,5182
700 DATA 545,989,137,180,731,101,580,1
07,225,606,205,576,58,325,66,5431
850 DATA 718,957,128,965,559,59,10,218
,495,102,334,108,812,606,491,6562
1000 DATA 275,137,283,209,772,280,138,
141,144,403,401,279,812,774,927,5975
1150 DATA 701,284,165,137,165,175,1627
```

•

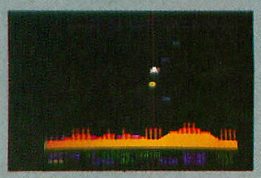
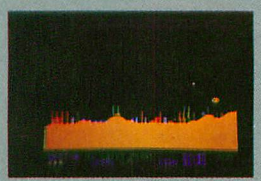
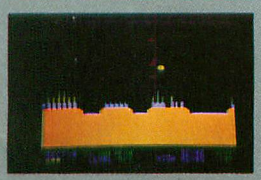
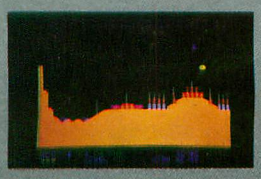
Two for the ATARI* from BRAM Inc. by Mike Edwards

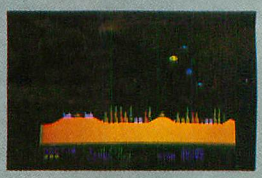
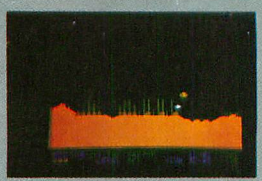
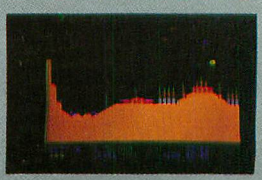
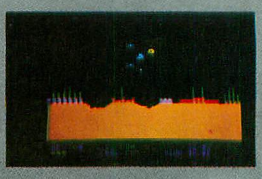

ZOMBIES©

The exciting new arcade hit that features scrolling 3D graphics, on-line instructions, one or two player cooperative, seven different dungeons, 74 different screens, high score save to disk, full sound and color, zombies, poisonous snakes, giant spiders, evil orbs, scrolls, talismans, magic spells, lost crowns and spectacular underground scenery. 48K tape and disk. $34.95

ATTACK at EP-CYG-4©

If you're ready for the most challenging arcade space-war game, you're ready for ATTACK at EP-CYG-4. This is an adventure for experienced arcaders. EP-CYG-4 features 3 missions, 130 screens*, Tartillian attack fighters, Cruisers, ground batteries, landscapes, seascapes, Gravitron drive ships, Lyso-blast weapons, advanced joy stick controls, one or two player cooperative.

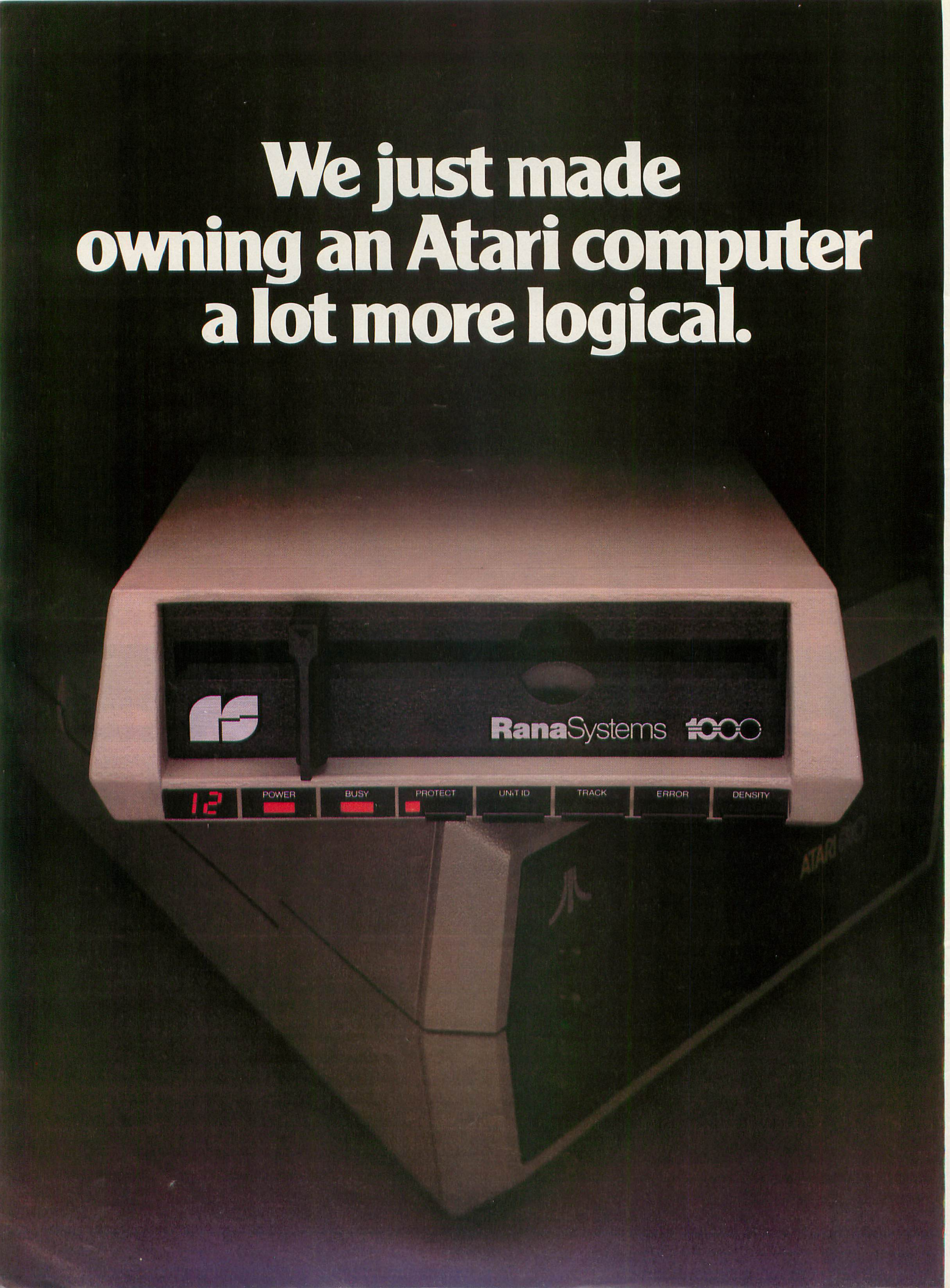
"I'd recommend EP-CYG-4 to any gamer who is looking for new concepts and challenges in his or her gaming life." D. Duberman, December, 1982 ANTIC

*Disk 32K $32.95; tape 16K $29.95; 2 missions, 70 screens

BRAM Inc.

18779 Kenlake Place N.E. Seattle, Washington 98155 (206) 486-8428

We just made owning an Atari computer a lot more logical.



Introducing the Rana 1000 disk drive. It's a whole new game for Atari computers.

This two digit LED readout displays a code that tells you everything you need to know.

This beeping button tells you your write protect feature is keeping your information safe.

The remaining buttons beep when touched, and provide readouts on density storage, error status, and drive number.

This button beeps when you touch it, and the LED readout tells you what track you're on.

When Rana Systems introduced the Elite Series of Apple® compatible disk drives, we didn't know what a tremendous impact they would make. It turned out to be a line so outstanding in performance, styling, capacity, and price, that it instantaneously made us a major force in the market. Well, needless to say, the response was so great that we were forced to create the same highly advanced disk drive for Atari.® A disk drive that when coupled with Atari's computer, could perform everything from accounting, financial planning, and stock charting, to word processing, business management, and letting you write your own programs. Plus, we made it simple enough for a child to use, for learning anything from the alphabet to a foreign language.

Working with a diskette versus playing with a cassette.

Let's face it. The only reason Atari made a cassette option to their computer was to make it affordable. But now you don't have to settle for less. Because now you can get a diskette for your Atari computer which outperforms their cassette and costs ⅓ less than their disk drive. With Atari's cassette you only get half the functions of a computer compared to what our floppy disk can give you. Their cassette is not only limited in the software available, but it also takes 20 times longer to get the information you need. And Rana's disk drive offers twice the storage capacity of either their cassette or disk drive.

Why even stylewise our new low profile design not only looks 100 times more spectacular, but it occupies 3 times less space. And our new Rana 1000 also gives you a piece of its mind every time you use it, because our disk drive gives you information as well as takes it. And we think that says a lot.

The disk drive that has all the answers.

Rana offers you a myriad of features Atari couldn't even conceive of. Like five electronic functions on the front panel that actually beep and give you a LED readout when touched. Our disk drive tells you what track you're on, and what density and how much information you're storing. It lets you switch from a single density of 90,000 letters to a double density of 180,000 letters, on a single diskette. And, we have a write protect feature which protects your diskette from being erased. In fact, no other disk drive can offer you that.

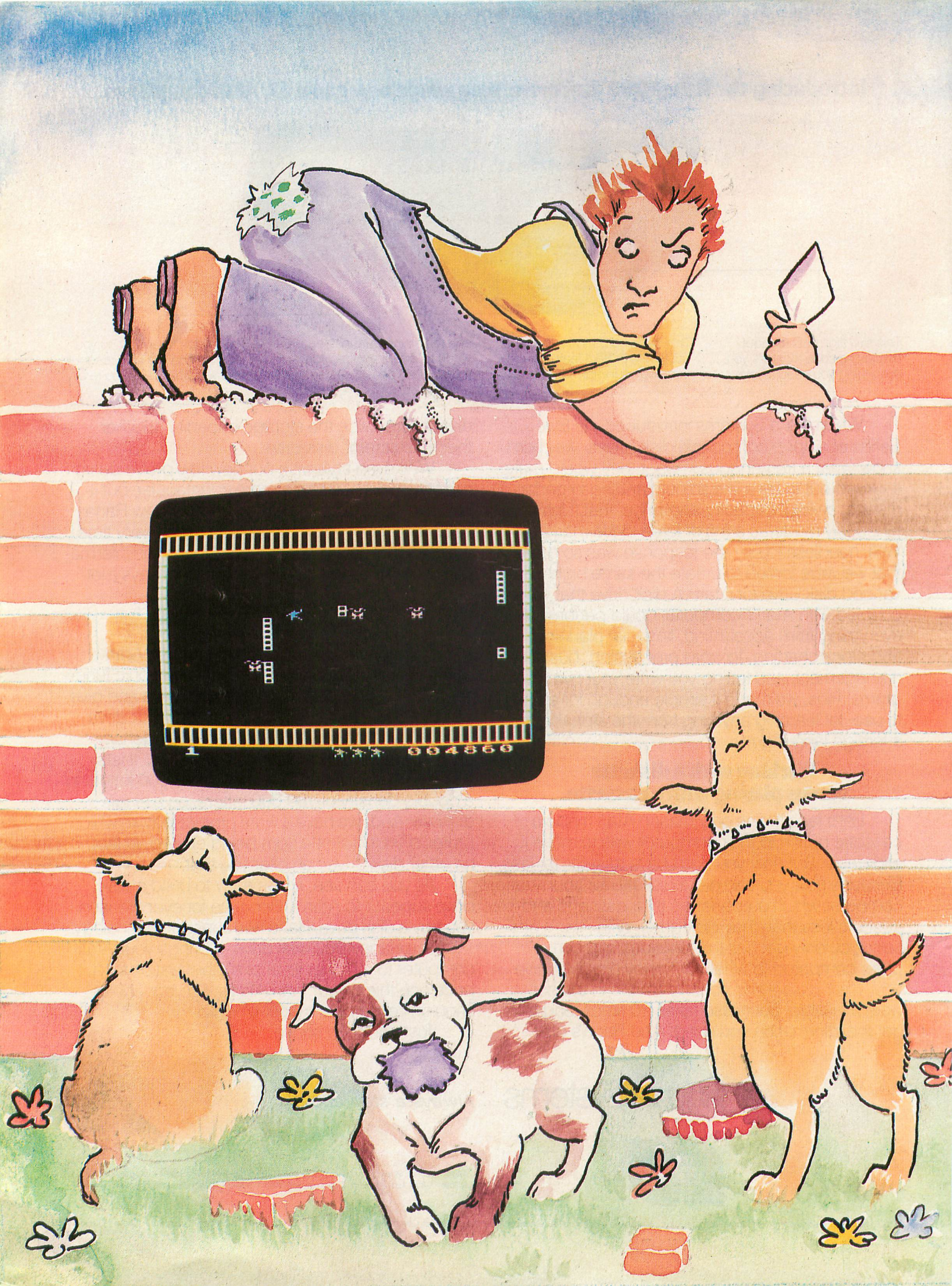
As you can see, it was easy to build a disk drive superior to Atari's. Because for every reason you buy a disk drive, Rana has superior technology.

The Rana 1000 disk drive. It brings your Atari computer to a higher level of sophistication for a price one third lower than Atari's. So your choice shouldn't even be a matter of logic.

Just common sense.

RanaSystems *Always a step ahead.*

21300 Superior Street, Chatsworth, CA 91311 213-709-5484. For dealer information call toll free: 1-800-421-2207. In California only call: 1-800-262-1221. Source Number: TCT-654


1
004560

BRICKLAYER'S NIGHTMARE

16K Cassette or Disk

by Gordon Robson

Bricklayer's Nightmare is a machine-language action game that turns your TV screen into a construction site, and you into a blue-collar hero. Your job is to move a wall of bricks from the middle of the screen to the right edge, one brick at a time. Your efforts will be constantly hampered by a pack of bloodthirsty neighborhood dogs, who will chase you around for a chance to sink a tooth into your leg.

45 points are awarded every time a brick is moved one position to the right. When you finish moving an entire wall of bricks, another will appear along with an even more relentless pack of mutts. The game has seven levels of difficulty; should you manage to complete all of them, the most difficult level will repeat.

Your construction worker is moved around the screen with joystick #1. Each brick must be "pulled" across the screen with the worker's electromagnetic attraction field. The procedure for moving a brick is as follows:

a. Position the worker to within touching distance of the right side of a brick;

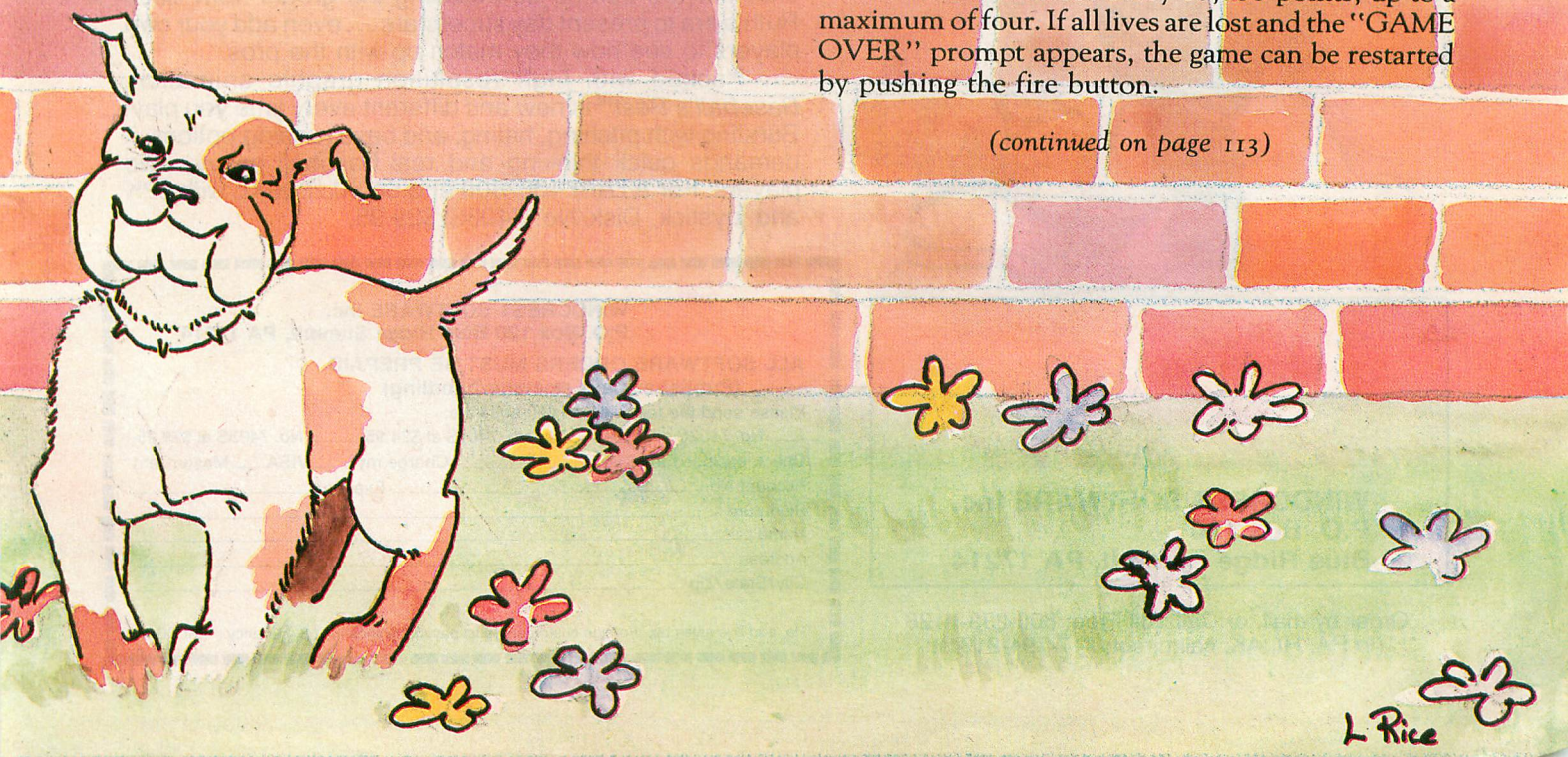
b. Push the red fire button to activate the attraction field;

c. Move the joystick to the right while holding down the fire button. The brick will follow the worker across the screen.

Bricks can only be moved from left to right, so be careful!

Your bricklayer begins the game with three lives. You will lose a life if one of the dogs touches you. An extra life is earned for every 10,000 points, up to a maximum of four. If all lives are lost and the "GAME OVER" prompt appears, the game can be restarted by pushing the fire button.

(continued on page 113)



Take Charge:

as a military commander . . .
or as a big league baseball manager . . .

PANZER WAR ™

Your own ingenuity and strategic know-how will determine the victor in this unique and exciting war game! It's the ultimate in tactical challenge, filled with fast-paced arcade action.

Set on the Eastern Front during WWII, it's a game that pits crack German Panzers against a strong and determined Russian Tank Cavalry. You choose from seven scenarios and four battle configurations . . . four difficulty levels. It's a tough assignment and situations can change rapidly. One or two players. For ATARI 400/800 and 600XL/800XL with 32K and Joystick. Disk/No. 7402S/ $29.95 *or* Cassette/No. 7401S/$24.95.

BASEBALL'S BEST ™

You're the manager . . . it's up to you to choose the teams and lineups, to call plays and make those critical decisions that can win or lose a ballgame!

Select pre-chosen teams, create your own teams, draft players from a file including the greats from Babe Ruth's era to present day superstars . . . even add your own players to see how they match up with the pros.

Packed with high resolution graphic animation, *Baseball's Best*™ is new and different every time you play! Bursting with pitching, hitting, and base running options, it demands quick thinking and real baseball savvy. Two players. For ATARI 400/800 and 600XL/800XL with 32K and joystick. Disk/No. 7403S/$29.95.

WINDCREST

WINDCREST SOFTWARE Inc.
P.O. Box 120
Blue Ridge Summit, PA 17214

Order by mail, or Call Toll Free: 800-233-1128
(In PA, HI, AK, call direct: 717-794-2191).

WINDCREST

WINDCREST SOFTWARE Inc.
P.O. Box 120 Blue Ridge Summit, PA 17214

ALL SOFTWARE ORDERS MUST BE PREPAID
(Publisher pays postage/handling)

Please send the following ATARI software:

___ No. 7402S at $29.95 ___ No. 7401S at $24.95 ___ No. 7403S at $29.95

Check enclosed for $ _____ Charge my ___ VISA ___ MasterCard

Account No. _____ Expires _____

Signature _____

Name _____

Address _____

City/State/Zip _____

Pa. add 6% sales tax. Foreign customers must pay in advance in U.S. currency. ALOG-14

(continued from p. 111)

Typing the program.

Before typing anything into your computer, let's look at the program listings included with this article and see what they do.

Listing 1 is the main data and data checking routine. This BASIC program is used to create both the disk and cassette versions of **Bricklayer's Nightmare.** The DATA statements are listed in hexadecimal format (base 16) so that the program will fit in a 16K cassette system.

Listing 2 must be added to **Listing 1** if you are using a 410 or 1010 program recorder.

Listing 3 must be added to **Listing 1** if you are using a disk drive.

Listing 4 is the assembly-language source code for **Bricklayer's Nightmare.** You do NOT have to type in this listing to play the game! It is provided for those readers interested in seeing how the program works.

Follow the instructions below to make either a cassette or disk version of **Bricklayer's Nightmare**.

Cassette instructions.

1. Type **Listing 1**into your computer using the BASIC cartridge, and verify your typing with **C:CHECK** (see page 70).

2. With **Listing 1** still in the computer, carefully type in **Listing 2**. The program lines will automatically merge together with **Listing 1**. It's a good idea to CSAVE the entire program at this point.

3. Type RUN and press RETURN. The program will begin checking the DATA statements, printing the line number of each statement as it goes. It will alert you if it finds any problems. Fix incorrect lines and re-RUN the program if necessary until all errors are eliminated.

4. When all DATA lines are correct, the computer will beep twice and prompt you to "READY CASSETTE AND PRESS RETURN." Insert a blank cassette in your recorder, press the RECORD and PLAY buttons simultaneously and hit RETURN. The message "WRITING FILE" will appear and the program will create a machine-language boot tape version of **Bricklayer's Nightmare**, printing each DATA line number as it goes. When the READY prompt appears, the game is recorded and ready to play. CSAVE the BASIC program onto a separate tape before continuing.

5. To play the game, rewind the tape created by the BASIC program to the beginning. Turn your computer OFF and remove all cartridges. Press the PLAY button on your recorder and turn ON your computer while holding down the START key. The computer will "beep" once. Hit the RETURN key and **Bricklayer's Nightmare** will load and run automatically.

Disk instructions.

1. Type **Listing 1** into your computer using the BASIC cartridge, and verify your typing with **D:CHECK2** (see page 70).

2. With **Listing 1** still in the computer, carefully type in **Listing 3**. The program lines will automatically merge together with **Listing 1**. It's a good idea to SAVE the entire BASIC program at this point.

3. Type RUN and press RETURN. The program will begin checking the DATA statements, printing the line number of each statement as it goes. It will alert you if it finds any problems. Fix incorrect lines and re-RUN the program if necessary until all errors are eliminated.

4. When all DATA lines are correct, you will be prompted to "INSERT DISK WITH DOS, PRESS RETURN." Put a disk containing DOS 2.0S into drive #1 and press RETURN. The message "WRITING FILE" will appear and the program will create an AUTORUN.SYS file on the disk, displaying each DATA line number as it goes. When the READY prompt appears, the game is ready to play. Be sure the BASIC program is SAVEd before continuing.

5. To play the game, insert the disk containing the AUTORUN.SYS file into drive #1. Turn your computer OFF, remove all cartridges and turn the computer back ON. **Bricklayer's Nightmare** will load and run automatically. □

Listing 1.

```
1 REM *** BRICKLAYER'S NIGHTMARE ***
10 DATA 0,1,2,3,4,5,6,7,8,9,0,0,0,0,0,
0,0,10,11,12,13,14,15
20 DIM DAT$(91),HEX(22):FOR X=0 TO 22:
READ N:HEX(X)=N:NEXT X:LINE=990:RESTOR
E 1000:TRAP 60:? "CHECKING DATA"
25 LINE=LINE+10:? "LINE:";LINE:READ DA
T$:IF LEN(DAT$)<>90 THEN 110
28 DATLIN=PEEK(183)+PEEK(184)*256:IF D
ATLIN<>LINE THEN ? "LINE ";LINE;" MISS
ING!":END
30 FOR X=1 TO 89 STEP 2:D1=ASC(DAT$(X,
X))-48:D2=ASC(DAT$(X+1,X+1))-48:BYTE=H
EX(D1)*16+HEX(D2)
35 IF PASS=2 THEN PUT #1,BYTE:NEXT X:R
EAD CHKSUM:GOTO 25
40 TOTAL=TOTAL+BYTE:IF TOTAL>999 THEN
TOTAL=TOTAL-1000
45 NEXT X:READ CHKSUM:IF TOTAL=CHKSUM
THEN 25
50 GOTO 110
60 IF PEEK(195)<>6 THEN 110
100 ? "WRITING FILE":PASS=2:LINE=990:R
ESTORE 1000:TRAP 60:GOTO 25
110 ? "BAD DATA: LINE ";LINE:END
1000 DATA 3F3F3F3F3F3F3F3F3F3F3F3F3F
3F3F3F3F3F3F7070707047802C44002A040404
0404040404040404040404040404,121
1010 DATA 47802C46473341942CA9008D1DD0
A9228D2F02A9208D3002A9BC8D3102A9E08DF4
0220C12D20422D20A0322008,169
1020 DATA 32208B2E20422D20F03120B731D0
3520BB2EADE933D0034CD52CCEFE33D01AADFC
338DFE33AD8402D006ADFD33,356
```

```
1030 DATA 8DFE33204D30205C30206B3020AA
31ADEA33D0C94CE42C206132CEEB3310B8A200
BD3D33C9FFF00909C09D4A33,306
1040 DATA E84C282DAD8402D0FB854D4CCF2C
A900858DA9238581A9008584A9058585A20020
6931A204A9009D07D0CAD0FA,487
1050 DATA A9868DC002A9488DC1028DC2028D
C302A9508DF4338D00D0A9788DF6338D01D0A9
8C8DF7338D02D0A9A08DF833,356
1060 DATA 8D03D0A9AF8DF5338DF9338DFA33
8DFB33A9018DF33320F22FA208BDC4339DAE25
9DAE269DAE27CAD0F1A9208D,447
1070 DATA 07D48D1ED060A9008582ADF40285
83A9008580A9288581A9008584A90285852057
31A218BD64339DE729CAD0F7,808
1080 DATA A9288DF40220752EA9948D3002A9
2C8D3102A9008DC802A9288DC402A90A8DC502
A9D88DC602A9388DC702A93E,615
1090 DATA 8D2F02A9038D1DD0A214A9009D46
33CAD0FAAD5B33A2069D5333CAD0FAAD5B3309
408D4833A9038DEB33A93F8D,702
1100 DATA EE338D002AA210A9008580A92A85
81A92720622ECAF008A90120622E4C512E6018
65808580A5816900858 1ADEE,320
1110 DATA 33A0009180 60A9008580A92A8581
A9808584A9028585A2002069316 0AD4833290F
AAA900CAF0061869044C932E,787
1120 DATA AABDCD338DFC33BDCE338DFD33BD
CF338DEC33BDD0338DED33A9FA8DE93360AD78
02C90BD03120D32ECEF333D0,190
1130 DATA 05A9058DF33320F22F60CEF433A9
33CDF433F012ACF433AEF53320D92FD007ADF4
338D00D060EEF43360C907F0,138
1140 DATA 0BC90EF034C90DF0664C842F200B
2F20C1324CC52EEEF433A9C6CDF433F016AEF5
33ADF43318690 7A820D92FD0,559
1150 DATA 07ADF4338D00D060CEF43360CEF5
33A938CDF533F00BACF433AEF53320B12FF003
EEF533EEF333A90ACDF333D0,943
1160 DATA 05A9068DF333AEF533A9009D0824
9DFF2320F22F60EEF533A9B0CDF533F00FACF4
33ADF53318690 7AA20B12FF0,635
1170 DATA C9CEF5334C472FC90AF00DC906F0
10C909F013C905F0166020D32E202F2F60200B
2F202F2F6020D32E20652F60,880
1180 DATA 200B2F20652F608CEE338EEF3320
1230D01CAEEF33ADEE3318690 4A8201230D00D
AEEF33ADEE3318690 6A82012,266
1190 DATA 30608CEE338EEF33201230D00DAC
EE33ADEF33186907AA20123060AEF333A9F818
6908CAD0FAA8A9088584AEF5,725
1200 DATA 33B97D339D0024E8C8C684D0F460
8A38E9384A4A4AAA9838E9304A4AA8A9008580
A92A8581CA30101 8A5806928,951
1210 DATA 8580A5816900858 14C2930981865
808580A900658185 81A000B18060A9008DFF33
201231F00160207730 60A901,443
1220 DATA 8DFF33207730F00160201231 60A9
028DFF33201231207730 60A9008580A9258581
ACFF33F006E681884C8230AE,178
1230 DATA FF33BDF933CDF533D00160B041A5
8048A58148BDF633A8BDF933186908AA20B12F
F0056868A90060AEFF336885,892
1240 DATA 81688580BCF9339180FEF933C8A2
00A9088584BDC5339180E8C8C684D0F5A90160
AEFF33A58048A58148BDF633,393
1250 DATA A8BDF933AACA20B12FF005686 8A9
0060AEFF3368858168858 0DEF933BDF9334818
6908A8A900918068A84CC530,30
1260 DATA AEFF33BDF633CDF433D00160B017
BDF633186908A8204731F00160FEF633BDF633
9D01D060BCF63388204731F0,792
1270 DATA 0160DEF6334C30318CF033BDF933
AA20D92F08AEFF332860A000B1829180207731
2085312093 1D0F1608AA000,816
1280 DATA 9180208531209331D0F36018A582
69018582A5836900858360 18A5806901858 0A5
8169008581603 8A584E90185,688
1290 DATA 84D005A585D00360A585E9008585
A90160AEEC33ACED3388D0FDCAD0FA60AD0CD0
290E60AD55338DEE33A205FE,679
1300 DATA 5433BD5433C91AD008A9109D5433
CA10EE88D0E9ADEE33CD5533F00DADEB33C904
F006EEEB3320F03160A004A9,331
1310 DATA 00994F3388D0FAACEB33F008A9BD
994F3388D0FA60AD4833290FC907F003EE4833
ADC50248A9058DEE33A90C8D,779
1320 DATA C502A9808D00D2A9A68D01D2A000
A200205832A9368DC502A9C08D00D2A9A68D01
D2A000A200205832CEEE33D0,962
1330 DATA CF688DC502A9008D01D2608AAACA
D0FD88D0F960A9098DEE338D00D2A9A08DF233
8D01D2A90548A2FFA0FF8EC0,376
1340 DATA 0288D0FDADF23338E9038DF2338D
01D2CAD0E9EEEE33ADEE338D00D26838E901D0
D78D01D26020752E203F2EA9,272
1350 DATA 3E8DEE338D5B2AA209A95B8580A9
2A8581A92820622ECAD0F860A9008DEA33AD84
02D047A900854DADF4332903,414
1360 DATA D03CADF43338E908A88DF133AEF5
33201230C93EF011ADF533186908AAACF13320
1230C93ED017A000A9009180,570
1370 DATA A93EA0019180A02D20BD31CEE933
20133360A9708DF233A9088D00D2ADF2338D01
D2A005A240CAD0FD88D0F8CE,311
1380 DATA F233ADF233C958D0E6A9018DEA33
6027212D25002F362532FF0000000000000000
0000000000000000000000,86
1390 DATA 101112131415161718198090783C
1A384848AA8282AAAA8282AAFF9999999999999
FF1010387CBA488482101038,417
1400 DATA 3878AC484410103838382828101 0
10387CBA3810281010383 87C38484880907 83C
1A3848480010FC3A18242400,279
1410 DATA 0010FC3A1824240002123C78B038
242266951C22363E143608080C0003030C8003
020B0002020B0002010B5002,7
1420 DATA 010B8002010A0000000000000000
0000000000000000000000000000000000000000
0000000000000000000000,160
```

●

CHECKSUM DATA

(see p. 70)

```
1 DATA 754,955,686,427,745,192,617,545
,276,445,496,549,150,583,550,7970
1020 DATA 963,62,822,977,236,668,958,8
1,606,614,440,44,39,122,64,6696
1170 DATA 863,16,11,886,312,648,928,17
,119,926,641,376,109,947,173,6972
1320 DATA 753,239,184,914,809,46,295,6
22,353,963,405,5583
```

●

Listing 2.

```
2 REM *** CASSETTE VERSION ***
65 IF PA55=2 THEN FOR X=1 TO 73:PUT #1
,0:NEXT X:CLOSE #1:END
70 ? "READY CASSETTE AND PRESS RETURN"
;:OPEN #1,8,128,"C:":RESTORE 200:FOR X
=1 TO 40:READ N:PUT #1,N:NEXT X
200 DATA 0,16,88,44,127,44,169,0,141,4
7,2,169,60,141,2,211,169,0,141,231,2,1
33,14,169,56,141,232,2
210 DATA 133,15,169,207,133,10,169,44,
133,11,24,96
```

●

Listing 3.

```
2 REM *** DISK VERSION ***
65 IF PA55=2 THEN PUT #1,224:PUT #1,2:
PUT #1,225:PUT #1,2:PUT #1,207:PUT #1,
44:CLOSE #1:END
70 ? "INSERT DISK WITH DOS, PRESS RETU
RN";:DIM IN$(1):INPUT IN$:OPEN #1,8,0,
"D:AUTORUN.SYS"
90 PUT #1,255:PUT #1,255:PUT #1,128:PU
T #1,44:PUT #1,14:PUT #1,52
```

●

```
0100 ;          BRICKLAYER'S NIGHTMARE
0101 ;          ######################
0102 ;          BY G.M. ROBSON
0103 ;          ANALOG COMPUTING #15
0104 ;
0105 ;          DISPLAY LIST VALUES
0106 ;          -------------------
0107 BLB      =   $70           BLANK 8 SCAN LINES
0108 CH2      =   $02           CHAR LINE 40H X 8V, 2 COLOR
0109 CH4      =   $04           CHAR LINE 40H X 8V, 4 COLOR
0110 CH6      =   $06           CHAR LINE 20H X 8V, 5 COLOR
0111 CH7      =   $07           CHAR LINE 20H X 16V, 5 COLOR
0112 JVB      =   $41           JMP ON VERT BLANK INST
0113 LMS      =   $40           MASK TO LOAD MEM SCAN ADR
0114 ;
0115 ;          OS EQUATES
0116 ;          ----------
0117 ATRACT   =   $4D           ATTRACT MODE COUNTER
0118 AUDC1    =   $D201         AUDIO CONTROL 1
0119 AUDF1    =   $D200         AUDIO FREQ 1
0120 GRACTL   =   $D01D         GRAPHICS CTRL REG
0121 HITCLR   =   $D01E         COLLISION CLEAR
0122 HPOSP0   =   $D000         MAN HORIZ POS
0123 HPOSP1   =   $D001         DOG 1 HORIZ POS
0124 HPOSP2   =   $D002         DOG 2 HORIZ POS
0125 P0PL     =   $D00C         PLYR 0(MAN) AND PLAYER COLLISIONS
0126 PMBASE   =   $D407         PM BASE ADR
0127 SIZEP0   =   $D008         SIZE OF PLAYER 0
0128 HPOSP3   =   $D003         DOG 3 HORIZ POS
0129 ;
0130 ;          SYSTEM SHADOW REGS
0131 ;          ------------------
0132 CHBAS    =   $02F4         CHAR SET HO ADR
0133 COLOR0   =   $02C4         PLAYFIELD 0 COLOR REG
0134 COLOR1   =   $02C5         PLAYFIELD 1 COLOR REG
0135 COLOR2   =   $02C6         PLAYFIELD 2 COLOR REG
0136 COLOR3   =   $02C7         PLAYFIELD 3 COLOR REG
0137 COLOR4   =   $02C8         BACKGROUND COLOR
0138 ;
0139 SDLSTL   =   $0230         DISPLAY LIST PNTR
0140 PCOLR0   =   $02C0         MAN COLOR REG
0141 PCOLR1   =   $02C1         DOG 1 COLOR REG
0142 PCOLR2   =   $02C2         DOG 2 COLOR REG
0143 PCOLR3   =   $02C3         DOG 3 COLOR REG
0144 SDMCTL   =   $022F         DMA CTRL
0145 STICK0   =   $0278         JOYSTICK 1
0146 STRIG0   =   $0284         TRIGGER 1
0147 ;
0148 ;          PAGE ZERO VARIABLES
0149 ;          -------------------
0150 DESADR   =   $80           DEST ADR
0151 SRCADR   =   $82           SRCE ADR
0152 SIZE     =   $84           # OF BYTES INVOLVED
0153 ;
0154 ;          JOYSTICK VALUES
0155 ;          ---------------
0156 DOWN     =   13
0157 DOWNLEFT =   9
0158 DOWNRGT  =   5
0159 LEFT     =   11
0160 RIGHT    =   7
0161 UP       =   14
0162 ;
0163 UPLEFT   =   10
0164 UPRIGHT  =   6
0165 ;
0166 ;          BRICKLAYER VALUES
0167 ;          -----------------
0168 BMPS     =   250          BRICK MOVES PER SCREEN
0169 BRKPOINT =   45           POINTS FOR 1 BRICK MOVE
0170 ;
0171 CHSIZE   =   512          SIZE OF CHAR SET (ONLY 512 NEEDED)
0172 DMAVAL   =   62           ENABLE INSTRUCTION FETCH DMA
0173 ;                         1 LINE P/M RESOLUTION
0174 ;                         ENABLE PLAYER DMA
0175 ;                         ENABLE MISSILE DMA
0176 ;                         16Ø COLOR CLOCK PLAYFIELD DMA
0177 ;
0178 GRAVAL   =   3            SET TRIG0-TRIG3 NORMAL INPUTS
0179 ;                         ENABLE DMA TO PLAYER GRAPHICS REGS
0180 ;                         ENABLE DMA TO MISSILE GRAPHICS REGS
0181 ;
0182 IND0Ø1H  =   120          INITIAL DOG 1 HORIZ POS
0183 IND02H   =   140          INITIAL DOG 2 HORIZ POS
0184 IND03H   =   160          INITIAL DOG 3 HORIZ POS
0185 INMANH   =   80           INITIAL MAN HORIZ POS
0186 INMANV   =   175          INITIAL MAN VERT POS
0187 ;
0188 M2LSIZE  =   40           MODE 2 LINE SIZE
0189 M4LSIZE  =   40           MODE 4 LINE SIZE
0190 M7LSIZE  =   20           MODE 7 LINE SIZE
0191 ;
0192 PF0CMASK =   $00          MASK TO USE PLAYFIELD 0 COLOR
0193 PF1CMASK =   $40          MASK TO USE PLAYFIELD 1 COLOR
0194 PF2CMASK =   $80          MASK TO USE PLAYFIELD 2 COLOR
0195 PF3CMASK =   $C0          MASK TO USE PLAYFIELD 3 COLOR
0196 ;
0197 PFLINES  =   16           # OF LINES WHERE PLAY OCCURS
0198 SCRSIZ   =   640          # OF BYTES IN DISPLAY AREA
0199 ;
0200 BC       =   62           BRICK CHAR
0201 LC       =   61           MAN LIFE CHAR
0202 WC       =   63           WALL CHARACTER
0203 ;
0204 ;          SCREEN VALUES
0205 ;          -------------
0206 BORDB    =   $B0          BOTTOM BORDER LINE FOR PLAY AREA
0207 BORDL    =   $33          LEFT BORDER COL FOR PLAY AREA
0208 BORDR    =   $C6          RIGHT BORDER COL FOR PLAY AREA
0209 BORDT    =   $38          TOP BORDER LINE FOR PLAY AREA
0210 ;
0211 ;          COLORS
0212 ;          ------
0213 GREY     =   $00
0214 ORANGE   =   $20
0215 REDORG   =   $30
0216 PINK     =   $40
0217 BLUE     =   $80
0218 YELGRN   =   $D0
0219 ;
0220 COLPF2   =   $D018
0221 COLOR    =   $80
0222 WSYNC    =   $D40A
0223 ;
0224          *=  $2000
0225 ;
0226 ;          P/M RAM SPACE
0227 ;          -------------
0228 PM       *=  *+768        1ST AREA NOT USED
0229 MISL     *=  *+256        MISSILE GRAPHICS AREA
0230 PLR0     *=  *+256        PLAYER 0 GRAPHICS AREA
0231 PLR1     *=  *+256        PLAYER 1 GRAPHICS AREA
0232 PLR2     *=  *+256        PLAYER 2 GRAPHICS AREA
0233 PLR3     *=  *+256        PLAYER 3 GRAPHICS AREA
0234 ;
0235 PLEND    =   *
0236 PLDIFF   =   PLEND-MISL
0237 ;
0238 CUSCS    *=  *+512        CUSTOM CHAR SET
0239 DISP     *=  *+SCRSIZ     ACTUAL PLAY AREA
0240 FIRSTBR  =   DISP+91      ADDR OF 1ST BRICK
0241 ;
0242 WALLLINE
0243          .BYTE WC,WC,WC,WC,WC,WC,WC,WC,WC,WC
0244          .BYTE WC,WC,WC,WC,WC,WC,WC,WC,WC,WC
0245 ;
0246 ;          BRICKLAYER DISPLAY LIST
0247 ;          -----------------------
0248 DL       .BYTE BL8,BL8    BLANK SCRN TOP
0249          .BYTE BL8,BL8
0250 ;
0251          .BYTE CH7+LMS
0252          .WORD WALLLINE   SHOW WALL LINE
0253          .BYTE CH4+LMS    SET MAIN SCRN DISPLAY ADR
0254          .WORD DISP
0255 ;
0256          .BYTE CH4,CH4    DEFINE MAIN SCREEN DISPLAY AREA
0257          .BYTE CH4,CH4
0258          .BYTE CH4,CH4
0259          .BYTE CH4,CH4
0260          .BYTE CH4,CH4
0261          .BYTE CH4,CH4
0262          .BYTE CH4,CH4
0263          .BYTE CH4
0264          .BYTE CH7+LMS
0265          .WORD WALLLINE   SHOW WALL LINE
0266 ;
0267          .BYTE CH6+LMS    SET SCORE LINE ADR
0268          .WORD SLINE
0269 ;
0270          .BYTE JVB        JMP ON VERT BLANK
0271          .WORD DL         TO DISPLAY LIST START
0272 ;
0273 ;          RETURN SCREEN TO NORMAL
0274 ;          -----------------------
0275 NORMAL
0276          LDA #$00
0277          STA GRACTL
0278          LDA #$22
0279          STA SDMCTL
0280          LDA #$20
0281          STA SDLSTL
0282          LDA #$BC
0283          STA SDLSTL+1
0284 ;
0285          LDA #$E0
0286          STA CHBAS
0287 ;
0288 ;          MAIN LINE
0289 ;          ---------
0290 MAINRTN
0291          JSR INIT         INITIALIZE
0292          JSR INITPM       INIT PLAYERS
0293 ;
0294 ;          START NEW LEVEL
0295 ;          ---------------
0296 STARTLEV JSR INITBRKS     SETUP INITIAL BRICKS
0297          JSR SETNLEV      SET NEXT LEVEL
0298          JSR INITLEV      INIT LEVEL VALUES
0299 ;
0300 ;          START NEW LIFE
0301 ;          --------------
0302 STLIFE JSR INITPM         INIT PLAYERS
0303          JSR SHOWLIVS     SHOW LIVES REMAINING
0304 MAIN1   JSR CHKMANOK     DID MAN COLLIDE WITH DOG ?
0305          BNE REMMAN       YES, SO REMOVE MAN
0306          JSR MOVMAN       MOVE THE MAN
0307          LDA BMOVES       GET # OF BRICKS LEFT
0308          BNE MAIN2        MORE, SO CONTINUE
0309          JMP STARTLEV     START NEXT LEVEL
0310 ;
0311 MAIN2   DEC DOGMOVCT     CAN DOGS MOVE YET ?
```

Assembly language listing.

```
0312         BNE Q1             NO
0313         LDA DOGFREQ1       YES, RESET NORMAL DOG MOVE CNT
0314         STA DOGMOVCT
0315         LDA STRIG0         IS TRIG DOWN ?
0316         BNE Q2             NO, LEAVE NORMAL DOG MOVE CNT
0317         LDA DOGFREQ2       ELSE, SET FASTER DOG COUNT
0318         STA DOGMOVCT       AND SAVE
0319 ;
0320 Q2     JSR MOVDOG1        MOVE DOG 1
0321         JSR MOVDOG2        MOVE DOG 2
0322         JSR MOVDOG3        MOVE DOG 3
0323 Q1     JSR MDELAY         DO MAIN DELAY
0324         LDA BSNDFLAG       DID BRICK JUST MAKE SOUND ?
0325         BNE MAIN1          YES, SO DON'T DELAY
0326         JMP MAIN1
0327 ;
0328 REMMAN
0329         JSR KILLMAN        SHOW MAN'S DOWNFALL
0330         DEC LIVES          REDUCE # OF LIVES LEFT
0331         BPL STLIFE         START NEW LIFE
0332 ;
0333 ;          END OF GAME
0334 ;          -----------
0335         LDX #0             INIT INDEX
0336 SHOWOVER LDA ENDMSG,X      GET 'GAME OVER' BYTE
0337         CMP #$FF           END OF MSG ?
0338         BEQ WAITSTRT       YES, SO WAIT TO START AGAIN
0339         ORA #PF3CMASK      MAKE CHAR PLAYFIELD 3 COLOR
0340         STA SLINE+3,X      PUT MSG BYTE ON SCRN
0341         INX                PNT TO NEXT BYTE
0342         JMP SHOWOVER       SHOW NEXT BYTE
0343 ;
0344 WAITSTRT LDA STRIG0        IS TRIGGER PRESSED ?
0345         BNE WAITSTRT       NO, SO WAIT
0346         STA ATRACT         TRIG PUSHED, SO DISABLE ATTRACT
0347         JMP MAINRTN        YES, SO START AGAIN
0348 ;
0349 ;          CLEAR PM AREA
0350 ;          -------------
0351 INITPM
0352         LDA #MISL&255
0353         STA DESADR         SET START ADR LO
0354         LDA #MISL/256
0355         STA DESADR+1       SET START ADR HO
0356         LDA #PLDIFF&255
0357         STA SIZE           SET SIZE LO
0358         LDA #PLDIFF/256
0359         STA SIZE+1         SET SIZE HO
0360         LDX #0             SET FILL CHAR
0361         JSR FILL           CLEAR PM AREA
0362 ;
0363 ;          INIT PLAYERS COLORS & POSITIONS
0364 ;          ------------------------------
0365         LDX #4             SET # OF PLAYERS
0366         LDA #0
0367 INITPSIZ STA SIZEP0-1,X    SET NORMAL PLAYER SIZE
0368         DEX                MORE PLAYER SIZES TO INIT ?
0369         BNE INITPSIZ       YES
0370 ;
0371         LDA #BLUE+6        SET COLOR & LUM
0372         STA PCOLR0         SET PLAYER 0 COLOR
0373         LDA #PINK+8        SET COLOR & LUM
0374         STA PCOLR1         FOR DOG
0375         STA PCOLR2         SET DOG 2 COLOR & LUM
0376         STA PCOLR3         SET DOG 3 COLOR & LUM
0377 ;
0378         LDA #INMANH        GET MAN'S INITIAL HORIZ POS
0379         STA MANPOSH        SAVE HORIZ POS
0380         STA HPOSP0         SET IT
0381         LDA #INDOG1H       GET DOG 1'S INITIAL HORIZ POS
0382         STA DOG1POSH       SAVE HORIZ POS
0383         STA HPOSP1         SET IT
0384         LDA #INDOG2H       GET DOG 2'S INITIAL HORIZ POS
0385         STA DOG2POSH       SAVE HORIZ POS
0386         STA HPOSP2         SET IT
0387         LDA #INDOG3H       GET DOG 3'S INITIAL HORIZ POS
0388         STA DOG3POSH       SAVE HORIZ POS
0389         STA HPOSP3         SET IT
0390 ;
0391         LDA #INMANV        GET MAN'S INITIAL VERT POS
0392         STA MANPOSV        SAVE MAN'S VERT POS
0393         STA DOG1POSV       SAVE DOG 1'S VERT POS
0394         STA DOG2POSV       SAVE DOG 2'S VERT POS
0395         STA DOG3POSV       SAVE DOG 3'S VERT POS
0396 ;
0397 ;          PUT INITIAL MAN IMAGE ON SCREEN
0398 ;          -------------------------------
0399         LDA #1
0400         STA MANPICNO       PNT TO 1ST MAN IMAGE
0401         JSR PUTMAN         PUT MAN IN PLAYER AREA
0402 ;
0403 ;          PUT INITIAL DOGS ON SCREEN
0404 ;          --------------------------
0405 INITDOG LDX #8             SET # OF BYTES IN DOG IMAGE
0406 Q3     LDA DOGIMAGE-1,X   GET DOG IMAGE BYTE
0407         STA PLR1+INMANV-1,X PUT DOG BYTE IN PLAYER 1
0408         STA PLR2+INMANV-1,X PUT DOG BYTE IN PLAYER 2
0409         STA PLR3+INMANV-1,X PUT DOG BYTE IN PLAYER 3
0410         DEX                MORE BYTES IN PLAYER ?
0411         BNE Q3             YES
0412 ;
0413         LDA #PM/256        GET HI BYTE OF PM ADR
0414         STA PMBASE         SET PM ADR
0415 ;
0416         STA HITCLR         CLEAR COLLISIONS
0417 ;
0418         RTS                RET TO MAIN RTN
0419 ;
0420 ;          SETUP CHAR SET
0421 ;          --------------
0422 INIT
0423         LDA #0
0424         STA SRCADR         SET ORIG CHAR SET LO ADR
0425         LDA CHBAS
0426         STA SRCADR+1       SET ORIG CHAR SET HO ADR
0427         LDA #CUSCS&255
0428         STA DESADR         SET CUSTOM CHAR SET LO ADR
0429         LDA #CUSCS/256
0430         STA DESADR+1       SET CUSTOM CHAR SET HO ADR
0431         LDA #CHSIZE&255    GET SIZE OF CHSET LO BYTE
0432         STA SIZE           SAVE FOR MOVE
0433         LDA #CHSIZE/256    SET # OF 256 BYTE BLOCKS TO MOVE
0434         STA SIZE+1         SAVE SIZE HO BYTE
0435         JSR MOVE           MOVE CHAR SET TO RAM
0436 ;
0437 ;          REDEFINE CHARACTERS
0438 ;          -------------------
0439         LDX #24            SET # OF BYTES IN CHARS
0440 Q4     LDA LIFEPAT-1,X    GET NEW CHAR BYTE
0441         STA CUSCS+488-1,X  PUT INTO CUSTOM CHAR SET
0442         DEX                MORE BYTES ?
0443         BNE Q4             YES
0444 ;
0445         LDA #CUSCS/256     GET CUSTOM CHSET HO ADR
0446         STA CHBAS          SET NEW CHAR SET ADR
0447 ;
0448 ;          INIT SCREEN
0449 ;          -----------
0450         JSR CLSCRN         CLEAR SCREEN
0451         LDA #DL&255        SET DL LO BYTE
0452         STA SDLSTL
0453         LDA #DL/256        SET DL HI BYTE
0454         STA SDLSTL+1
0455 ;
0456 ;          SET PLAYFIELD COLORS
0457 ;          --------------------
0458         LDA #GREY+0
0459         STA COLOR4         SET BACKGROUND COLOR
0460         LDA #ORANGE+8
0461         STA COLOR0         SET PLAYFIELD 0 COLOR
0462         LDA #GREY+$0A
0463         STA COLOR1         SET PLAYFIELD 1 COLOR
0464         LDA #YELGRN+8
0465         STA COLOR2         SET PLAYFIELD 2 COLOR
0466         LDA #REDORG+8
0467         STA COLOR3         SET PLAYFIELD 3 COLOR
0468 ;
0469         LDA #DMAVAL        SET DMA CONTROL
0470         STA SDMCTL
0471         LDA #GRAVAL        SET GRAPHICS CONTROL
0472         STA GRACTL
0473 ;
0474 ;          INIT SCORE LINE
0475 ;          ---------------
0476         LDX #SLEND-SLINE SET SIZE OF SCORE LINE
0477         LDA #0             SET BLANK CHAR
0478 Q5     STA SLINE-1,X      BLANK SCORE LINE
0479         DEX                MORE TO BLANK ?
0480         BNE Q5             YES
0481 ;
0482         LDA DIGITS+0       GET ATASCII FOR '0'
0483         LDX #6             SET # OF DIGITS IN SCORE
0484 INITSCOR STA SCORE-1,X     PUT 0 SCORE DIGIT
0485         DEX                MORE DIGITS ?
0486         BNE INITSCOR       YES
0487 ;
0488         LDA DIGITS+0       GET ATASCII FOR '0'
0489         ORA #PF1CMASK      MAKE IT PLAYFIELD 1 COLOR
0490         STA LEVEL          SET LEVEL 1
0491 ;
0492         LDA #3
0493         STA LIVES          INIT # OF LIVES
0494 ;
0495 ;          DRAW WALLS
0496 ;          ----------
0497 DRAWWALL
0498         LDA #WC            GET WALL CHAR
0499         STA TEMP           SAVE FOR TRANSFER
0500         STA DISP           PUT WALL CHAR ON SCRN
0501         LDX #PFLINES       SET # OF LINES TO PUT WALL ON
0502         LDA #DISP&255
0503         STA DESADR         SAVE LO ADR
0504         LDA #DISP/256
0505         STA DESADR+1       SAVE HO ADR
0506 PNTWALL LDA #M4LSIZE-1     SET VALUE TO PNT TO LINE END
0507         JSR PUTSCRCH       PUT WALL CHAR AT LINE END
0508         DEX                MORE WALL CHARS ?
0509         BEQ Q6             NO, SO EXIT
0510         LDA #1             SET VALUE TO PNT TO NEXT LINE
0511         JSR PUTSCRCH       PUT WALL CHAR AT NEXT LINE START
0512         JMP PNTWALL        PUT NEXT WALL CHAR
0513 Q6      RTS                RET TO MAIN PROGRAM
0514 ;
0515 PUTSCRCH CLC               SETUP FOR ADD
0516         ADC DESADR         ADD TO DEST LO ADR
0517         STA DESADR         SAVE DEST LO ADR
0518         LDA DESADR+1       GET DEST HO ADR
0519         ADC #0             ADD CARRY IF ANY
0520         STA DESADR+1       SAVE DEST HO ADR
0521         LDA TEMP           GET CHAR TO PUT ON SCRN
0522         LDY #0             ZERO INDEX
0523         STA (DESADR),Y     PUT WALL CHAR ON SCRN
0524         RTS
0525 ;
```

Turn your Atari into a Ferrari.

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

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

Looks like a Ferrari.
The Indus GT is only 2.65" high. But under its front-loading front end is slimline engineering with a distinctive European-Gran flair.

Touch its LED-lit CommandPost™ function control AccuTouch™ buttons. Marvel at how responsive it makes every Atari home computer.

Drives like a Rolls.
Nestled into its soundproofed chassis is the quietest and most powerful disk drive power system money can buy. At top speed, it's virtually unhearable. Whisper quiet.

Flat out, the GT will drive your Atari track-to-track 0-39 in less than one second. And when you shift into SynchroMesh DataTransfer,™ you'll increase your Atari's baud rate an incredible 400%. (Faster than any other Atari system drive.)

And, included as standard equipment, each comes with the exclusive GT DrivingSystem™ of software programs. World-class word processing is a breeze with the GT Estate WordProcessor.™ And your dealer will describe the two additional programs that allow GT owners to accelerate their computer driving skills.

Also, the 1984 Indus GT is covered with the GT PortaCase.™ A stylish case that conveniently doubles as a 80-disk storage file.

Parks like a Beetle.
The GT's small, sleek, condensed size makes it easy to park.

And its low $449 price makes it easy to buy.

So see and test drive the incredible new 1984 Indus GT at your nearest computer dealer soon.

The drive will be well worth it.

INDUS™

The all-new 1984 Indus GT Disk Drive.

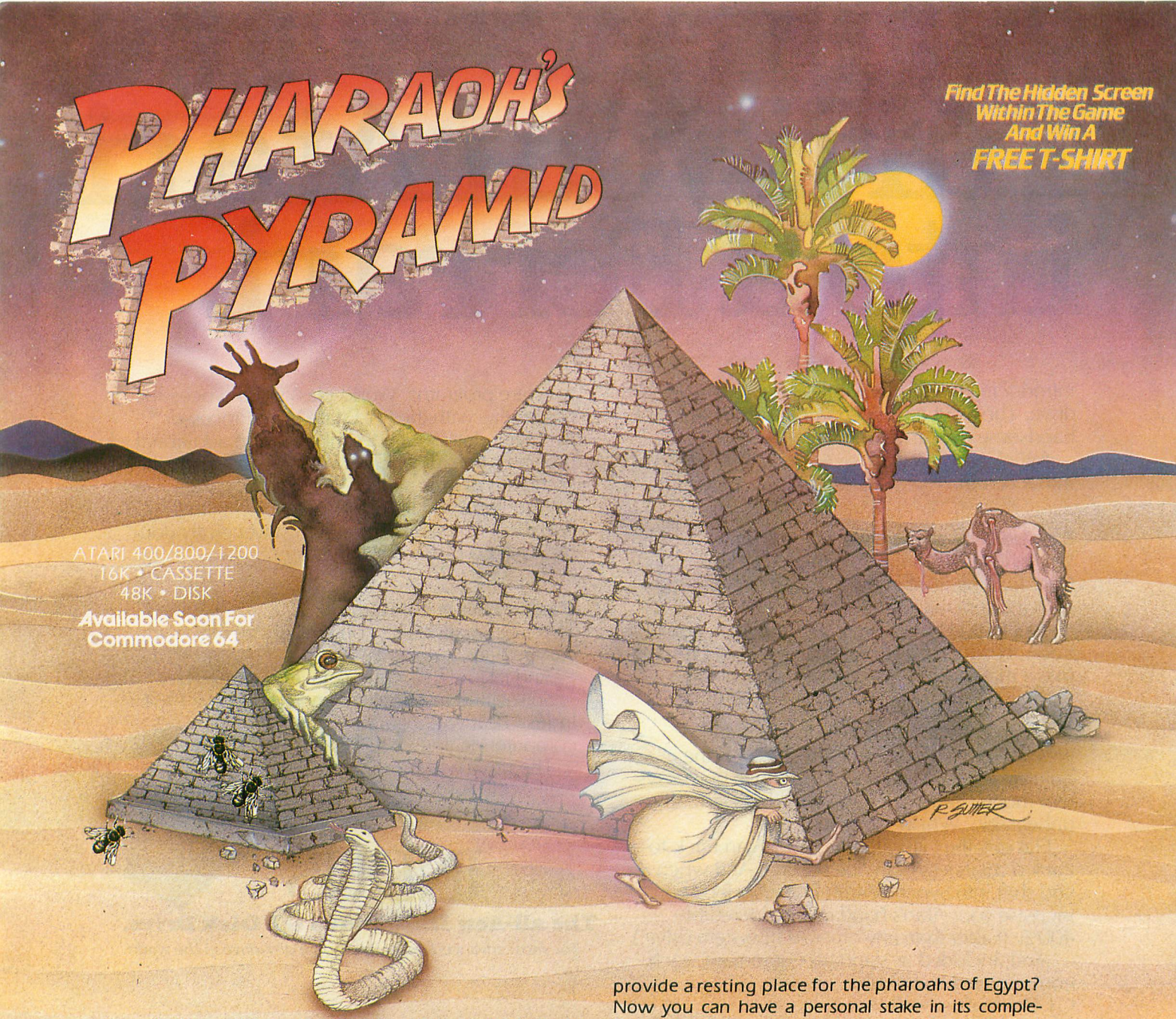
The most advanced, most handsome disk drive in the world.

For additional information, call 1-800-33-INDUS. In California, call 1-800-54-INDUS. 213/882-9600.

PHARAOH'S PYRAMID

Find The Hidden Screen Within The Game And Win A FREE T-SHIRT

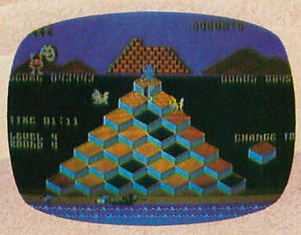
ATARI 400/800/1200
16K • CASSETTE
48K • DISK
Available Soon For Commodore 64



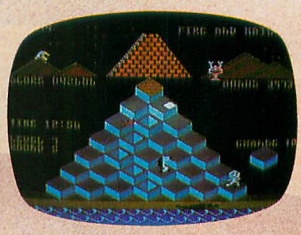
It sits on the sands of Egypt–silent, foreboding; the blue waters of the Nile sliding silently by. Pharoah's Pyramid beckons to us across millennia; mysterious, alluring. Where did the stones come from? How were they laid one on top of another? How many lives were lost under hot suns and cool moons to provide a resting place for the pharoahs of Egypt? Now you can have a personal stake in its completion. Match wits with the ancient Egyptian God, Osiris. Take advantage of gifts given you by the beautiful Isis. Fight through the plagues of Moses as brick by brick you build one of the wonders of the world. There hasn't been a greater challenge since the original Pharoah's Pyramid!

HISTORY PROVES THAT ALL GREAT GAMES HAVE THREE THINGS IN COMMON.

FAST-PACED FUN THAT LASTS



EVER INCREASING CHALLENGE



GRAPHICS THAT LIVE

PHAROAH'S PYRAMID is a fast action game with that special kind of challenge that makes you say... "This time I go all the way!"

AVAILABLE NOW FROM... Master Control Software Inc

The company that loves great games as much as you do, see your local dealer or call toll free. 1•800•624•5596

P.O. BOX 26714 • SALT LAKE CITY • UTAH • 84126

```
;         CLEAR PLAYFIELD
;         ---------------
CLSCRN
        LDA #DISP&255
        STA DESADR          SET DEST LO ADR
        LDA #DISP/256
        STA DESADR+1        SET DEST HO ADR
        LDA #SCRSIZ&255
        STA SIZE            SET SIZE OF FILL LO BYTE
        LDA #SCRSIZ/256
        STA SIZE+1          SET SIZE OF FILL HO BYTE
        LDX #0              SET FILL CHAR
        JSR FILL            CLEAR THE SCRN AREA
        RTS
;
;         INIT LEVEL VALUES
;         -----------------
INITLEV LDA LEVEL           GET CURRENT LEVEL
        AND #$0F            MAKE IT BINARY #
        TAX
        LDA #0              INIT OFFSET
Q7      DEX                 IS THIS THE LEVEL ?
        BEQ Q8              YES,SET LEVEL VALUES
        CLC                 SETUP FOR ADD
        ADC #4              PNT TO NEXT LEVEL VALUES
        JMP Q7              SEE IF THIS IS THE LEVEL
Q8      TAX                 GET OFFSET
        LDA LEVTBL,X        GET NORMAL DOG MOVES/MAN MOVE
        STA DOGFREQ1        SET IT
        LDA LEVTBL+1,X      GET DOG/MAN MOVES (PULLING WALL)
        STA DOGFREQ2        SAVE IT
        LDA LEVTBL+2,X      GET MAIN LOOP HO VALUE
        STA MDELHO          SET IT
        LDA LEVTBL+3,X      GET MAIN LOOP LO VALUE
        STA MDELLO          SET IT
;
        LDA #BMPS           GET # OF BRICKS TO MOVE
        STA BMOVES
;
        RTS                 EXIT
;
;         READ JOYSTICK & MOVE MAN
;         ------------------------
MOVMAN  LDA STICK0          READ JOYSTICK
        CMP #LEFT           MOVE LEFT ?
        BNE CHKJSR          NO, SO CHECK RIGHT
;
;
MOVMANL JSR SETMANL         SET MAN LEFT
SETHIM  DEC MANPICNO        ELSE, SET NEXT MAN IMAGE #
        BNE SHONEWM         IF VALID, SHOW NEW MAN
        LDA #5              ELSE, SET VALID MAN IMAGE
        STA MANPICNO        SAVE MAN IMAGE #
SHONEWM JSR PUTMAN          SHOW MAN
        RTS                 EXIT
;
SETMANL DEC MANPOSH         PNT TO COL TO LEFT
        LDA #BORDL          GET LEFT LIMIT
        CMP MANPOSH         AT LEFT LIMIT ?
        BEQ MOVLERR         YES, SO ERROR
        LDY MANPOSH         GET HORIZ POS
        LDX MANPOSV         GET VERT POS
        JSR CHKLRM          SEE IF MOVE IS OK
        BNE MOVLERR         SOMETHING THERE, CAN'T MOVE
;
        LDA MANPOSH         GET CURRENT MAN HORIZ POS
        STA HPOSP0          SET HARDWARE HORIZ REG
        RTS
;
MOVLERR INC MANPOSH         SET ORIG COL
        RTS                 EXIT
;
CHKJSR  CMP #RIGHT          MOVE RIGHT ?
        BEQ MOVMANR         YES, SO MOVE MAN RIGHT
;
        CMP #UP             MOVE UP ?
        BEQ MOVMANU         YES, MOVE MAN UP
;
        CMP #DOWN           MOVE DOWN ?
        BEQ MOVMAND         YES, MOVE MAN DOWN
        JMP CHKANGLS        CHECK JOYSTICK ANGLES
;
;
MOVMANR JSR SETMANR         SET MAN TO RIGHT
        JSR MOVBRICK        MOVE BRICK IF APPROPRIATE
        JMP SETHIM          SET NEW HORIZ IMAGE
;
SETMANR INC MANPOSH         SET COL TO RIGHT
        LDA #BORDR          GET # OF RIGHT BORDER
        CMP MANPOSH         AT RIGHT BORDER ?
        BEQ MOVRERR         YES, SO ERROR
;
        LDX MANPOSV         GET MAN VERT POS
        LDA MANPOSH         GET HORIZ POS
        CLC                 SETUP FOR ADD
        ADC #7              PNT TO RIGHT SIDE OF PLAYER
        TAY
        JSR CHKLRM          SEE IF MOVE OK
        BNE MOVRERR         CHAR THERE, SO ERROR
;
;
        LDA MANPOSH         GET CURRENT MAN HORIZ POS
        STA HPOSP0          SET HARDWARE HORIZ REG
        RTS
MOVRERR DEC MANPOSH         SET ORIGINAL MAN HORIZ POS
        RTS                 EXIT
;
```

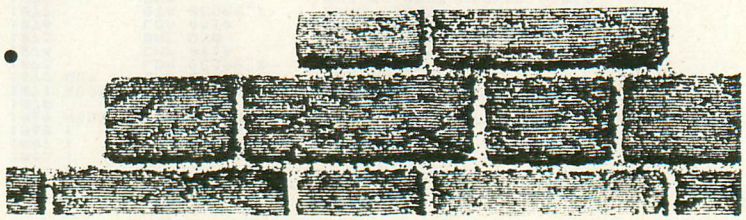
```
MOVMANU DEC MANPOSV         SET LINE ABOVE
        LDA #BORDT          GET # OF TOP BORDER
        CMP MANPOSV         AT TOP BORDER ?
        BEQ MOVUERR         YES, SO RESET PNTR
;
        LDY MANPOSH         GET HORIZ POS
        LDX MANPOSV         GET VERT POS
        JSR CHKUDM          CHK UP DOWN MOVE
        BEQ SETVIM          BLANK, SO MOVE OK
MOVUERR INC MANPOSV         SET ORIGINAL POS
SETVIM  INC MANPICNO        SET NEXT VERT IMAGE
        LDA #10             GET 1 BEYOND MAX IMAGE #
        CMP MANPICNO        IS MAN IMAGE # VALID ?
        BNE Q9              YES
        LDA #6              NO, SO SET VALID IMAGE #
        STA MANPICNO        SAVE NEW MAN IMAGE #
Q9      LDX MANPOSV         GET MAN POSITION FOR INDEX
        LDA #0
        STA PLR0+8,X        BLANK BYTES AROUND OLD MAN IMAGE
        STA PLR0-1,X
        JSR PUTMAN          PUT NEW IMAGE IN PLAYER AREA
        RTS                 EXIT
;
;
MOVMAND INC MANPOSV         SET LINE BELOW
        LDA #BORDB          GET # OF BOTTOM BORDER
        CMP MANPOSV         AT BOTTOM BORDER ?
        BEQ MOVDERR         YES, SO RESET PNTR
;
        LDY MANPOSH         SET HORIZ POS
        LDA MANPOSV         GET VERT POS
        CLC                 PREPARE FOR ADD
        ADC #7              PNT TO LINE BELOW
        TAX                 SAVE VERT POS
        JSR CHKUDM          CHK UP DOWN MOVE
        BEQ SETVIM          BLANK, SO MOVE OK
;
MOVDERR DEC MANPOSV         YES, SO SET ORIGINAL LINE #
        JMP SETVIM          SET NEW VERT IMAGE
;
CHKANGLS CMP #UPLEFT        MOVE UP LEFT ?
        BEQ MOVMANUL        YES
;
        CMP #UPRIGHT        MOVE UP RIGHT ?
        BEQ MOVMANUR        YES
;
        CMP #DOWNLEFT       MOVE DOWN LEFT ?
        BEQ MOVMANDL        YES
;
        CMP #DOWNRGT        MOVE DOWN RIGHT ?
        BEQ MOVMANDR        YES
;
        RTS                 EXIT TO MAIN LINE
;
MOVMANUL JSR SETMANL        SET MAN LEFT
        JSR MOVMANU         MOVE MAN UP
        RTS
;
MOVMANUR JSR SETMANR        SET MAN RIGHT
        JSR MOVMANU         MOVE MAN UP
        RTS
;
MOVMANDL JSR SETMANL        SET MAN LEFT
        JSR MOVMAND         MOVE MAN DOWN
        RTS
;
MOVMANDR JSR SETMANR        SET MAN RIGHT
        JSR MOVMAND         MOVE MAN DOWN
        RTS
;
;         CHECK UP DOWN MOVE
;         ------------------
CHKUDM
        STY TEMP            SAV HORIZ POS
        STX TEMP+1          SAV VERT POS
        JSR GETSCRN         GET SCRN CHAR
        BNE Q10             CHAR THERE, SO EXIT
        LDX TEMP+1          GET VERT POS
        LDA TEMP            GET HORIZ POS
        CLC                 SETUP FOR ADD
        ADC #4              PNT TO MIDDLE OF PLAYER
        TAY
        JSR GETSCRN         GET SCRN CHAR
        BNE Q10             CHAR THERE, SO EXIT
        LDX TEMP+1          GET VERT POS
        LDA TEMP            GET HORIZ POS
        CLC                 SETUP FOR ADD
        ADC #6              PNT TO RIGHT SIDE OF PLAYER
        TAY
        JSR GETSCRN         GET SCRN CHAR
Q10     RTS
;
;         CHECK LEFT RIGHT MOVE
;         ---------------------
CHKLRM  STY TEMP            SAVE HORIZ POS
        STX TEMP+1          SAVE VERT POS
        JSR GETSCRN         GET SCRN CHAR
        BNE Q11             SOMETHING THERE, SO EXIT
        LDY TEMP            GET HORIZ POS
        LDA TEMP+1          GET VERT POS
        CLC                 PREPARE FOR ADD
        ADC #7              PNT TO PLAYER BOTTOM LINE
        TAX
        JSR GETSCRN         GET SCRN CHAR
Q11     RTS                 EXIT
;
```

```
;         PUT MAN IMAGE IN PLAYER AREA
;         ----------------------------
PUTMAN LDX MANPICNO     GET CURRENT MAN IMAGE #
       LDA #-8
Q12    CLC              CALC
       ADC #8           IMAGE
       DEX              OFFSET
       BNE Q12
;
       TAY              SET IMAGE OFFSET
       LDA #8           SET # OF BYTES IN IMAGE
       STA SIZE         AND SAVE
       LDX MANPOSV      GET CURRENT VERT POS
;
Q13    LDA MANH1,Y      GET MAN IMAGE BYTE
       STA PLR0,X       PUT IN PLAYER AREA
       INX              PNT TO NEXT PLAYER AREA BYTE
       INY              PNT TO NEXT IMAGE BYTE
       DEC SIZE         MORE TO MOVE ?
       BNE Q13          YES, SO MOVE THEM
       RTS
;
;         GET SCREEN CHAR
;         ---------------
;         ENTER WITH:
;
;         REG X = SCAN LINE #
;         REG Y = COLOR CLOCK #
;
GETSCRN TXA              GET LINE #
       SEC              SETUP FOR SUBT
       SBC #$38         PNT TO PLAY AREA
       LSR A            DIVIDE LINE # BY 8
       LSR A            # OF LINES IN CHAR
       LSR A
       TAX              AND SAVE
       TYA              GET COL #
       SEC              SETUP FOR SUBT
       SBC #$30         COMPENSATE FOR SCRN LEFT EDGE
       LSR A            DIVIDE COL BY 4
       LSR A            # OF COLS IN CHAR
       TAY              AND SAVE
       LDA #DISP&255    GET PLAY AREA START (LO)
       STA DESADR       ; & SAVE
       LDA #DISP/256    GET PLAY AREA START (HO)
       STA DESADR+1     ; & SAVE
Q14    DEX              IS LINE # 0 ?
       BMI Q15          YES, SO ADD COL VALUE
       CLC              SETUP FOR ADD
       LDA DESADR       GET PLAY AREA ADR (LO)
       ADC #M4LSIZE     PNT TO NEXT LINE
       STA DESADR       ; & SAVE
       LDA DESADR+1     GET PLAY AREA ADR (HO)
       ADC #0           ADD CARRY IF ANY
       STA DESADR+1     ; & SAVE
       JMP Q14          SEE IF LINES ACCOUNTED FOR
;
Q15    TYA              GET COL #
       CLC              SETUP FOR ADD
       ADC DESADR       ADD COL TO ADR (LO)
       STA DESADR       ; & SAVE
       LDA #0
       ADC DESADR+1     ADD CARRY IF ANY TO HO ADR
       STA DESADR+1     ; & SAVE
       LDY #0           CLEAR INDEX
       LDA (DESADR),Y   GET SCREEN CHAR
       RTS              ELSE, EXIT
;
;         MOVE DOG 1
;         ----------
;         DOG 1 ALIGNS ITSELF ON MAN'S HORIZ AXIS
;         AND ATTACKS ON VERT AXIS
;
MOVDOG1
       LDA #0
       STA DOGNO        SET DOG 1 OFFSET
       JSR MOVDOGH      MOVE DOG HORIZ
       BEQ CHKDOG1V     NO MOVE, SO TRY VERTICAL
       RTS              EXIT
CHKDOG1V JSR CHKDOGV    MOVE DOG VERTICAL
       RTS              EXIT
;
;         MOVE DOG 2
;         ----------
;         DOG 2 ALIGNS ITSELF ON MAN'S VERT AXIS
;         AND ATTACKS ON HORIZ AXIS
;
MOVDOG2
       LDA #1
       STA DOGNO        SET DOG 2 OFFSET
       JSR CHKDOGV      MOVE DOG VERT IF POSSIBLE
       BEQ CHKDOG2H     NO MOVE, SO TRY HORIZ
       RTS              ELSE, EXIT
CHKDOG2H
       JSR MOVDOGH      MOVE DOG HORIZ
       RTS              EXIT
;
;         MOVE DOG 3
;         ----------
;         DOG 3 MOVES ON VERT & HORIZ PLANES ON SAME TURN
;         TO REACH THE MAN
;
MOVDOG3
       LDA #2
       STA DOGNO        SET DOG 3 OFFSET
       JSR MOVDOGH      MOVE DOG HORIZ
       JSR CHKDOGV      MOVE DOG 3 VERT IF POSSIBLE
       RTS              EXIT
;
;         MOVE DOG VERTICALLY
;         -------------------
CHKDOGV
       LDA #PLR1&255    GET 1ST DOG PLAY ADR (LO)
       STA DESADR       ; & SAVE
       LDA #PLR1/256    GET 1ST DOG PLAY ADR (HO)
       STA DESADR+1     ; & SAVE
       LDY DOGNO        GET CURRENT DOG #
Q16    BEQ Q17          MATCH, SO START MOVE
       INC DESADR+1     ELSE, SET NEXT DOG PLAY AREA
       DEY              SET Z FLAG IF MATCH
       JMP Q16          SEE IF MATCH NOW
;
Q17    LDX DOGNO        GET CURRENT DOG #
       LDA DOGPOSV,X    GET DOG VERT POS
       CMP MANPOSV      COMPARE WITH MAN VERT POS
       BNE Q18          NOT SAME, SO MOVE VERTICAL
       RTS              SAME, SO EXIT
Q18    BCS MOVDOGU      MAN ABOVE, MOVE DOG UP
MOVDOGD LDA DESADR
       PHA              SAVE DOG PLAY AREA (LO)
       LDA DESADR+1
       PHA              SAVE DOG PLAY AREA (HO)
       LDA DOGPOSH,X    GET DOG HORIZ POS
       TAY
       LDA DOGPOSV,X    GET DOG VERT POS
       CLC              SETUP ADD
       ADC #8           PNT TO LINE BELOW
       TAX
       JSR CHKUDM       CHECK FOR BRICKS
       BEQ Q19          NO BRICKS, SO CONTINUE
       PLA
       PLA              CLEAR STACK
       LDA #0           ELSE, SET Z FLAG(NO MOVE)
       RTS              AND EXIT
Q19    LDX DOGNO        GET CURRENT DOG #
       PLA              GET DOG PLAY ADR (HO)
       STA DESADR+1
       PLA              GET DOG PLAY ADR (LO)
       STA DESADR
       LDY DOGPOSV,X    GET LINE #
       STA (DESADR),Y   BLANK AROUND OLD DOG
       INC DOGPOSV,X    PNT TO NEW LINE
       INY
SHONEWD LDX #0          SET SRCE INDEX
       LDA #8           SET # OF BYTES IN DOG IMAGE
       STA SIZE         SAVE # OF BYTES
Q20    LDA DOGIMAGE,X   GET DOG IMAGE BYTE
       STA (DESADR),Y   PUT IN DOG PLAY AREA
       INX              PNT TO NEXT PLAYER AREA BYTE
       INY              PNT TO NEXT DOG IMAGE BYTE
       DEC SIZE         MORE BYTES TO MOVE ?
       BNE Q20          YES
       LDA #1           NO, SET NZ FLAG(MOVE OCCURRED)
       RTS
;
MOVDOGU LDX DOGNO       GET CURRENT DOG #
       LDA DESADR
       PHA              SAVE DOG PLAY AREA (LO)
       LDA DESADR+1
       PHA              SAVE DOG PLAY AREA (HO)
       LDA DOGPOSH,X    GET DOG HORIZ POS
       TAY
       LDA DOGPOSV,X    GET DOG VERT POS
       TAX
       DEX              PNT TO LINE ABOVE
       JSR CHKUDM       SEE IF ANY BRICKS
       BEQ Q21          NO BRICKS, SO OK
       PLA
       PLA              CLEAR STACK
       LDA #0           SET Z FLAG (NO MOVE)
       RTS
Q21    LDX DOGNO        GET CURRENT DOG #
       PLA              GET DOG PLAY ADR (HO)
       STA DESADR+1
       PLA              GET DOG PLAY ADR (LO)
       STA DESADR
       DEC DOGPOSV,X    PNT TO LINE ABOVE
       LDA DOGPOSV,X    GET NEW LINE #
       PHA              SAVE IT
       CLC              SETUP ADD
       ADC #8           PNT TO END OF DOG
       TAY              SET INDEX REG
       LDA #0           SET VALUE TO BLANK DOG CHAR
       STA (DESADR),Y   BLANK AROUND OLD DOG
       PLA              GET NEW LINE #
       TAY
       JMP SHONEWD      SHOW NEW DOG IMAGE
;
;         MOVE DOG HORIZONTALLY
;         ---------------------
MOVDOGH
       LDX DOGNO        GET CURRENT DOG #
       LDA DOGPOSH,X    GET HORIZ POS
       CMP MANPOSH      COMPARE WITH MAN'S HORIZ POS
       BNE Q22          NOT SAME, SO MOVE
       RTS              ELSE, EXIT
;
Q22    BCS MOVDOGL      MAN TO LEFT, SO MOVE LEFT
MOVDOGR LDA DOGPOSH,X   GET DOG HORIZ POS
       CLC              SETUP ADD
       ADC #8           PNT TO DOG RIGHT SIDE
       TAY              SAVE IN INDEX
       JSR CHKBRKH      CHECK FOR BRICKS THERE
       BEQ Q23          NO CHAR SO CONTINUE
       RTS              CAN'T MOVE SO EXIT
Q23    INC DOGPOSH,X    SET COL TO RIGHT
```

```
0955 SAVDOSH LDA DOSPOSH,X   GET NEW COL
0956         STA HPOSP1,X    SET HARDWARE HORIZ POS
0957         RTS             EXIT
0958 ;
0959 MOVDOSL LDY DOSPOSH,X   GET DOG HORIZ POS
0960         DEY             SET COL TO LEFT
0961         JSR CHKBRKH     CHECK FOR BRICKS
0962         BEQ Q24         NO BRICKS, SO CONTINUE
0963         RTS             ELSE, EXIT
0964 Q24    DEC DOSPOSH,X   SET COL TO LEFT
0965         JMP SAVDOSH     SET DOG HORIZ POS
0966 ;
0967 ;         CHECK FOR BRICKS (HORIZ)
0968 ;         ------------------------
0969 CHKBRKH STY TEMP+2      SAVE HORIZ POS TO CHECK
0970         LDA DOSPOSV,X   GET DOG VERT POS
0971         TAX             PUT IN INDEX REG
0972         JSR CHKLRM      CHK IF MOVE OK
0973         PHP             SAVE Z FLAG
0974         LDX DOGNO       GET CURRENT DOG #
0975         PLP             RESTORE Z FLAG
0976         RTS             ELSE, CAN'T MOVE
0977 ;
0978 ;         MOVE BLOCK OF DATA
0979 ;         ------------------
0980 ;         ENTER WITH:
0981 ;             LOC $00,01 = DEST
0982 ;             LOC $02,03 = SRCE
0983 ;             LOC $04,05 = SIZE
0984 ;
0985 MOVE
0986         LDY #0
0987 Q25    LDA (SRCADR),Y  GET SRCE CHAR
0988         STA (DESADR),Y  MOVE TO DEST
0989         JSR BUMPSRC     PNT TO NEXT SRCE ADR
0990         JSR BUMPDES     PNT TO NEXT DEST ADR
0991         JSR DECSIZE     MORE TO MOVE ?
0992         BNE Q25         YES, SO MOVE THEM
0993         RTS             NO, SO EXIT
0994 ;
0995 ;         FILL BLOCK OF MEMORY
0996 ;         --------------------
0997 ;         ENTER WITH:
0998 ;             LOC $00,01 = DEST
0999 ;             LOC $04,05 = SIZE
1000 ;             REG X      = FILL CHAR
1001 ;
1002 FILL
1003         TXA             GET FILL CHAR
1004         LDY #0
1005         STA (DESADR),Y  PUT CHAR IN MEMORY
1006         JSR BUMPDES     PNT TO NEXT DEST ADR
1007         JSR DECSIZE     MORE TO FILL ?
1008         BNE FILL        YES, SO FILL THEM
1009         RTS             NO, SO EXIT
1010 ;
1011 BUMPSRC
1012         CLC             SETUP FOR ADD
1013         LDA SRCADR      GET LO SRCE ADR
1014         ADC #1          ADJUST LO BYTE
1015         STA SRCADR      SAVE LO SRCE ADR
1016         LDA SRCADR+1    GET HO SRCE ADR
1017         ADC #0          ADD CARRY IF ANY
1018         STA SRCADR+1    SAVE HO SRCE ADR
1019 ;
1020         RTS
1021 ;
1022 BUMPDES
1023         CLC             SETUP FOR ADD
1024         LDA DESADR      GET LO DEST ADR
1025         ADC #1          ADJUST LO BYTE
1026         STA DESADR      SAVE LO DEST ADR
1027         LDA DESADR+1    GET HO DEST ADR
1028         ADC #0          ADD CARRY IF ANY
1029         STA DESADR+1    SAVE HO DEST ADR
1030         RTS
1031 ;
1032 DECSIZE
1033         SEC             PREPARE FOR SUBTRACT
1034         LDA SIZE        GET LO BYTE OF SIZE
1035         SBC #1          ADJUST LO SIZE BYTE
1036         STA SIZE        SAVE LO SIZE
1037         BNE Q26         LO BYTE NOT 0, SO CONTINUE
1038         LDA SIZE+1      GET HO SIZE BYTE
1039         BNE Q27         NOT 0, SO CONTINUE
1040         RTS             SIZE NOW 0, EXIT WITH Z FLAG
1041 Q26    LDA SIZE+1      GET HO BYTE OF SIZE
1042 Q27    SBC #0          ADJUST HO SIZE BYTE
1043         STA SIZE+1      SAVE HO SIZE BYTE
1044         LDA #1          SIZE NOT 0, SET NZ FLAG
1045         RTS             EXIT
1046 ;
1047 ;         MAIN DELAY
1048 ;         ----------
1049 MDELAY LDX MDELHO      GET DELAY HO VALUE
1050         LDY MDELLO      GET DELAY LO VALUE
1051 Q28    DEY             LO COUNT DONE ?
1052         BNE Q28         NO
1053         DEX             LO COUNT DONE ?
1054         BNE Q28         NO
1055         RTS             YES, SO EXIT
1056 ;
1057 ;         CHECK IF MAN COLLIDED WITH DOG
1058 ;         ------------------------------
1059 CHKMANOK
1060         LDA P0PL        GET MAN/DOG COLLISION INDICATOR
1061         AND #$0E        GET RID OF UNUSED BITS
1062         RTS             EXIT WITH Z FLAG SET IF NO HITS
1063 ;
1064 ;         UPDATE THE SCORE
1065 ;         ----------------
1066 ;
1067 ;         REG Y=VALUE TO ADD TO SCORE
1068 ;
1069 ADDSCORE LDA SCORE+1    GET 10,000 UNIT VALUE
1070         STA TEMP        ; & SAVE
1071 ADDSCOR1 LDX #5         SET TO PNT TO LO DIGIT
1072 Q29      INC SCORE,X     INC DIGIT
1073         LDA SCORE,X     GET DIGIT
1074         CMP #$1A        PAST '9' ?
1075         BNE Q30         NO, SO CONTINUE ADDING
1076         LDA #$10        YES, SO SET ATASCII 0
1077         STA SCORE,X     SAVE DIGIT
1078         DEX             PNT TO NEXT HIGHER DIGIT
1079         BPL Q29         IF WITHIN 6 DIGITS, CONTINUE
1080 Q30    DEY             MORE TO ADD ?
1081         BNE ADDSCOR1    YES
1082 ;
1083         LDA TEMP        GET ORIGINAL 10,000 UNIT VALUE
1084         CMP SCORE+1     HAS ANOTHER 10,000 BEEN REACHED?
1085         BEQ Q31         NO, SO EXIT
1086         LDA LIVES       YES, SO GET # OF LIVES
1087         CMP #4          ALREADY HAVE MAXIMUM ?
1088         BEQ Q31         YES
1089         INC LIVES       NO, SO ADD A LIFE
1090         JSR SHOWLIVS    SHOW EXTRA LIFE ON SCRN
1091 Q31      RTS             ELSE, EXIT
1092 ;
1093 ;         SHOW LIVES REMAINING
1094 ;         --------------------
1095 SHOWLIVS LDY #4          SET MAX # OF LIVES
1096         LDA #0
1097 Q32      STA SLINE+8,Y   BLANK SCRN LIVES
1098         DEY             MORE TO BLANK ?
1099         BNE Q32         YES
1100 ;
1101         LDY LIVES       GET # OF LIVES
1102         BEQ Q33         NO LIVES LEFT, SO EXIT
1103         LDA #LC+PF2CMASK GET LIFE CHAR
1104 Q34      STA SLINE+8,Y   PUT LIFE CHAR ON SCRN
1105         DEY             MORE TO SHOW ?
1106         BNE Q34         YES
1107 Q33      RTS
1108 ;
1109 ;         SET NEXT LEVEL
1110 ;         --------------
1111 SETNLEV LDA LEVEL       GET CURRENT LEVEL
1112         AND #$0F        MAKE IT BINARY
1113         CMP #7          HIGHEST LEVEL ?
1114         BEQ Q35         YES
1115 ;
1116         INC LEVEL       NO, SO SET NEXT LEVEL
1117 Q35      LDA COLOR1      GET ORIG PLAYFIELD 1 COLOR
1118         PHA             SAVE ON STACK
1119         LDA #5          SET # OF TIMES TO LOOP
1120         STA TEMP
1121 Q36      LDA #$REY+12    SET COLOR & LUMINANCE
1122         STA COLOR1
1123 ;
1124         LDA #$80
1125         STA AUDF1       SET FREQ
1126         LDA #$A6
1127         STA AUDC1       SET DISTORTION & VOLUME
1128         LDY #0          SET OUTER LOOP VAL
1129         LDX #0          SET INNER LOOP VAL
1130         JSR WAIT
1131 ;
1132         LDA #REDORG+6   SET NEW COLOR & LUMINANCE
1133         STA COLOR1
1134 ;
1135         LDA #$C0
1136         STA AUDF1       SET FREQ
1137         LDA #$A6
1138         STA AUDC1       SET DISTORTION & VOLUME
1139         LDY #0          SET OUTER LOOP VAL
1140         LDX #0          SET INNER LOOP VAL
1141         JSR WAIT
1142 ;
1143         DEC TEMP        CHANGE COLORS AGAIN ?
1144         BNE Q36         YES
1145 ;
1146         PLA             GET ORIG PLAYFIELD 1 COLOR
1147         STA COLOR1      RESTORE IT
1148         LDA #0
1149         STA AUDC1       SET 0 VOLUME
1150         RTS
1151 ;
1152 WAIT     TXA             SAVE INNER LOOP VAL
1153 Q37      TAX             GET INNER LOOP VAL
1154 Q38      DEX             INNER LOOP DONE ?
1155         BNE Q38         NO
1156         DEY             OUTER LOOP DONE ?
1157         BNE Q37         NO
1158         RTS
1159 ;
1160 ;         KILL MAN
1161 ;         --------
1162 KILLMAN
1163         LDA #9          SET INITIAL FREQUENCY
1164         STA TEMP        SAVE FREQ
1165         STA AUDF1       SET FREQUENCY
1166         LDA #$A0        SET INITIAL VOLUME
1167         STA VOLUME      SAVE FOR LATER
1168         STA AUDC1       SET HARDWARE DIST & VOL
1169         LDA #5          SET # OF TIMES TO DO COLOR CHANGE
```

```
1170 Q39     PHA               SAVE ON STACK
1171         LDX #$FF          SET INITIAL COLOR
1172 Q40     LDY #$FF          SET DELAY VALUE
1173         STX PCOLR0        SET MAN COLOR
1174 Q41     DEY               DELAY
1175         BNE Q41
1176         LDA VOLUME        GET VOLUME
1177         SEC               SETUP SUBTRACT
1178         SBC #3            SET NEW VOLUME
1179         STA VOLUME        ; & SAVE
1180         STA AUDC1         SET HARDWARE DIST & VOL
1181         DEX               MORE COLORS ?
1182         BNE Q40           YES, SO SHOW THEM
1183         INC TEMP          SET NEW FREQ
1184         LDA TEMP          GET IT
1185         STA AUDF1         SET HARDWARE FREQ
1186         PLA               GET COLOR CHANGE COUNT
1187         SEC               SETUP FOR SUBTRACT
1188         SBC #1            SHOW COLOR CHANGE AGAIN ?
1189         BNE Q39           YES
1190         STA AUDC1         SET 0 VOLUME
1191         RTS
1192 ;
1193 ;        SETUP INITIAL BRICKS
1194 ;        --------------------
1195 INITBRKS JSR CLSCRN       CLEAR THE PLAY AREA
1196         JSR DRAWWALL      DRAW LEFT/RIGHT WALL
1197         LDA #$BC          GET BRICK CHAR
1198         STA TEMP          SAVE IT
1199         STA DISP+91       PUT BRICK ON SCREEN
1200         LDX #9            SET # OF BRICKS TO PUT ON SCRN
1201         LDA #FIRSTBR&255  GET LO ADR OF 1ST BRICK
1202         STA DESADR        SAVE IT
1203         LDA #FIRSTBR/256  GET HO ADR OF 1ST BRICK
1204         STA DESADR+1      SAVE IT
1205 Q42     LDA #M4LSIZE      SET AMT TO GET TO NEXT LINE
1206         JSR PUTSCRCH      PUT BRICK ON SCRN
1207         DEX               MORE BRICKS ?
1208         BNE Q42           YES
1209         RTS               EXIT
1210 ;
1211 ;        MOVE BRICK CHAR
1212 ;        ---------------
1213 MOVBRICK LDA #0
1214         STA BSNDFLAG      CLEAR BRICK SOUND FLAG
1215         LDA STRIG0        IS TRIG DOWN ?
1216         BNE Q43           NO, SO EXIT
1217 ;
1218         LDA #0            TRIG DOWN, SO
1219         STA ATRACT        DISABLE ATTRACT MODE
1220 ;
1221         LDA MANPOSH       GET MAN HORIZ POS
1222         AND #3            IS MAN AT START OF CHAR BYTE ?
1223         BNE Q43           NO, SO EXIT
1224         LDA MANPOSH       GET MAN HORIZ POS
1225         SEC               PREPARE FOR SUBT
1226         SBC #8            PNT TO CHAR BEFORE MAN
1227         TAY
1228         STA TEMP+3        SAVE IN CASE ITS NEEDED
1229         LDX MANPOSV       GET MAN VERT POS
1230         JSR GETSCRN       GET CHAR THERE
1231         CMP #BC           IS IT A BRICK ?
1232         BEQ Q44           YES, SO MOVE BRICK
1233 ;
1234         LDA MANPOSV       GET VERT POS
1235         CLC               SETUP ADD
1236         ADC #8            PNT 1 CHAR DOWN
1237         TAX               SET VERT POS
1238         LDY TEMP+3        GET HORIZ POS
1239         JSR GETSCRN       GET SCRN CHAR
1240         CMP #BC           BRICK ?
1241         BNE Q43           NO, SO EXIT
1242 Q44     LDY #0            SET INDEX
1243         LDA #0            SET BLANK CHAR
1244         STA (DESADR),Y    BLANK OLD BRICK CHAR
1245         LDA #BC           GET BRICK CHAR
1246         LDY #1            SET INDEX FOR CHAR TO RIGHT
1247         STA (DESADR),Y    PUT BRICK AT NEW LOC
1248         LDY #BRKPOINT     GET POINTS FOR BRICK MOVE
1249         JSR ADDSCORE      ADD TO SCORE
1250         DEC BMOVES        REDUCE # OF BRICKS TO MOVE
1251         JSR BRKSND        MAKE BRICK SOUND
1252 Q43     RTS
1253 ;
1254 ;        MAKE BRICK SOUND
1255 ;        ----------------
1256 BRKSND  LDA #$70          SET INITIAL VOL,DISTORTION
1257         STA VOLUME
1258         LDA #$08          SET FREQUENCY
1259         STA AUDF1
1260 Q45     LDA VOLUME
1261         STA AUDC1         SET VOL & DISTORTION
1262         LDY #5            SET OUTER LOOP COUNT
1263 Q46     LDX #$40          SET INNER LOOP COUNT
1264 Q47     DEX               INNER LOOP DONE ?
1265         BNE Q47           NO
1266         DEY               OUTER LOOP DONE ?
1267         BNE Q46           NO
1268         DEC VOLUME        MORE SOUND ?
1269         LDA VOLUME
1270         CMP #$58
1271         BNE Q45           YES
1272         LDA #1
1273         STA BSNDFLAG      BRICK MADE SOUND,SO SET FLAG
1274         RTS
1275 ;
```

```
1276 ENDMSG .BYTE 39,33,45,37,0,47,54,37,50,255 ; "GAME OVER"
1277 ;
1278 ;        SCORE LINE
1279 ;        ----------
1280 SLINE   .BYTE 0
1281 LEVEL   .BYTE 0,0,0,0,0,0,0,0,0,0,0,0,0,0
1282 SCORE   .BYTE 0,0,0,0,0,0,0,0
1283 ;
1284 SLEND   =   *
1285 ;
1286 DIGITS .BYTE 16,17,18,19,20,21,22,23,24,25
1287 ;
1288 LIFEPAT .BYTE $80,$90,$78,$3C,$1A,$38,$48,$48
1289 ;
1290 BRICKPAT .BYTE $AA,$82,$82,$AA,$AA,$82,$82,$AA
1291 ;
1292 WALLPAT .BYTE $FF,$99,$99,$99,$99,$99,$99,$FF
1293 ;
1294 ;        HORIZ MOVING MAN IMAGES
1295 ;        -----------------------
1296 MANH1   .BYTE $10,$10,$38,$7C,$BA,$48,$84,$82
1297 MANH2   .BYTE $10,$10,$38,$38,$78,$AC,$48,$44
1298 MANH3   .BYTE $10,$10,$38,$38,$38,$28,$28,$10
1299 MANH4   .BYTE $10,$10,$38,$7C,$BA,$38,$10,$28
1300 MANH5   .BYTE $10,$10,$38,$38,$7C,$38,$48,$48
1301 ;
1302 ;        VERTICAL MOVING MAN IMAGES
1303 ;        --------------------------
1304         .BYTE $80,$90,$78,$3C,$1A,$38,$48,$48
1305         .BYTE $00,$10,$FC,$3A,$18,$24,$24,$00
1306         .BYTE $00,$10,$FC,$3A,$18,$24,$24,$00
1307         .BYTE $02,$12,$3C,$78,$B0,$38,$24,$22
1308 ;
1309 ;        DOG IMAGE
1310 ;        ---------
1311 DOGIMAGE .BYTE $66,$95,$1C,$22,$36,$3E,$14,$36
1312 ;
1313 ;        LEVEL CHARACTERISTICS TABLE
1314 ;        ---------------------------
1315 ;        EACH ENTRY CONSISTS OF:
1316 ;
1317 ;        E1,E2,E3,E4
1318 ;
1319 ;        E1 = NORMAL DOG MOVES / MAN MOVE
1320 ;        E2 = MAN PULLING WALL DOG MOVES/MAN MOVE
1321 ;        E3 = OUTER LOOP DELAY VALUE (MAIN LOOP)
1322 ;        E4 = INNER LOOP DELAY VALUE (MAIN LOOP)
1323 ;
1324 LEVTBL .BYTE 8,8,12,0  LEVEL 1
1325         .BYTE 3,3,12,$80 LEVEL 2
1326         .BYTE 3,2,11,0  LEVEL 3
1327         .BYTE 2,2,11,0  LEVEL 4
1328         .BYTE 2,1,11,80 LEVEL 5
1329         .BYTE 2,1,11,$80 LEVEL 6
1330         .BYTE 2,1,10,0  LEVEL 7
1331 ;
1332 BMOVES =*  *+1            # OF BRICKS TO BE MOVED
1333 BSNDFLAG *= *+1           0 IF NO BRICK SOUND MADE
1334 ;                       1 IF BRICK SOUND MADE
1335 LIVES  =*  *+1            # OF LIVES REMAINING
1336 MDELHO =*  *+1            MAIN DELAY HO VALUE
1337 MDELLO =*  *+1            MAIN DELAY LO VALUE
1338 TEMP   =*  *+4            TEMPORARY WORK STORAGE
1339 VOLUME =*  *+1            VOLUME FOR SOUNDS
1340 ;
1341 ;        MAN VARIABLES
1342 ;        -------------
1343 MANPICNO *= *+1           MAN'S CURRENT IMAGE #
1344 MANPOSH *= *+1           MAN'S CURRENT HORIZ POS
1345 MANPOSV *= *+1           MAN'S CURRENT VERT POS
1346 ;
1347 ;        DOG VARIABLES
1348 ;        -------------
1349 DOGPOSH =  *
1350 DOG1POSH *= *+1          DOG 1 HORIZ POS
1351 DOG2POSH *= *+1          DOG 2 HORIZ POS
1352 DOG3POSH *= *+1          DOG 3 HORIZ POS
1353 ;
1354 DOGPOSV =  *
1355 DOG1POSV *= *+1          DOG 1 VERT POS
1356 DOG2POSV *= *+1          DOG 2 VERT POS
1357 DOG3POSV *= *+1          DOG 3 VERT POS
1358 ;
1359 DOGFREQ1 *= *+1           # OF TIMES MAN MOVES BEFORE DOG DOES
1360 ;                       WHEN MAN NOT PULLING WALL
1361 DOGFREQ2 *= *+1             # OF TIMES MAN MOVES BEFORE DOG
1362 ;                       WHEN MAN PULLING WALL
1363 DOGMOVCT *= *+1           INDICATES WHEN DOG CAN MOVE
1364 DOGNO  =*  *+1           CURRENT DOG #
1365 ;
```



CASADAPTER
"IT REALLY WORKS!"

CASADAPTER is a cassette interface that allows you use your own cassette recorder or stereo with the Atari 400/800/1200®. **CASADAPTER** will handle motor control, audio and data channels.

$34.95

48K **RAM BOARD FOR: THE ATARI 400®.**

$115.00

SAR-AN
COMPUTER PRODUCTS

12 Scamridge Curve
Buffalo, New York 14221
(716) 632-3441

Dealer Inquiries Invited
*Product of Gemini Software

Add $2.50 shipping.
Send check or money order
C.O.D. accepted
New York State Residents
add 7% tax
®Atari is a trademark of
Atari, Inc.

***MAGIC DUMP** is a screen dump utility that allows you to dump a Hi-Resolution graphics picture to a printer in a variety of different sizes. **MAGIC DUMP** is used in the RIGHT hand cartridge slot, so it is always ready to use.

MAGIC DUMP will work with all Epson printers and Gemini printers, the Centronics 739 printer, and the Prowriter or N.E.C. printers. **$59.95**

***GTIA DRAW** is a drawing program that uses all the features of Atari's® new GTIA chip. **GTIA DRAW** will give you three extra graphics modes, Mode 9 (16 luminances and one color), Mode 10 (8 luminances and color), or Mode 11 (16 colors and one luminance).

GTIA DRAW will allow you to:
- Add text to pictures
- Blank horizontal or vertical lines
- Shift the screen in any direction
- 'ZOOM' in on certain areas of a picture
- Fill portions of a screen
- Work on two different screens simultaneously

$49.95

***THE DRUMESISER** is a unique sound synthesizing tool that allows the creation of sounds such as a drum, piano, organ, harpsichord, or electronic synthesizers. The different types of sounds created are limited by the user's imagination and ambition.

THE DRUMESISER comes with an Editor, which allows you to create your own sounds, a Player, which will allow you to play the different instruments, and the Memory Options, which make it possible for you to playback any sounds that you have recorded.

$49.95

ATARI

Software Gallery® MAIL ORDER

ATARI

WE'RE THE ONLY SOFTWARE AND COMPUTER STORE YOU'LL EVER NEED!
P.O. BOX 3354, CHERRY HILL, N.J. 08034

TOLL FREE (800) 992-3300

FOR ORDER ONLY

FOR INFORMATION & N.J. CALL (609) 596-1234

ATARI HOME COMPUTERS

Atari 600	$149
Atari 800 XL	Call
Atari 1200XL	Call
1050 Disk Drive	Call

1010 RECORDER	$74.95
1020 PRINTER	$264.95
1027 DAISY WHEEL PRINTER	$299.95
830 MODEM	$154.95
820 PRINTER	$254.95
CX419 BOOKKEEPER KIT	$194.95
CX481 ENTERTAINER PKG.	$64.95
CX482 EDUCATOR PKG.	$131.95
CX483 PROGRAMMER PKG.	$52.95
CX484 COMMUNICATOR PKG.	$324.95

SOFTWARE

ASTEROIDS	$25.95	STAR RAIDERS	$30.95
CENTIPEDE	$30.95	ATARI WRITER	$69.95
CAVERNS OF MARS	$30.95	DIG DUG	$34.95
DEFENDER	$30.95	QIX	$34.95
GALAXIAN	$30.95	MICKEY MOUSE	$37.95
MISSILE COMMAND	$25.95	PAINT	$34.95
PAC-MAN	$30.95	SUPERMAN	$39.95

Microsoft Basic 2 $82.95
Donkey Kong $37.95
Omnimon $81.95

PRINTERS

EPSON

RX-80	CALL
MX-80 FT	CALL
FX-80	CALL
FX-100	CALL

OKIDATA

82-A	$395
83-A	$634
84-P	$949
92	$495
93	$809

C-ITOH

PROWRITER	$369
PROWRITER 2	$645
STARWRITER	$1199
PRINTMASTER	$1545
GORRILA	$199

STAR

GEMINI 10	$314
GEMINI 15	$449

Smith Corona

TP-1	$525

DISKS

ELEPHANT
SS/SD (10) $18 SS/DD(10) $25

VERBATIM
CALL

MODEM

Hayes

Smart	$204.95
Smart 1200	$534.95
Chronograph	$199.95
Micromodem 100	$299.95
Micromodem II	$279.95

MONITORS

AMDEK (300)	$154.95
COLOR I	$295.00
GORILLA	$89.00

WICO

The Boss	$15.95
Joystick	$22.50
Famous Red Ball	$25.95
Track Ball	$49.95

RANA Disk Drive CALL

RAM BOARD
Intec

32K	$62.95
48K	$81.95
64K	$105.95
Clock	$29.95

COMPUTER COVER

$7.99 ea.

Features heavy duty canvas with vinyl interior - waterproof.

ATARI 800	$7.99
ATARI 810	$7.99
EPSON MX 80	$7.99
EPSON MX 80FT	$7.99
OKIDATA 182A	$7.99

In Home Key Board for 400 $54.95

Surge Protector

4 OUTLET	$37.95
6 OUTLET	$82.95

We carry a large selection of Atari software, call for FREE catalog with order ...or send $1 (refundable with order) UPS shipping 3% for hardware and $3 for software. N.J. resident add 6% sales tax.

VISA MasterCard

Note: Atari, Epson, IBM and Okidata are registered trademarks.

PRICES SUBJECT TO CHANGE WITHOUT NOTICE

MASTER CARD OR VISA ADD 3%

BOOT CAMP

AN ASSEMBLY LANGUAGE TUTORIAL COLUMN

by Tom Hudson

Welcome to the third installment of **Boot Camp**. As promised last issue, we're going to cover more 6502 instructions this time, and begin exploring the world of simple mathematical operations.

Before we start with the math operations, though, let's look at an instruction that will help us during the testing of the programs we write in this column.

BREAKing away.

Remember the do-nothing program from last issue? When we executed it with the "G" (execute program) command with the assembler editor cartridge, it ran forever. This is hardly a good way to test programs. Imagine trying to stop the program at a specific instruction with the BREAK key when hundreds of thousands of operations are being executed each second. You can see that this would be nearly impossible.

Luckily for us, the 6502 has a handy instruction called BRK (or BREAK). This instruction does the same thing as the BREAK key on the keyboard when an assembly program is executing. The nice part is that it will stop the program EXACTLY where we want it to stop.

The short program in **Figure 1** has a BRK instruction after the load accumulator (LDA) instruction. The accumulator will be loaded with $4F (79 decimal) and the computer will stop. Type the program into your computer and assemble it into memory with the ASM command.

```
10  *=   $0600    ;START ADDRESS
20  LDA  #$4F     ;LOAD ACCUMULATOR
30  BRK           ;AND STOP
40  .END
```

Figure 1.

After the program is assembled, go to the DEBUG mode with the BUG command. To execute the short program, type:

G 600

The program will execute in a fraction of a second and the computer will return with a display similar to **Figure 2.**

```
0602        A=4F X=00 Y=00 P=30 S=00
```

Figure 2.

Note that the accumulator (A) equals $4F. The X, Y, processor status and stack registers are also displayed, but have no significance to us at this time, since we didn't change them.

Now you can see that the BRK instruction can be helpful in the debugging stage of a program. We will be using it to stop the computer when we want to check the results of certain operations.

Using index registers.

Index registers were mentioned briefly last issue. As you may recall, there are two index registers in the 6502, the X and Y registers. These two registers are built into the 6502 microprocessor chip. Each is made up of 8 bits, allowing a range of values from 0-255.

The first instructions we'll look at are the LDX (load X) and LDY (load Y) instructions. These instructions are similar to the LDA (load Accumulator) instruction we examined last time. Their formats are:

LDX #n (IMMEDIATE)
LDX nn (ABSOLUTE)
LDX n (ZERO PAGE)

LDX nn,Y (INDEXED Y)
LDX n,Y (ZERO PAGE INDEXED Y)
LDY #n (IMMEDIATE)
LDY nn (ABSOLUTE)
LDY n (ZERO PAGE)
LDY nn,X (INDEXED X)
LDY n,X (ZERO PAGE INDEXED X)

The LDX and LDY instructions place a specified value in the X or Y register, respectively. For example, the following instruction will load the X register with $3A (58 decimal):

LDX #$3A

The following instruction will load the Y register with the contents of memory location $3F00:

LDY $3F00

The following instruction will load the X register from the page zero location $4D, which is the attract mode counter:

LDX $4D

Like the LDA instruction, both the LDX and LDY instructions set the sign and zero flags depending on the number loaded into the register.

Storing the contents of the X and Y registers is just as easy as loading them. The following addressing modes are available with the STX (store X) and STY (store Y) instructions:

STX nn (ABSOLUTE)
STX n (ZERO PAGE)
STX n,Y (ZERO PAGE INDEXED Y)
STY nn (ABSOLUTE)
STY n (ZERO PAGE)
STY n,X (ZERO PAGE INDEXED X)

Unfortunately for us, the designers of the 6502 decided to limit indexed store X and Y instructions to page zero, even though there is a non-zero page load instruction. This is simply something assembly programmers must live with.

Like the STA instruction, the STX and STY instructions do not affect any status flags.

The STX and STY instructions are very easy to use. For example, to store the X register at location $4FFB, simply use the instruction:

STX $4FFB

In addition to the LDX/LDY and STX/STY instructions, the 6502 provides four more instructions which help the programmer with X/Y operations. These are the TRANSFER instructions.

The transfer instructions allow quick movement of information from one register to another. They are TAX, TAY, TXA and TYA. Two other transfer instructions, TSX and TXS, are used in stack operations, and we'll look at them in a later article.

The TAX and TAY instructions transfer the contents of the Accumulator (A) to the X or Y register, respectively. The A register is unchanged.

Figure 3 illustrates how the TAX instruction works. Type this short program into your computer and assemble it into memory.

```
10  *=   $0600    ;START ADDRESS
20  LDA  #$0F     ;PUT $0F IN A
30  TAX           ;PUT IN X, TOO
40  LDA  #$6A     ;PUT $6A IN A
50  TAY           ;NOW PUT IN Y, TOO
60  BRK           ;AND STOP!
70  .END
```

Figure 3.

Line 20 loads the accumulator with $0F (15 decimal).

Line 30 transfers the contents of the accumulator to the X register. At this point both the accumulator and the X register will contain $0F.

Line 40 loads the accumulator with $6A (106 decimal).

Line 50 transfers the contents of the accumulator to the Y register. Now the accumulator and the Y register will contain $6A. The X register will be unchanged.

Line 60 will BREAK the execution of the program.

After the program in **Figure 3** is assembled into memory, go to DEBUG mode and execute it by typing:

G 600

After execution, the screen of your computer should look like **Figure 4.**

```
0606        A=6A X=0F Y=6A P=30 S=00
```

Figure 4.

You can see that the X register contains $0F and that the A and Y registers contain $6A. Try some different combinations and observe the results.

The two other transfer instructions we are concerned with here are the TXA and TYA instructions. As you may have guessed, these instructions do the opposite of the TAX and TAY instructions. That is, TXA will transfer the contents of the X register to the accumulator, and TYA will move the Y register's contents to the accumulator.

Here's a small problem for you to solve using the instructions we've covered so far. This is a simple data manipulation operation using the A, X and Y registers and as many locations as necessary.

PROBLEM: Write a program which starts with A=$03, X=$07 and Y=$14. Then write the code necessary to change these registers so that when the program ends, the registers are A=$07, X=14 and Y=$03.

The code necessary to perform this change is only four lines long, and there are many ways to do it. Next issue I'll show several possible solutions.

This issue, we've only shown how to make the X

and Y registers contain the values we want. In order to make the X and Y registers do some real work, we'll need to cover the branch-on-condition intructions. These will be discussed next issue, along with X and Y register indexing techniques.

It all ADDs up.

I'm sure that just about every person reading this column by now wants to start working with something more interesting than loading and storing bytes, right? Well, let's take a break from all that admittedly dull stuff and get on with something fun, actual addition.

We'll start out with some simple addition, working with values from 0-255. This is known as single-byte integer arithmetic, and is the simplest kind of math on the 6502.

Why only integers from 0-255? Remember that all arithmetic operations must be processed through the accumulator, or A register. The accumulator is made up of only 8 bits, and can't hold any number greater than 255. The accumulator doesn't know what a decimal point is, either, so we are limited to integers for the time being.

Binary or BCD?

The 6502 microprocessor has the option of performing arithmetic instructions in two different modes, BINARY and BINARY CODED DECIMAL (BCD). Let's look at how both these systems work.

Binary arithmetic, as we have noted before, produces numbers from 0-255 in one byte. All 8 bits are used for the number. These numbers can be considered either signed or unsigned by the programmer, but they are handled the same by the computer. Since all 8 bits are used to represent the number, the value of a byte is simply the byte's decimal contents.

BCD arithmetic, on the other hand, is a more human approach to computer math, and easier to use in input-output operations.

In BCD math, the byte is split into two 4-bit sections, or NYBBLES. Each nybble contains one decimal number, from 0-9. With this system, each byte contains two decimal numbers, allowing easy base-10 number storage. Of course, the BCD numbering system requires more storage than binary, since the value of a byte can now only range from 0-99, rather than 0-255. The nice thing about BCD is that when looking at the hexadecimal representation of the byte, you see the decimal value of the byte. For example, $56 is 56 decimal.

We'll cover BCD math later in this series, when we get into screen I/O. For now we'll stick with binary math. Even though it may seem more difficult, binary math is much more important at this early stage.

Getting into BINARY.

The 6502 can handle two different types of math, so how does it know which one you want to use? The answer lies in a single-bit flag in the processor status register, called the DECIMAL MODE flag.

The decimal mode flag has two states. When set (1), the decimal mode is selected. When cleared (0), the binary mode is selected. This flag is *extremely* important! The following example illustrates this fact.

Let's say you want to add two binary numbers, $23 and $18. A normal binary add would give a result of $3B.

What if the decimal mode flag was set by mistake? The add would give a result of $41, the sum of 23 and 18. If your program adds or subtracts numbers with the decimal mode incorrectly set, the results can be very confusing. Moral: ALWAYS know the setting of the decimal mode flag.

For our purposes, until further notice, we will always CLEAR the decimal mode with the CLD (clear decimal mode) instruction. The format of this instruction is:

```
CLD
```

This is a very simple instruction, but easy to forget. If you have trouble remembering things (like myself), I suggest that you tape an appropriate message to your monitor, computer, forehead, etc. This will save an incredible amount of debugging time.

Important: When writing assembly subroutines for BASIC programs, you *must* clear the decimal mode if you're doing any arithmetic in the subroutine. BASIC uses the floating-point arithmetic package built into the computer, which sets the decimal mode. The first time I wrote a BASIC assembly subroutine with math, it took me two days to find the problem. Once again, write a note.

Now that I've warned you about the evils of decimal mode ignorance, let's get on with some actual addition!

Add 'em up!

First we'll cover single-byte additions, the simplest kind. These types of additions are sufficient for general counters, changing color registers, or any operation in which the result will not exceed 255.

The 6502 has only one add instruction, ADC add with carry). This instruction has the following formats:

```
ADC #n      (IMMEDIATE)
ADC nn      (ABSOLUTE)
ADC n       (ZERO PAGE)
ADC (n,X)   (PRE—INDEXED INDIRECT)
ADC (n),Y   (POST-INDEXED INDIRECT)
ADC n,X     (ZERO PAGE INDEXED X)
ADC nn,X    (INDEXED X)
ADC nn,Y    (INDEXED Y)
```

The ADC instruction adds the number at the memory location specified in the operand to the accumulator and places the result in the accumulator. Depending on the result, the 6502 will alter the sign, overflow, zero and carry flags.

Let's look at a simple single-byte addition operation, using the immediate format. We will add 23 and 14 decimal and place the result in a location called ANSWER. **Figure 5** shows the code needed to perform this operation.

```
10  LDA  #23      ;PUT 23 IN A
20  CLC           ;CLEAR CARRY FOR ADD
30  ADC  #14      ;AND ADD 14 TO IT!
40  STA  ANSWER   ;SAVE RESULT
```

Figure 5.

The first line in **Figure 5** places the number 23 in the accumulator. Simple enough, right?

The second line introduces a new operation code, CLC (clear carry). The CLC instruction places a zero in the 6502 carry flag. This is an important instruction to remember, and should always be present in single-byte addition operations.

Why is the CLC instruction so important? The answer lies in the structure of the 6502 ADC instruction. Remember, ADC means "add with CARRY." Whenever an addition is performed on the 6502, the result is set to ACCUMULATOR + OPERAND + CARRY.

Here's an example of what can go wrong when the programmer is not sure of the contents of the carry flag. Let's say the carry happens to be set to 1. Fred the careless programmer wants to add 1+1 to verify that the answer is indeed 2, so he writes the following code:

```
LDA #1
ADC #1
STA ANSWER
```

When Fred runs the program, he is astounded to find that one plus one is three! If Fred had only inserted a simple CLC instruction, his life would have been much happier, as well as more accurate.

Suffice it to say that in any single byte addition operation, you should *always* clear the carry flag BEFORE the ADC instruction.

The third line adds 14 to the accumulator, giving a result of 37 ($25 hex), which is, of course, correct.

You can use any of the 8 addressing modes with the ADC instruction. All produce the same results, they just get their data with different methods.

Flag-waving.

Earlier I mentioned the flags altered by the ADC instruction. These are the sign, overflow, zero and carry.

The SIGN flag indicates the sign of the result. The contents of the accumulator's 7th bit are placed in this flag. If the flag is zero after an add, the result is considered positive. A one in this flag indicates a negative result. See Issue 13's **Boot Camp** for an in-depth discussion of the sign flag.

The OVERFLOW flag is set to the exclusive-or of bits 6 and 7 of the result. The overflow flag is rarely used, but it's a good idea to know what happens to it during processing.

The ZERO flag is set to one if the result of the add was zero, and is set to zero if the result was NOT zero.

The CARRY flag is set to one if the result of the add is greater than 255. This flag is important in multi-byte addition (for numbers greater than 255). We'll be examining multi-byte operations next issue.

All these flags are important in the computer's decision-making process. Depending on the result of an operation, the programmer can go to other parts of the program using comparison and branch instructions (similar to IF/THEN statements in BASIC). We will also cover these operations next issue.

Starting with subtraction.

Now that we've covered simple addition, let's do a little subtraction. Subtraction is just as easy as addition, with a couple of simple differences. Shown below are the formats of the 6502 subtraction instruction, SBC (subtract with borrow). You will notice that the SBC has the same formats as the ADC instruction.

```
SBC #n      (IMMEDIATE)
SBC nn      (ABSOLUTE)
SBC n       (ZERO PAGE)
SBC (n,X)   (PRE-INDEXED INDIRECT)
SBC (n),Y   (POST-INDEXED INDIRECT)
SBC n,X     (ZERO PAGE INDEXED X)
SBC nn,X    (INDEXED X)
SBC nn,Y    (INDEXED Y)
```

The SBC instruction subtracts the number at the memory location specified in the operand from the accumulator and places the result in the accumulator. Like the ADC instruction, the sign, overflow, zero and carry flags will be altered.

For the time being we'll work only with single-byte subtractions, since they're the easiest to understand. We will subtract 14 from 23 decimal and place the answer in a location called ANSWER. **Figure 6** shows the code needed for this operation.

```
10  LDA  #23      ;PUT 23 IN A
20  SEC           ;SET CARRY FOR SUB.
30  SBC  #14      ;AND SUB 14 FROM IT!
40  STA  ANSWER   ;SAVE RESULT
```

Figure 6.

The first line in **Figure 6** simply places the number 23 in the accumulator.

The second line introduces another new operation code, SEC (set carry). This instruction sets the carry flag to one. Like the CLC instruction in single-byte additions, the SEC instruction is a must for all single-byte subtractions.

The SBC instruction is strange in that it subtracts the contents of the memory byte indicated in the operand and the complement of the carry flag from

the accumulator, placing the result back in the accumulator. Here's an example. Let's say the accumulator contains 4 decimal, and we're subtracting 3 decimal from this. Assume the carry flag is clear (0). The computer will subtract 3 from 4, then subtract 1 from this (the complement of the carry flag), giving a result of zero.

By setting the carry to 1, we make sure that the subtraction of our two numbers is unaffected by the subtraction of the carry's complement, which in this case is zero. The carry flag is used as a borrow in subtraction and not necessary in single-byte operations.

The third line of **Figure 6** performs the subtraction. The result will be 23-14-0 or 9.

The last line of the program places the result in the location labeled ANSWER. The result will still be in the accumulator.

Like the ADC instruction, the SBC instruction works the same with all 8 addressing modes available with the instruction. The SBC instruction affects the 6502 status flags in the same way as ADC.

Applying what we've covered.

We've now progressed to the point where we can write simple math programs using addition and subtraction. Let's write a program to solve the equation:

4+5+34-(8-7) = ?

Unlike BASIC, we can't simply code this equation right into our computer. In assembly language, it's up to the programmer to figure out the procedure needed to obtain the result and code it.

Let's look at the equation shown above. In any mathematical equation, the expressions in parentheses must be solved before proceeding with the rest of the equation. If we simply solve the equation from left to right, we will get an incorrect answer:

4+5+34-8-7 = 28

In order to solve the equation correctly, we must solve it as follows:

(8-7) = 1

4+5+34-(1) = 42

Now that we know how to proceed, let's write a section of code to solve the equation. **Figure 7** shows one possible solution.

```
10  *=   $0600
20  CLD              ;NO DECIMAL MODE!
30  LDA #8           ;PUT 8 IN A
40  SEC              ;SET CARRY,
50  SBC #7           ;SUBTRACT 7 FROM 8
60  STA  HOLD        ;AND SAVE RESULT
70  LDA  #4          ;NOW PUT 4 IN A
80  CLC              ;CLEAR CARRY,
```

```
90  ADC  #5        ;ADD 4 & 5
0100  CLC           ;CLEAR CARRY AGAIN
0110  ADC  #34      ;ADD 34 TO LAST #
0120  SEC           ;SET CARRY,
0130  SBC  HOLD     ;SUBTRACT EARLIER #
0140  STA  ANSWER   ;AND SAVE ANSWER!
0150  BRK           ;ALL DONE!
0155 ;
0160 HOLD   *=*+1 ;TEMP. HOLD AREA
0170 ANSWER *=*+1 ;FINAL RESULT
0180  .END
```

Figure 7.

Line 10 tells the assembler to place the program at location $0600, a safe location in computer memory.

Line 20 clears the decimal mode, to avoid any accidental BCD results.

Line 30 places the number 8 in the accumulator.

Line 40 sets the carry flag to get ready for a single-byte subtract.

Line 50 subtracts 7 from 8, leaving the result in the accumulator.

Line 60 stores the result of the expression in parentheses at a memory location called HOLD. This is done because we will need this number in a moment.

Line 70 places a 4 in the accumulator in order to start solving the first part of the equation.

Line 80 clears the carry flag to get ready for a single-byte add.

Line 90 adds 5 to the accumulator, leaving the result in the accumulator.

Line 100 clears the carry again for the next addition. In this case, the CLC is not necessary since we know the previous add did not exceed 255, but it's a good idea to get into the CLC habit.

Line 110 adds 34 to the accumulator, once again leaving the result in the accumulator.

Line 120 sets the carry flag for the next subtract operation.

Line 130 subtracts the result of the expression in parentheses (stored in HOLD) from the accumulator and gets the final result.

Line 140 places the final result in the memory location called ANSWER.

Line 150 BREAKS the program execution. At this point the accumulator should equal 42 decimal ($2A hex).

Lines 160 and **170** set up the one-byte storage areas, HOLD and ANSWER. The assembler directive *=*+1 simply tells the assembler to reserve one byte for each label.

Line 180 tells the assembler that the end of the source code has been reached.

After this code is typed in and assembled into memory, execute the program from DEBUG mode with the command:

G 600

The program will execute very quickly and return with a screen similar to **Figure 8.**

```
0618          A=2A X=0F Y=6A P=31 S=00
```

Figure 8.

Note that the accumulator contains $2A (42 decimal). This is the correct answer to our equation.

This example shows how you can perform simple add-subtract operations in assembly language. Of course, we're limited to one-byte integers, but we'll soon exceed these limitations.

Until next time...

Try your own problems until you're proficient with the 6502 add and subtract operations. Try using the various addressing modes to see how they work. In order to learn assembly language (or any other language, for that matter), you'll have to roll up your sleeves and dig in.

Next issue will cover a lot of material, including the assembly equivalent of the BASIC IF/THEN statement, index register usage and multi-byte addition and subtraction. □

(continued from page 28)

speedster can drive over water and residents' lawns, or through houses and trees (something I could never accomplish in my Supra).

In the more exciting two player version, both cars remain on the screen simultaneously until one or the other eventually pulls ahead to the edge of the screen, thus penalizing his opponent. Competition can be fierce, and you'll find yourself gritting your teeth as you strive to speed on ahead of your pursuer.

Steering of the race is accomplished by joystick movements, with the fire-button acting as the brake. The number of laps defaults to 3 if you don't overide it with your own selection. An information readout at the bottom of the screen displays current lap time, last lap time and best time. On two player games it displays lap times of both players and the current lap number. On the one player game, you're alerted by a sound cue if you've beaten your best lap time.

In the two player game, the cars just bump into one another when they collide, rather than crash (a nice feature). Though the maximum speed of both cars will be identical, you'll find yourself pulling ahead or dropping behind depending on how you take the corners.

Two pre-defined tracks (complete with orchards, houses and ponds) are included in the program. However the neatest feature in the game is, perhaps, *Make Trax*. This option lets you define nearly any type of road course your imagination can come up with, plausible or not. And all of the scenery is there with it: trees, buildings, swamps; all kinds of neat things to hit. Your newly laid out course can be saved to cassette or disk, to be recalled at a later date.

A well designed menu screen allows you to choose and construct everything from mile-long straight-aways to barely negotiable curves, including intersections. This is the next step up from the Hot Wheels cars you had as a kid. (I've almost been tempted to lay out how some of the crazily designed New England roads should have been arranged!)

Rally Speedway (compatible with the 400/800/1200XL) is one of the nicest designed and executed games for the ATARI, and right up there with **Pole Position** as one of the best of the auto-race games. □

INDEX TO ANALOG COMPUTING ISSUES 1-14

by Brian Moriarty

ANALOG Computing's first monthly issue seemed like the ideal opportunity to publish our first general index. Those of you who've been with us since the beginning will enjoy many a nostalgic moment as you peruse these listings.

I've divided the articles into three major catagories: General-Interest Features, Product Reviews and Program Listings. The Listings category is further subdivided into Programming Aids, Disk Utilities, Entertainment and a Miscellaneous slush pile. Titles appear in chronological order, with all articles in a given issue listed alphabetically by author. Columns, reader comments and other regular sections of **ANALOG Computing** are not included.

We have a few copies of Issue 2 left in a closet somewhere, plus a limited supply of Issues 7, 8 and 10 through 14. The price is $4.00 per issue, which includes first class shipping and handling. Send check or money order to Back Issues Department, **ANALOG Computing** Magazine, P.O. Box 23, Worcester, MA 01603. □

Feature Articles

Author	Title	Issue	Page
Bachand, Charles	Parlez-Vous Pascal?	1	6
Pappas, Lee	How To Become A Star Commander	1	24
Walter, Russ	Graphically Speaking	1	28
Bachand, Charles	You're Wasting Arrays	2	12
Stewart, Ed	Unleash The Power Of Atari's CPU	2	20
Pappas, Lee	Program In Style	3	44
LaFerla, Robert	Player/Missile Graphics	5	33
Latino, Bill	Is There A Laser Disk In Your Future?	5	20
Patchett, Craig	Atari Display Lists	5	43
Rowland, Rick	Asteroids Controller	5	62
Smith, George	More On PEEKs and POKEs	5	12
Roberts, B.B.	Pirating!	6	39
Meyer, Robert	A Low-Bucks Memory Upgrade	6	68
Kelley, Patrick	Atari: 2019	7	12
Knoble, Peggy	BASIC Crossword Puzzle	7	71
Tillman, Raymond	Atari BASIC Meets Complex Data Structures	7	49
Sueirro, Michael	Restore Your Mental Health	7	19
Hough, William	Mixing GTIA and CTIA Graphics	8	36
Kalagher, Richard	On Converting To Atari Microsoft BASIC	8	77
Krischan, Thomas	Beginner's BASIC	8	54
Rosen, Marc	Audio In Your Programs	8	27
Seftor, Larry	Missing Capabilities In Atari BASIC	8	88
Sueirro, Michael	Relieve Your Floating Point Blues	8	44
Bell, Jon	A Trip To The Nimitz	9	89
Editorial	Whither Atari?	9	4
Layne, Darren	Left-Handed Joysticks	9	23
Meyer, Robert	Build A Keyboard For Your 400	9	102
Pappas, Lee	Atari In 1983	9	117
Strauss, Daryll	Have The Computer Type It In	9	119
Volk, William	Go FORTH!	9	42
Griffin, Brad	The Challenge Of The Quest	10	56
Herring, Richard	Installing Your Own GTIA Chip	10	94
Hough, William	Accessing Bulletin Board Systems	10	49
Krischan, Thomas	Beginner's PILOT	10	113
Krischan, Thomas	Epson Printing Modes	10	61
Lever, Kevin	Using A Percom Drive With DD Disks	10	92
Litkowski, Ken	A Lisp Tutorial	10	85
Krischan, Thomas	BASIC Programming Techniques	11	20
Krischan, Thomas	Copyright Basics	11	66
Lyons, Richard	Strings In Atari BASIC	11	35
Staff	The 1983 West Coast Computer Faire	11	12
Stephens, Jessie	What Software Reviewers Are Saying	11	46
Caruthers, Marcy	Atari Word Search	12	45
Despain, Gerald	Notes To The Atari Microsoft BASIC Manual	12	48
Leyenberger, Arthur	De Re Letter Perfect	12	43
Caruthers, Marcy	Atari Graph	13	47
Peacock, Kyle	Fine Scrolling, Part 1	13	113
Peacock, Kyle	Fine Scrolling, Part 2	14	25

Product Reviews

Program Listings

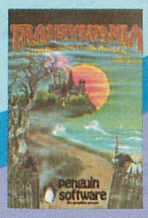
penguin software™

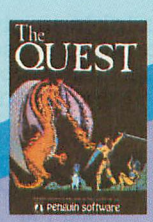
the graphics people

The Next Generation Is Here!

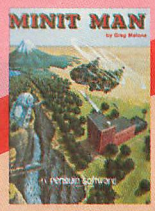
Strategy

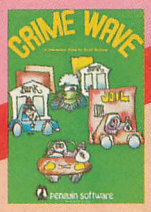
Exciting Adventures

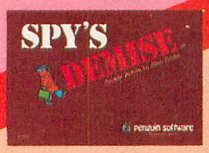




Arcade Fun







Fantasy Role Playing

Games for Apple, Atari, Commodore, and IBM computers

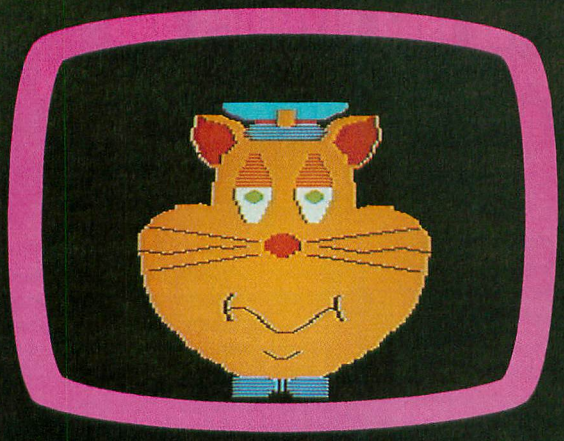
At Penguin, we don't rely on the same old game formulas, adding to the mish-mash of look-alike games already on the market. We look for creativity, originality, and innovation. Games like **Minit Man,** challenging you with two types of arcade action simultaneously; **Coveted Mirror,** an adventure with animation and arcade games throughout; **The Spy Strikes Back,** an arcade game with strategy and a touch of adventure; **Pensate** and **Tactic 9,** games that make you think; and **Expedition Amazon,** an intriguing and humorous fantasy game.

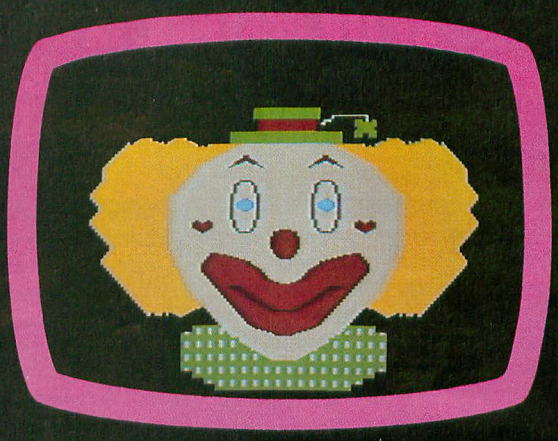
And Each Is Only $19.95! More Fun For Your Money From Penguin Software

Write for a free catalog

830 Fourth Avenue, Box 311 Dept. G Geneva, IL 60134 For information - Call (312) 232-1984 Dealer orders only - Call (800) 323-0116

SOFTWARE MOVIES™





NEW!

SOFTWARE DESIGNS FOR DEVELOPING YOUNG MINDS™

MAXIMUS, Inc. is excited to present two new educational software movies for kids of all ages . . . and more are on the way!

STORYLINE™ makes bedtime, or anytime, a warm and friendly funtime. Clover the Clown is your tourguide for two fairytales: The Ugly Duckling and Rumpelstiltskin. Through computer magic you become a part of each story!

SAFETYLINE™ combines fun with important lessons about safety. Max the Cat shows you how to cross the street safely going to school and what to do when lost at the zoo.

Both software movies have interactive games that reinforce the fun and learning. Both have real human voices; you don't need a voice synthesizer!

Features of the software include:

- Sophisticated Face Animation
- Lip-Synchronized Voices
- Original Musical Scores
- Superior Full-Color Graphics
- Optional Joystick Control
- Machine Language Programs

Relax and enjoy yourself. Let Max, Clover, and their colorful friends guide you through a new world of fun and learning. You'll be amazed at how real they are!

STORYLINE™ and SAFETYLINE™ are currently available for any Atari computer with 48K. Each software movie comes in either CASSETTE ONLY ($29.95) or CASSETTE/DISK ($34.95) versions. Order by mail or phone. Visa and Mastercard welcome. Include $2.50 postage and handling; Virginia residents add 4% sales tax. Dealer inquiries invited. Write for our free catalog. MAXIMUS, Inc., 6723 Whittier Avenue, McLean, Virginia 22101.

ORDER TOLL-FREE 1-800-368-2152

MAXIMUS ...*Where Excellence is the Standard*

Commodore 64 Version Coming Soon!

PMI CELEBRATES THE OLYMPICS!!!

PURCHASE *ANY* PMI PROGRAM AND WE WILL GIVE YOU, ABSOLUTELY *FREE, NORDIC SKI,* A 100% MACHINE LANGUAGE ARCADE-STYLE GAME BY LEN DORFMAN, IN HONOR OF THE 1984 OLYMPICS *AND* PMI WILL DONATE $1.00 TO THE U.S. OLYMPIC TEAM FOR EACH PROGRAM SOLD!! HURRY! THIS OFFER IS FOR A LIMITED TIME ONLY!

WAMPYR'S TOMB

WARNING!! This adventure may be addictive. WAMPYR'S TOMB, by James Glines, will take you on a journey from the village of Amberpool through a 'mid-evil' countryside filled with magical spells, demons and dragons. Can you stay alive? To prevail in WAMPYR'S TOMB you must do more! You must equip yourself to do battle with the LICH and win. (Of course there is the option of dying!) We will supply you with a little money and the opportunity to go where you've never been; the courage and the desire to conquer the unknown must come from within. WAMPYR'S TOMB will not only challenge your bravery, your intelligence and your cunning, but will take you beyond your imagination into a realm of new and exciting forces where only legends survive. Requires 48K RAM and one disk drive.

$29.95

SWORD POINT

SWORD POINT, by William A. Austin, is a very unique and exciting adventure. You will assume the role of a young man seeking to climb to the pinnacle of success in the mythical monarchy of Simafranconia, a nation not unlike feudal 17th Century Europe. It is a country of totally masculine values, where style supersedes substance, and social level is not the best thing, it is the only thing worth acquiring. Dueling, gambling, wenching and high living are among a few of the things that you will do to gain the King's attention. So tankards up ...a toast to bravery...with a little luck and by dancing to the daredevil's tune on the Swords Point...you may just win! Requires 48K RAM and one disk drive.

$31.95

PROGRAMMERS!

If you have written a program that is technically sophisticated and also "user-friendly", PMI would be interested in publishing your work. For information on how to submit your material write to: PMI, P.O. Box 2895, Winter Park, FL 32790-2895, Or phone (305) 644-3822.

COM-CON

Have you ever wondered what your computer would have to say about that party you had last weekend? Or the shape the world is in today? Well now you can find out. *COM-CON* has arrived! *COM-CON* by William Austin is an unique, exciting and entertaining program for everyone. Your computer can answer your questions on virtually any topic. While carrying on a discussion with *COM-CON* you will notice that your computer actually has a "personality" of its own. And like any human being the answers will sometimes be as vague and nebulous as a political candidate. And at other times quite philosophical. If you're 4 to 104 *COM-CON* is for you. *COM-CON* is available for the Votrax Type 'n Talk, $31.95, the Alien Group Voice Box $31.95, *SAM* $31.95, and in a text only version $26.95. Requires *48K RAM, BASIC* cartridge, one disk drive and optional voice synthesizer.

KARMIC CAVERNS

KARMIC CAVERNS, by Len Dorfman, is a 100% machine language arcade-style game that will entertain you hour after hour! You must find your way through the maze-like caverns avoiding the deadly plasmatic guards (not to mention the electrified walls) and acquire as many energy pods as you can. You must climb ever upwards striving to acquire enough energy points to reach the next level before time runs out for you. KARMIC CAVERNS utilizes the full sound and graphics capabilities of the Atari computer! Fun for all ages. Requires 48K RAM and one disk drive.

$34.95

ERG NEW

ERG, another 100% machine language arcade-style game by Len Dorfman, is the ultimate maze escape challenge!! Multiple skill levels, multiple mazes, roving energy fields which will drain your energy and proton barriers which cause disintegration upon contact are the obstacles you will face. If you are an expert strategist and your reflexes are honed to a razor's edge (i.e., you can pull a gnat's whisker out as he flies by) the odds are about even. You might just win! Requires *48K RAM* and one disk drive.

$34.95

RESTORE

List the unlistable! *RESTORE* by William Fletcher will allow you to make a *LISTABLE, EXECUTABLE* copy of any *BASIC* program. *RESTORE* is extremely user-friendly and the simplicity of operation is astounding. *RESTORE* is a compiled program and executes at machine language speed. Utilizing the graphics and sound capability of the Atari, *RESTORE* is not only an indispensible programmer's tool, but is fun to use. If you have two disk drives *RESTORE* will automatically seek out relocated directories and give you the option to either restore all *BASIC* programs on the disk or make a copy of the disk putting the directory in its normal location. Requires *48K RAM* and one or more disk drive.

$27.95

PROGRAMMIN' STUFF — $26.95
VARICOMMANDER — $29.95
PROGRAMMIN' STUFF/VARICOMMANDER PACKAGE — $45.95
WORDPOWER — $27.95
BIOGRAPH — $24.95

PMI, P.O. Box 2895, 2500 Lee Road, Suite 210, Winter Park, FL 32790-2895

☐ *SWORD POINT* ☐ *KARMIC CAVERNS* ☐ *WAMPYR'S TOMB*
☐ *PROGRAMMIN' STUFF* ☐ *T.I.P.S.* ☐ *BIOGRAPH* ☐ *VARICOMMANDER* ☐ *RESTORE* ☐ *ERG*
☐ *COM-CON* ☐ *VARICOMMANDER and PROGRAMMIN' STUFF PACKAGE* ☐ *WORDPOWER*

Total $ _____ , plus $3.00 postage & handling (Florida residents add 5% sales tax)

☐ CHECK (allow 3 weeks for personal checks to clear) ☐ MONEY ORDER
☐ C.O.D. (add an additional $2.00 for C.O.D. orders)

NAME (print) _____

Address _____

City _____ State _____ Zip _____

S.A.M. is a trademark of DON T ASK

DEALER INQUIRIES INVITED

ATARI is a trademark of ATARI INC

INDEX TO ADVERTISERS

NEXT ISSUE:
SPECTACULAR GRAPHICS!

- Shooting Stars
- Fine Scrolling Part IV
- 3-D Rotation
- Create-A-Font
- Stars 3-D
- Bar Chart Subroutine

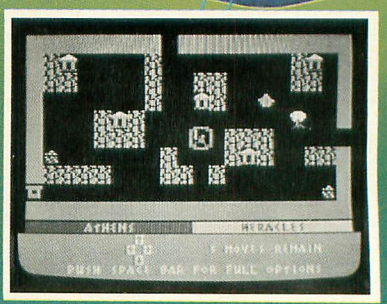
Coming in the February ANALOG Computing.

The computer adventure you've been waiting for...

THE RETURN OF HERACLES™

An exploration of Greek mythology translated into modern electronics
by Stuart Smith

The creative mind of Stuart Smith, author of ***Ali Baba and the Forty Thieves,*** brings to life the world of mythic Greece in fantastic color and sound! Carefully researched and skillfully programmed, ***Return of Heracles*** is computer entertainment at its best. Twelve difficult and dangerous tasks will be assigned to you by Zeus, and your heroes must accomplish them all. One or more players take on the role of an ancient Greek hero or heroine. There are 19 heroes to choose from, or choose them all! May the gods favor you!



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

For Atari home computers.
On diskette. Requires 48K
$32.95

Correction.

It's simple.

New **Spell Perfect**™ from LJK does what your second grade teacher wanted to do. It corrects your spelling. Used with Letter Perfect™, it checks your work against an expandable dictionary. Gives "sounds like" suggestions. Counts words. Add words. Technical terms. States and towns.

You can do it all, all-in-one dictionary.

When it comes to practical software for Ataris, Apples and look alikes, Spell Perfect is simple to learn. And hard to beat. Ask your dealer for a demonstration, or write LJK for more information.

LJK, INC. • 7852 Big Bend Blvd., St. Louis, MO 63119 • (314) 962-1855

YOU'LL LIKE IT!

ACTUALLY, YOU'LL LOVE IT!

Here's why! TRIAD is nine separate pulse-pounding games in one package. Select one of nine weird alien menaces to battle from the Master Board. But careful — these guys can be tricky! Defeat three aliens in a row tic-tac-toe style and you earn a crack at the next skill level.

The whole neighborhood'll thrill to TRIAD's crisp colors, super sounds and the oddest assortment of alien "baddies" ever assembled on one screen. Snappy arcade excitement for all ages!

TRIAD — from Adventure International. Why get one game when you can have nine?

TRIAD — YOU'LL ~~LIKE~~ *love* IT!

(152)

APPLE (DOS 3.3 REQUIRED) 48K DISK $34.95
ATARI 48K DISK $34.95
COMMODORE 64 TAPE & DISK $34.95
TRS-80 CoCo 32K TAPE $34.95

ANOTHER WINNER FROM

Adventure® INTERNATIONAL
A DIVISION OF SCOTT ADAMS, INC.

P.O. BOX 3435 • LONGWOOD, FL 32750
1-800-327-7172